В.В. Липаев

ПРОГРАММНАЯ ИНЖЕНЕРИЯ

МЕТОДОЛОГИЧЕСКИЕ ОСНОВЫ

ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ — ВЫСШАЯ ШКОЛА ЭКОНОМИКИ

В.В. ЛИПАЕВ

ПРОГРАММНАЯ ИНЖЕНЕРИЯ

МЕТОДОЛОГИЧЕСКИЕ ОСНОВЫ

Допущено УМО по образованию в области менеджмента
в качестве учебника для студентов высших учебных заведений,
обучающихся по направлению «Бизнес-информатика» (080700)

Москва
2006

УДК 004.41(075.8)
ББК 32.973.26–018я73
Л42

Издание осуществлено в рамках
Инновационной образовательной программы ГУ ВШЭ
«Формирование системы аналитических компетенций
для инноваций в бизнесе и государственном управлении»

Рецензенты:
заместитель директора МСП РАН, доктор технических наук,
профессор *В.З. Шнитман*;
проректор по информатизации ГОУ МГТУ «Станкин»,
доктор технических наук, профессор *Б.М.Позднеев*

ISBN 5-7598-0424-3

СОДЕРЖАНИЕ

ПРЕДИСЛОВИЕ

Основная цель курса лекций — представить студентам, аспирантам и менеджерам проектов современный комплекс задач, методов и стандартов ***программной инженерии*** — создания и развития сложных, многоверсионных, тиражируемых программных средств (ПС) и баз данных (БД) требуемого высокого качества. Изложение ориентировано на коллективную, групповую работу специалистов над крупными программными проектами. Внимание акцентировано ***на комплексе методов и процессов,*** которые способны непосредственно обеспечить ***эффективный жизненный цикл сложных высококачественных программных продуктов и баз данных***. При этом предполагается, что процессы и технология создания комплексов программ и документов опираются на совокупность современных, автоматизированных методов и инструментальных средств поддержки длительного жизненного цикла программных продуктов. Однако не всегда это может быть рентабельно вследствие высокой стоимости таких средств. В результате может снижаться качество программных продуктов и повышаться их стоимость.

Быстрый рост областей применения, сложности функций и масштабов комплексов программ привел к принципиальному изменению методов в этой сфере и к переходу от технологии индивидуального программирования отдельных небольших программок к коллективному созданию крупных комплексов программ инженерными методами проектирования и разработки. Накопление в мире знаний, опыта разработки и применения огромного количества различных сложных программ для ЭВМ способствовало систематизации и обобщению методов и технологий их разработки, сокращению дефектов и неопределенностей в характеристиках и качестве поставляемых и применяемых программных продуктов. В результате сформировалась ***современная методология и инженерная дисциплина*** обеспе-

чения процессов жизненного цикла сложных программных продуктов — ***программная инженерия*** для различных областей применения.

Программная инженерия — это область компьютерной науки и технологии, которая занимается построением программных систем, настолько больших и сложных, что для этого требуется участие слаженных команд разработчиков различных специальностей и квалификаций. Обычно такие системы существуют и применяются долгие годы, развиваясь от версии к версии, претерпевая на своем жизненном пути множество изменений, улучшение существующих функций, добавление новых или удаление устаревших возможностей, адаптацию для работы в новой среде, устранение дефектов и ошибок. ***Суть методологии программной инженерии*** состоит в применении систематизированного, научного и предсказуемого процесса проектирования, разработки и сопровождения программных средств.

Массовое создание сложных программных средств промышленными методами и большими коллективами специалистов вызвало необходимость их четкой организации, планирования работ по требуемым ресурсам, этапам и срокам реализации. Совокупные затраты в мире на такие разработки составляют миллиарды, а для отдельных проектов — миллионы долларов в год, поэтому требуется тщательный анализ экономической эффективности создания и использования конкретных ПС. Для решения этих задач в программной инженерии формируется новая область знания и научная дисциплина — ***экономика жизненного цикла программных средств***, как часть экономики промышленности и вычислительной техники в общей экономике государств и предприятий. Объективно положение осложнено трудностью измерения экономических характеристик таких объектов. Широкий спектр количественных и качественных показателей, которые с различных сторон характеризуют содержание этих объектов, и невысокая достоверность оценки их значений определяют значительную дисперсию при попытках описать и измерить экономические свойства создаваемых или используемых крупных ПС.

Вследствие роста сфер применения и ответственности функций, выполняемых программами, резко возросла необходимость ***гарантирования высокого качества программных продуктов***, регламентирования и корректного формирования требований к характеристикам реальных комп-

лексов программ и их достоверного определения. В результате специалисты в области теории и методов, определяющих качество продукции, вынуждены осваивать область развития и применения нового, специфического продукта — программных средств и систем в целом и их качество при использовании. Сложность анализируемых объектов — комплексов программ и психологическая самоуверенность ряда программистов в собственной «непогрешимости» часто приводят к тому, что реальные характеристики качества функционирования программных продуктов остаются неизвестными не только для заказчиков и пользователей, но также для самих разработчиков. Отсутствие четкого декларирования в документах понятий и требуемых значений характеристик качества ПС вызывает конфликты между заказчиками-пользователями и разработчиками-поставщиками из-за разной трактовки одних и тех же характеристик.

Методы программной инженерии поддерживают и конкретизируют ***технологический процесс***, а также отслеживание значений качества компонентов на этапах жизненного цикла ПС. Для каждого проекта, выполняющего ответственные функции, должны разрабатываться и применяться ***система качества***, специальные планы и Программа, методология и инструментальные средства разработки и испытаний, обеспечивающие ***требуемые качество, надежность и безопасность функционирования программных продуктов***. Эти методы и процессы позволяют разработчикам и заказчикам программных продуктов более корректно взаимодействовать при определении и реализации требований контрактов и технических заданий.

Основные концепции программной инженерии сконцентрировались и формализовались в ***целостном комплексе систематизированных международных стандартов***, охватывающих и регламентирующих практически все процессы жизненного цикла сложных программных средств. Несколько десятков стандартов этого комплекса допускают целеустремленный отбор необходимых процессов, в зависимости от характеристик и особенностей конкретного проекта, а также формирование на их базе проблемно-ориентированных ***профилей стандартов*** для определенных типов проектов и/или предприятий.

Практическое применение профилей стандартов, ***сосредоточивших мировой опыт*** создания различных типов крупных комплексов программ,

способствует значительному ***повышению производительности труда*** специалистов ***и качества*** создаваемых программных продуктов. Эти стандарты определяют модификацию, мобильность и возможность повторного применения программных компонентов и комплексов, их расширяемость и переносимость на различные аппаратные и операционные платформы, что непосредственно отражается на росте экономической эффективности технологий и процессов создания различных программных средств и систем. Для регламентирования процессов жизненного цикла такие профили стандартов должны ***адаптироваться*** и конкретизироваться применительно к определенным классам и функциям проектов, процессов и компонентов программных средств. При этом должны сохраняться концептуальная целостность применяемой совокупности стандартов и их эффективное, положительное влияние на процессы и результаты, на качество, надежность и безопасность программных продуктов ***при реальных ограничениях на использование доступных ресурсов проектов***.

В ***жизненном цикле комплексов программ*** сложно сочетаются содержание, этапы и распределение работ, возможен ряд возвратов на более ранние технологические этапы в процессе создания компонентов ПС, они имеют не совсем определенные границы начала и завершения. Специалисты в коллективе могут на некотором интервале времени решать несколько производственных задач и заменять друг друга. Положение усугубляется трудностью поэтапного определения качества компонентов и его прогнозирования в процессе разработки, что непосредственно отражается на проекте в целом. ***Методология программной инженерии*** и стандарты ***регламентируют современные процессы управления проектами*** сложных систем и программных средств. Они обеспечивают организацию, освоение и применение апробированных, высококачественных процессов проектирования, программирования, верификации, тестирования и сопровождения программных средств и их компонентов. Тем самым эти проекты и процессы позволяют получать стабильные, предсказуемые результаты и программные продукты требуемого качества.

Многообразие классов и видов сложных комплексов программ, обусловленное различными функциями и сферами применения систем, определяет формальные трудности, связанные с методами и процедурами ***доказательства соответствия создаваемых и поставляемых программ-***

ных продуктов условиям контрактов, требованиям заказчиков и потребителей. По мере расширения применения и увеличения сложности систем выделились области, в которых дефекты, недостаточное качество комплексов программ или данных могут наносить катастрофический ущерб, значительно превышающий положительный эффект от их использования. В таких ***критических системах*** (например, управления атомными электростанциями, крупными банками или системами вооружения) ***недопустимы проявления катастрофических рисков функционирования*** программных продуктов при любых искажениях исходных данных, сбоях, частичных отказах аппаратуры, ошибках пользователей и других нештатных ситуациях. Подобные риски комплексов программ могут определять безопасность функционирования объектов, предприятий и даже страны. Вследствие этого резко ***повысилась ответственность специалистов за качество*** результатов их труда и создаваемых программных продуктов. Это требует непрерывного совершенствования, обучения и повышения квалификации заказчиков, разработчиков и пользователей в области программной инженерии, освоения ими современных методов, процессов и международных стандартов, а также ***высокой корпоративной культуры коллективов специалистов***, обеспечивающих жизненный цикл критических программных продуктов.

В ***курсе лекций предполагается***, что проектирование, разработка, сопровождение и документирование программных продуктов ведется преимущественно с использованием регламентированных процессов на основе профилей стандартов **ISO** в соответствии с формализованными требованиями, определенными заказчиком. Тем самым не отражено создание и применение ***открытого кода*** в программных продуктах, при котором невозможна их сертификация на соответствие международным стандартам, вследствие неоднозначности версий ПС, изменяемых несинхронно различными пользователями и различными предприятиями.

Значительное внимание в курсе методологии программной инженерии уделено фрагментам и компонентам профилей стандартов **ISO**, целесообразным для обеспечения высокого качества и безопасности применения программных продуктов в их жизненном цикле. Менеджмент рассматриваемых проектов ориентирован преимущественно на базовые стандарты серии **ISO 9000:2000**. Стандарты де-факто менеджмента **CMM / CMMI**

изложены как возможная альтернатива стандартам **ISO** при управлении проектированием ПС, однако они не покрывают все процессы жизненного цикла сложных комплексов программ. Наиболее распространенная сертификация предприятий, производящих программные продукты, на соответствие стандартам **CMM / CMMI** существенно проще, чем сертификация на соответствие полному профилю стандартов всего жизненного цикла **ISO**. При применении моделей **CMM / CMMI** учитывается преимущественно качество менеджмента проектов, однако не стандартизируются и не контролируются некоторые важные компоненты процессов ЖЦ: регламентированные характеристики качества ПС; интерфейсы Открытых систем; функциональная и информационная безопасность; комплекс технологической и эксплуатационной документации. Это может быть оправдано для предприятий, создающих программные продукты средней или относительно невысокой сложности, и вряд ли допустимо для разработчиков крупных, особо сложных, критических ПС. Акцент методологии программной инженерии на крупные проекты предполагает сохранение и возможность применения ее базовых методов, процессов и стандартов при обеспечении жизненного цикла относительно небольших проектов. Для этого должна быть предусмотрена и использоваться ***адаптация*** стандартов, технологий и инструментария для различных по сложности и размеру проектов.

В публикациях в области программной инженерии широко распространено изобретение многочисленных аббревиатур на основе английских, иногда полужаргонных, слов, которые выдаются за принципиальные достижения авторами публикаций. В тексте лекций для облегчения восприятия учащимися избегается применение аббревиатур, за исключением небольшого их числа, на основе часто используемых словосочетаний русских слов. В официальных документах в имени стандартов **ISO** часто присутствуют через слеш имя **IEC** (Международная электротехническая комиссия). Для сокращения и упрощения текста лекций имя **IEC** всюду опускается, однако его следует учитывать при использовании и формировании официальных документов.

Предлагаемый ***курс основ методологии программной инженерии*** предназначен для заказчиков, менеджеров крупных проектов, для аналитиков и ведущих специалистов, обеспечивающих все этапы жизненного

цикла крупных программных средств и систем. Лекции рекомендуется использовать при обучении ***по специальности «бизнес-информатика»***, студентов старших курсов, аспирантов и менеджеров проектов, созданию сложных комплексов программ на всем их жизненном цикле. Курс полезен исполнителям научных проектов и опытно-конструкторских работ, к которым предъявляются высокие требования к качеству функционирования и ограничены доступные ресурсы разработки программных продуктов. Материалы лекций могут служить базой при подготовке и внедрении систем качества предприятий, создающих сложные конкурентоспособные программные продукты, для их сертификации на соответствие международным стандартам.

Считаю необходимым выразить особую благодарность профессору, заведующему кафедрой Программной инженерии ГУ ВШЭ Сергею Михайловичу Авдошину за весьма полезные замечания к рукописи книги, а также за активную поддержку и организацию издания учебного пособия.

ЛЕКЦИЯ 1

ПРОГРАММНАЯ ИНЖЕНЕРИЯ В ЖИЗНЕННОМ ЦИКЛЕ ПРОГРАММНЫХ СРЕДСТВ

1.1. Основы жизненного цикла программных средств

Термином ***жизненный цикл*** (ЖЦ) принято отражать совокупность процессов и этапов развития организмов живой природы, технических систем, продуктов производства от моментов зарождения или появления потребности их создания и использования до прекращения функционирования или применения. Это соответствует всеобщему закону развития любых изделий, событий или процессов между их началом и концом, которые определяют цикл их создания, существования и применения. Программы для вычислительных машин обычно являются компонентами жизненного цикла технических систем, но по своей природе значительно отличаются от аппаратурных, технических изделий, поэтому их жизненный цикл имеет характерные особенности по сравнению с другими техническими объектами. Программы и данные в системах и вычислительных машинах являются наиболее ***гибкими компонентами программной инженерии и подвержены изменениям*** в течение всего их ЖЦ.

Типовая модель процессов жизненного цикла сложной системы начинается с концепции идеи системы или потребности в ней, охватывает проектирование, разработку, применение и сопровождение системы и заканчивается снятием системы с эксплуатации. Программные средства служат для выполнения определенных ***функций систем*** на компьютерах. Модель жизненного цикла системы обычно разделяют на последовательные периоды реализации — ***стадии или этапы***. Каждый подобный пери-

од включает основные реализуемые в нем процессы, работы и задачи, при завершении которых может потребоваться переход к следующему периоду реализации. Общую модель жизненного цикла сложной системы обычно разделяют на следующие основные этапы с последующей адаптацией каждого из них в модели ***жизненного цикла конкретной системы***:

— определение потребностей;
— исследование и описание основных концепций;
— проектирование и разработка;
— испытания системы;
— создание и производство;
— распространение и продажа;
— эксплуатация;
— сопровождение и мониторинг;
— снятие с эксплуатации (утилизация).

По ***особенностям и свойствам жизненного цикла*** программ их целесообразно делить на ряд классов и категорий, из которых наиболее различающимися являются два крупных класса — ***малые и большие***.

Первый класс составляют относительно небольшие программы, создаваемые одиночками или небольшими коллективами (3—5) специалистов, которые:

— создаются преимущественно для получения конкретных результатов автоматизации научных исследований или для анализа относительно простых процессов самими разработчиками программ;

— не предназначены для массового тиражирования и распространения как программного продукта на рынке, их оценивают качественно и интуитивно преимущественно как «художественные произведения»;

— не имеют конкретного независимого заказчика-потребителя, определяющего требования к программам и их финансирование;

— не ограничиваются заказчиком допустимой стоимостью, трудоемкостью и сроками их создания, требованиями заданного качества и документирования;

— не подлежат независимому тестированию, гарантированию качества и/или сертификации.

Для таких, а также для многих других видов относительно несложных программ нет необходимости в регламентировании их жизненного

цикла, в длительном применении и сопровождении множества версий, в формализации и применении профилей стандартов и сертификации качества программ. Их разработчики не знают и не применяют регламентирующих, нормативных документов, вследствие чего жизненный цикл таких изделий имеет ***непредсказуемый характер*** по структуре, содержанию, качеству и стоимости основных процессов «творчества».

Второй класс составляют крупномасштабные комплексы программ для сложных систем управления и обработки информации, оформляемые в виде ***программных продуктов*** с гарантированным качеством, и отличаются следующими особенностями и свойствами их жизненного цикла:

— большая размерность, высокая трудоемкость и стоимость создания таких комплексов программ определяют необходимость тщательного анализа экономической эффективности всего их жизненного цикла и возможной конкурентоспособности на рынке;

— от заказчика, финансирующего проект программного средства и/или базы данных, разработчикам необходимо получать квалифицированные конкретные требования к функциям и характеристикам проекта и продукта, соответствующие выделенному финансированию и квалификации исполнителей проекта;

— для организации и координации деятельности специалистов-разработчиков при наличии единой, крупной целевой задачи, создания и совершенствования программного продукта необходимы квалифицированные менеджеры проектов;

— в проектах таких сложных программных средств и баз данных с множеством различных функциональных компонентов участвуют специалисты разной квалификации и специализации, от которых требуется высокая ответственность за качество результатов деятельности каждого из них;

— от разработчиков проектов требуются гарантии высокого качества, надежности функционирования и безопасности применения компонентов и поставляемых программных продуктов, в которые недопустимо прямое вмешательство заказчика и пользователей для изменений, не предусмотренных эксплуатационной документацией разработчиков;

— необходимо применять индустриальные, регламентированные стандартами процессы, этапы и документы, а также методы, методики и комплексы, средства автоматизации, технологии обеспечения жизненного цикла комплексов программ.

Такие крупномасштабные комплексы программ ***являются компонентами систем***, реализующими обычно их основные, функциональные свойства, увеличивающими сложность и создающими предпосылки для последующих изменений их жизненного цикла. Реализация ЖЦ, методологии управления и изменения ПС зависит от многих факторов, от персонала, технических, организационных и договорных требований и сложности проекта. Множество текущих состояний и модификаций компонентов сложных ПС менеджерам необходимо упорядочивать, контролировать их развитие и применение участниками проекта. Организованное, контролируемое и методичное отслеживание динамики изменений в жизненном цикле программ и данных, их слаженная разработка при строгом учете и контроле каждого изменения являются основой эффективного, поступательного развития каждой крупной системы ***методами программной инженерии***.

Существует ***множество моделей процессов жизненного цикла систем и программных средств***, но три из них ***в международных стандартах*** обычно квалифицируются как фундаментальные: каскадная; инкрементная; эволюционная. Каждая из указанных моделей может быть использована самостоятельно или скомбинирована с другими для создания гибридной модели жизненного цикла конкретного проекта. При этом конкретную модель жизненного цикла системы или ПС следует выбирать так, чтобы процессы и задачи были связаны между собой и определены их взаимосвязи с предшествующими процессами, видами деятельности и задачами.

Каскадная модель жизненного цикла наиболее известна и применяется достаточно широко. Она, по существу, реализует принцип однократного выполнения каждого из базовых процессов и этапов в их естественных границах. На рис. 1.1 представлен ***пример*** этапов каскадной модели ЖЦ ПС, которая в последующих лекциях используется ***как ориентир*** при изложении процессов программной инженерии. При этом в лекциях акцентируется внимание на методах обеспечения качества программных продуктов и не отражено программирование модулей и компонентов, которое остается за границами программной инженерии. Связь между этапами показана только сверху вниз, тогда как в реальных процессах жизненного цикла следует учитывать возможность возврата на предшествующие этапы, снизу вверх, для их уточнения и корректировки результатов.

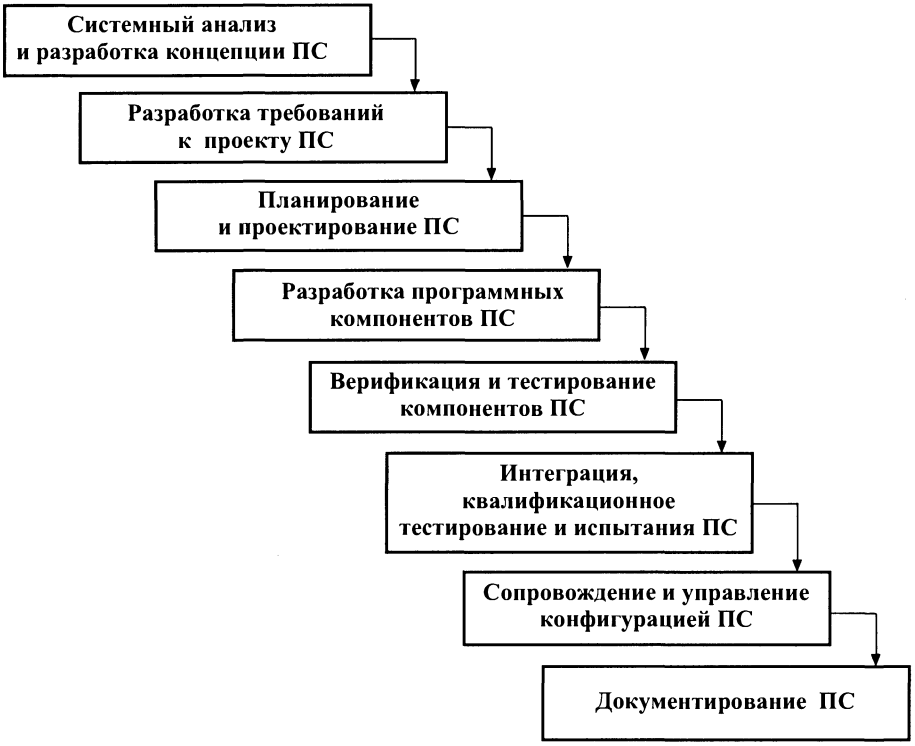

Рис. 1.1

При применении этой модели для создания каждого программного компонента соответствующие работы и задачи ***процесса жизненного цикла*** обычно выполняют последовательно. Однако они могут быть частично выполнены параллельно в случаях перекрытия последовательных работ. Когда несколько компонентов разрабатывают одновременно, для них работы и задачи процесса разработки могут быть выполнены параллельно. Процессы заказа и поставки, а также вспомогательные и организационные процессы выполняются параллельно с процессами разработки. Процессы сопровождения и эксплуатации обычно реализуются после процесса разработки.

Модель процессов жизненного цикла системы и степень ее практического применения в качестве ***обязательного или рекомендуемого доку-***

мента зависит от роли конкретного программного продукта в системе. Должна быть определена соответствующая модель жизненного цикла системы, в которой программный продукт становится ее частью. Установление этого поможет определить, можно ли использовать конкретную модель для разработки, эксплуатации или сопровождения программного средства. Программные средства могут быть постоянно (резидентно) размещены в компьютерах, встроены как часть программно-аппаратных средств или интегрированы в объект технических средств. В любом случае заказ, поставку, разработку, эксплуатацию или сопровождение программных средств необходимо координировать и гармонизировать с аналогичными процессами для всей исходной системы.

Для проекта системы должен быть проведен выбор одной или нескольких соответствующих моделей жизненного цикла. Необходимо установить, является ли модель жизненного цикла программного средства ***составной частью*** модели жизненного цикла системы либо полной моделью жизненного цикла ПС. Каждая модель жизненного цикла содержит некоторые процессы, которые могут быть выполнены ***последовательно, повторно или комбинированно***. Процессы должны быть отображены в выбранной модели жизненного цикла, с точки зрения создания модифицируемого, развивающегося, структурированного и планируемого продукта, результаты одного процесса из модели жизненного цикла должны быть переданы следующему. В этом случае соответствующие документы должны быть созданы к окончанию определенного процесса, до начала следующей работы.

Должны быть ***определены стороны (специалисты, предприятия)***, участвующие в проекте системы, и их ***ответственность за конкретные процессы и результаты в ЖЦ***. Следует учесть все работы и задачи, связанные с взаимодействиями (интерфейсами) между этими сторонами. Для большого проекта, в который вовлечено много лиц, необходимы развитой административный надзор и контроль, проведение внутренних и независимых оценок, анализов, аудиторских проверок, инспекций и подготовка отчетов, являющихся главным инструментарием для большого проекта.

Современные предприятия ***широко используют модели процессов жизненного цикла*** в качестве составной части деятельности по определению и усовершенствованию процессов, связанных с программными сред-

ствами. Применение стандартов жизненного цикла позволяет ориентироваться специалистам на построение систем и комплексов программ из крупных функциональных узлов, отвечающих требованиям стандартов, применять отработанные и проверенные проектные решения. Они определяют унифицированные интерфейсы взаимодействия компонентов таким образом, что разработчику системы, как правило, не требуется вдаваться в детали внутреннего устройства этих компонентов. Стандарты, относящиеся к программным комплексам (функциональным частям) систем, облегчают повторное использование в новых системах готовых и апробированных программных продуктов. Для унификации и регламентирования процессов ЖЦ ПС такие совокупности — ***профили стандартов*** должны адаптироваться и конкретизироваться применительно к определенным классам проектов, процессов и компонентов ПС. Таким образом, разработка программного продукта в значительной степени может сводиться к интеграции и комплексированию из стандартизированных компонентов.

Методы и процессы стандартизации жизненного цикла ПС играют ***стабилизирующую и организующую роль*** во всем жизненном цикле многих сложных систем. Они обеспечивают:

— расширение и совершенствование функций систем и компонентов с сохранением их целостности и первичных затрат;

— систематическое повышение качества функционирования комплексов программ и баз данных для решения задач пользователей в различной внешней среде;

— улучшение технико-экономических характеристик применения систем и программных продуктов;

— совершенствование технологий обеспечения жизненного цикла сложных систем и комплексов программ.

Для этого при создании и сопровождении сложных, распределенных систем, формировании их архитектуры, при выборе стандартов для программных компонентов и их связей целесообразно учитывать ***ряд современных концептуальных требований программной инженерии и формирования их жизненного цикла***:

— архитектура комплекса программ должна соответствовать текущим и перспективным целям и стратегическим, функциональным задачам создаваемой системы, быть достаточно гибкой и допускать относительно

простое, без коренных структурных изменений, развитие и наращивание функций и ресурсов системы в соответствии с расширением сфер и задач ее применения;

— в структуре и компонентах ПС и системы следует предусматривать обеспечение максимально возможной сохранности инвестиций в аппаратные и программные средства, а также в базы данных при длительном развитии, сопровождении и модернизации системы;

— необходимо обеспечивать эффективное использование ресурсов в ЖЦ системы и минимизировать интегральные затраты на обработку данных в типовых режимах ее функционирования с учетом эксплуатационных затрат и капитальных вложений в создание системы и программного продукта;

— должны быть обеспечены безопасность функционирования системы и надежная защита данных от ошибок, от разрушения или потери информации, а также авторизация пользователей, управление рабочей загрузкой, резервированием и оперативным восстановлением функционирования системы и программного продукта;

— для обеспечения перспективы развития жизненного цикла системы и комплекса программ целесообразно предусматривать возможность интеграции гетерогенных вычислительных компонентов и возможность переноса ПС и БД на различные аппаратные и операционные платформы на основе концепции и стандартов открытых систем;

— следует обеспечить комфортное обучение и максимально упрощенный доступ конечных пользователей к управлению и результатам функционирования системы и программного продукта на основе современных графических средств и наглядных пользовательских интерфейсов.

Наиболее актуальна ***стандартизация процессов жизненного цикла*** комплексов программ при коллективной разработке и сопровождении крупных ***критических систем управления в реальном времени***, к которым предъявляются высокие требования к качеству. В этих случаях особенно необходимо четкое планирование и управление технологическими процессами их жизненного цикла. Созданы или разрабатываются комплексы международных стандартов, в той или иной степени регламентирующие процессы проектирования, разработки, эксплуатации и сопровождения в ЖЦ программ и баз данных. Они обычно ориентированы на ПС, выполня-

ющие важные функции в системах управления объектами, технологическими процессами или при обработке ответственной информации. Применение таких стандартов полностью при создании и использовании простых программ, узкого или экспериментального назначения (первого класса — см. выше) не всегда может быть оправдано. Однако они определяют ***современную культуру программной инженерии и стандартизации жизненного цикла комплексов программ высокого качества***.

1.2. Роль системотехники в программной инженерии

Система — это совокупность взаимодействующих компонентов, работающих совместно для достижения определенных целей. Определяющим признаком системы является то, что свойства и поведение системных компонентов влияют друг на друга сложным образом. Корректное функционирование каждого системного компонента зависит от функционирования многих других компонентов. Системы часто имеют иерархическую структуру, т.е. в качестве компонентов содержат другие системы (подсистемы). Определяющее свойство подсистем заключается в том, что они могут функционировать самостоятельно, независимо от тех систем, в состав которых входят. Вместе с тем их поведение в составе конкретной системы зависит от взаимодействия с другими подсистемами.

Сложность взаимодействия между системными компонентами означает, что система не сводится просто к сумме ее составных частей. Она имеет определенные свойства, которые присущи ей именно как ***целостной системе***. Такие ***интеграционные свойства*** не могут быть свойствами отдельной части системы. Они проявляются тогда, когда система рассматривается как единое целое. Некоторые из этих свойств можно вывести из аналогичных свойств отдельных подсистем, но чаще они являются комплексным результатом взаимодействия подсистем и их невозможно оценить, исходя из анализа отдельных системных компонентов.

Системотехника — как технология создания систем охватывает все аспекты создания и модернизации сложных вычислительных комплексов, где программные продукты играют ведущую роль. Сюда можно отнести технологию разработки аппаратных средств, внутренних вычислительных процессов и связей всей системы, а также технологию создания ПС.

Инженеры-системотехники на основе спецификации требований системы определяют ее архитектуру и затем, собрав воедино ее отдельные части, создают законченную систему.

По мере увеличения в системах роли программных компонентов методы программной инженерии все шире используются в процессе создания разнообразных систем. Системотехника, как технология создания систем, охватывает процессы создания спецификаций, проектирования, разработки, тестирования, внедрения и сопровождения систем ***как единого целого***. Системотехник, занимающийся разработкой вычислительных систем, не может быть сосредоточен только на программном комплексе, он должен уделять равное внимание программному средству, аппаратным средствам и средствам взаимодействия с пользователями и системным окружением. Специалист по созданию программного продукта ***должен понимать задачи и методы системотехники***, поскольку возникающие проблемы часто являются результатом решений, принятых системотехниками.

Программный менеджер и/или системный инженер должен быть знаком с несколькими способами проектирования, знать, ***как перевести расплывчатые требования и пожелания заказчика в четкое техническое задание***, и уметь разговаривать с пользователем системы на языке предметной области, а не на профессиональном программистском жаргоне. Такие способности требуют, в свою очередь, гибкости и открытости, чтобы ухватить сущность предметной области различных приложений и стать в ней специалистом. Программный инженер должен обладать возможностью переходить от одного уровня абстракции к другому на разных стадиях проекта: от особых процедур и требований приложения к абстракциям программной системы, к специфике дизайна системы и, наконец, к уровню детального кодирования программ.

Неформальный подход, применяющийся к построению некоторых программ, недостаточен для разработки больших систем. Стоимость аппаратных средств постепенно снижается, тогда как ***стоимость программных продуктов стремительно возрастает***. Возникла необходимость в новых технологиях и методах управления комплексными проектами разработки больших программных систем. Такие методы составили ***программную инженерию***. Возрастает как объем производства программного продукта, так и его сложность. Кроме того, сближение вычислительной и

коммуникационной техники ставит новые требования перед специалистами. Это также является одной из причин возникновения проблем при разработке программных систем, как и то, что многие компании, занимающиеся производством ПС, не уделяют должного внимания эффективному применению современных методов и стандартов, разработанных в программной инженерии.

Программная инженерия — как часть системотехники охватывает все аспекты ***жизненного цикла ПС*** от начальной стадии разработки системных требований до завершения использования программного продукта. При этом специалисты выполняют практическую, инженерную работу. Они применяют теоретические построения, методы и средства там, где это необходимо, но делают это выборочно и всегда пытаются найти практическое решение задачи, даже если не существует подходящей теории или методов решения. Инженеры всегда должны понимать, что они работают в организационных и финансовых рамках заключенных контрактов и ищут решение поставленной перед ними задачи ***с учетом условий контракта***. Программная инженерия не рассматривает технические аспекты детального создания компонентов — в ее ведение входят такие задачи, как управление проектами ПС и разработка средств, методов и теорий, необходимых для обеспечения жизненного цикла комплексов программ. Программирование компонентов — это дело, главным образом, индивидуальное, а программная инженерия систем — всегда ***коллективная работа***.

Программные средства все больше встраиваются в различные системы. Работа с такими проектами требует от программного инженера широкого взгляда на общие ***задачи проектирования систем***. Программному инженеру необходимо участвовать в выработке требований для всей системы, а также пытаться понять прикладную область ПС еще до начала обдумывания абстрактных интерфейсов, требованиям которых должен будет отвечать программный продукт. Рассматривая программную инженерию ***как часть системотехники***, обнаруживается ***важность компромисса*** как отличительного признака любой инженерной дисциплины. Существуют принципиальные трудности изменения масштаба при попытке привнести приемы написания малых программ в проектирование больших программных комплексов.

Разработчики проекта системы вынуждены тратить время на общение друг с другом вместо того, чтобы писать программы. Иногда люди

покидают проект, и это влияет не только на работу, выполняемую непосредственно ими, но и на работу тех, кто от них зависит. Замена разработчика в проекте может требовать обучения и серьезнейшей подготовки нового специалиста для освоения им технических условий проекта и текущего состояния системы. Любое изменение первоначальных требований к системе влияет на многие составные части проекта, выливаясь в дальнейшем в задержку поставки готового продукта. Как в любой инженерной отрасли, программный инженер должен развивать умения, позволяющие построить набор моделей и оценить эти модели, управляя выбором компромиссов. Такие модели используются на этапе определения требований к проектируемой системе, в разработке архитектуры программного средства и на стадии реализации проекта. Программный инженер — это ***член команды***, поэтому должен обладать навыками общения и межличностных отношений, а также уметь планировать не только свою работу, но и координировать ее с работой других.

Специалист по программной инженерии должен знать системотехнику вычислительных систем, поскольку здесь программный компонент играет определяющую роль. Таким образом, технологии программной инженерии часто являются критическим фактором при разработке сложных вычислительных систем. Интеграционные свойства систем проявляются только тогда, когда система рассматривается как единое целое. В этом состоит сложность прогнозирования и оценки ее свойств, поскольку иногда можно измерить характеристики только подсистем, из которых состоит комплексная система. Высокие темпы роста основных ресурсов аппаратных средств (приблизительно на порядок каждые пять лет) и сохраняющаяся потребность в увеличении их использования со стороны различных пользователей и сфер применения приводят к необходимости ***адекватного совершенствования технологий создания программных средств и баз данных***.

1.3. Системные основы современных технологий программной инженерии

Основная цель современных технологий программной инженерии состоит в обеспечении эффективности всего жизненного цикла комплек-

сов программ для ЭВМ в различных проблемно-ориентированных областях. В понятие современной технологии включается совокупность методов и инструментальных средств автоматизации, а также технологические процессы, обеспечивающие жизненный цикл сложных ПС с заданными функциональными и конструктивными характеристиками качества. Для этого рекомендуется использовать наиболее эффективные и совершенные методы проектирования и проводить комплексную автоматизацию ЖЦ ПС (см. рис. 1.1). Целеустремленная деятельность разработчиков-поставщиков должна быть направлена на удовлетворение требований заказчиков и пользователей программных продуктов при их применении по прямому назначению.

Эта деятельность регламентируется рядом методов и стандартов, которые являются компонентами технологического обеспечения сложных ПС в течение их жизненного цикла. Их применение предполагает высокую дисциплину коллектива специалистов, использование им методик, стандартов, типовых нормативных документов и средств автоматизации разработки, которые регламентируют порядок организации и проведения работ по выполнению технологических операций, направленных на получение, в имеющихся организационно-технических условиях, готового ***программного продукта с заданными функциями и качеством***.

Методической основой технологии, регламентирующей деятельность специалистов, является типовой технологический процесс. Он отражается набором этапов и операций в последовательности их выполнения и взаимосвязи, обеспечивающих ведение работ на всех стадиях от инициирования проекта и подготовки технического задания до завершения испытаний или применения версии ПС. В современных технологиях объединяются методы непосредственной разработки программ и данных с методами обеспечения качества и организации управления их созданием с учетом технологических и человеческих факторов.

Индустриализация технологий программной инженерии ***базируется на стандартизации процессов*** разработки программ, их структурного построения и интерфейсов с операционной и внешней средой. Для этого с самого начала разработки должны определяться состав и этапы работ, необходимые для достижения конечной цели, а также требуемые для их выполнения ресурсы. Технические и управленческие проверки, анализ качества результатов промежуточных работ и компонентов, а также коррект-

ности их взаимосвязей должны обеспечивать руководителям и всем разработчикам уверенность достижения требуемого конечного результата проекта.

Достижение высоких значений качества комплексов программ существенно зависит от качества технологии и инструментальных средств, используемых разработчиками для обеспечения ЖЦ ПС. Уровень автоматизации, качество технологии и средств, применяемых для поддержки процессов жизненного цикла ПС, обычно сильно коррелированы с качеством создаваемых комплексов программ, а также с качеством средств автоматизации для их оценивания. Оценивание достоинств технологической базы ЖЦ позволяет прогнозировать возможное качество ПС и ориентировать заказчика и пользователей при выборе разработчика и поставщика для определенного проекта с требуемыми характеристиками. Поэтому определение уровня технологической поддержки процессов жизненного цикла, организационного и инструментального обеспечения ПС непосредственно ***связано с оцениванием реальных или возможных характеристик качества*** конкретного комплекса программ.

Значительные достижения в развитии и применении современных методов и технологии обеспечения ***крупномасштабных проектов ПС*** сосредоточены в методологии **CMM** (Capability Maturity Model — ***система и модель для оценки зрелости***) комплекса технологических процессов жизненного цикла ПС, а также в ее последующем развитии в **CMMI:2003** (см. лекцию 3). Она основана на формализации и использовании пяти уровней зрелости технологий поддержки ЖЦ ПС, которые также определяют потенциально возможное качество создаваемых на предприятии комплексов программ. Чем выше уровень зрелости, тем выше статус предприятия среди поставщиков, доверие к его продукции, его конкурентоспособность, а также возможное качество программных продуктов. Тем самым при выборе значений характеристик качества ПС можно в соответствующей степени доверять поставщику и предприятию разработчика, что они смогут полностью реализовать требования заказчика. Эти уровни зрелости характеризуются степенью формализации, адекватностью измерения и документирования процессов и продуктов в ЖЦ ПС, полнотой применения стандартов и инструментальных средств автоматизации работ, наличием и глубиной реализации функций, системой качества технологических процессов и их результатов.

Методология обеспечения качества ПС в программной инженерии поддержана рядом методических документов и инструментальных средств, а также формализована комплексом международных стандартов (см. Приложение 1). Внедрение комплекса требует больших усилий и затрат, что ограничило его массовое использование для относительно простых и средней сложности проектов. Концептуальные и организационные основы административного управления жизненным циклом и качеством ПС в системе **CMM**, а также **CMMI:2003**, определены в ***восьми базовых принципах***, которые декларированы в стандартах **ISO 9000:2000** и **ISO 15504:1-9.**

Принцип 1 — Ориентация предприятия-разработчика на потребителя-заказчика. «Предприятия зависят от своих потребителей и, таким образом, должны понимать текущие и будущие потребности потребителей-заказчиков, удовлетворять их требования и стремиться превзойти их ожидания».

Принцип 2 — Лидерство-руководство. «Лидеры обеспечивают единство назначения и направления деятельности предприятия. Они должны создавать и поддерживать внутреннюю окружающую среду, в которой специалисты могут в полной мере участвовать в достижении стратегических целей предприятия».

Принцип 3 — Вовлечение персонала. «Люди составляют сущность предприятия на всех уровнях, и их полноценное участие в деятельности способствует применению их способностей на благо целей предприятия».

Принцип 4 — Процессный подход. «Желаемый результат достигается более эффективно, когда требуемые ресурсы и деятельность специалистов предприятия управляются как единый связанный процесс».

Принцип 5 — Системный подход к административному управлению. «Выявление и понимание задач и административное управление системой взаимосвязанных процессов для заданной стратегической цели повышает эффективность и результативность предприятия».

Принцип 6 — Постоянное усовершенствование. «Непрерывное усовершенствование процессов и повышение качества продукции должно быть постоянной стратегической целью предприятия и его специалистов».

Принцип 7 — Подход к принятию решений, основанный на фактах. «Эффективные решения должны базироваться на анализе только реальных данных и достоверной информации».

Принцип 8 — Взаимовыгодные отношения с поставщиками. «Предприятие-пользователь и его поставщики-разработчики взаимозависимы, и взаимовыгодные отношения между ними повышают способность обоих производить качественную продукцию».

В стандарте **ISO 15504** каждый из приведенных принципов прокомментирован комплексом действий, необходимых для их реализации в проектах. Выполнение этих принципов способствует повышению управленческой культуры, применению системы административного управления качеством во всех видах деятельности предприятий и, как следствие, обеспечению высокого качества и конкурентоспособности создаваемой продукции, проектов и систем. ***Эти принципы рекомендуется применять при***:

— формулировке политики и стратегии обеспечения всего ЖЦ ПС;

— выборе целей проекта, требований и характеристик качества ПС, непосредственно связанных с потребностями и ожиданиями заказчиков и потребителей;

— управлении операциями в процессе реализации проекта и для удовлетворения требований заказчика и потребителей;

— управлении людскими ресурсами предприятия для обеспечения ЖЦ ПС и его качества.

Описание процессов ЖЦ ПС в **CMM** сфокусировано на поэтапном определении реально достигаемых результатов и на оценивании качества их выполнения. Качество процессов зависит от технологической среды, в которой они выполняются. ***Зрелость процессов*** — это степень их управляемости, возможность поэтапной количественной оценки качества, контролируемость и эффективность результатов. Модель зрелости предприятия представляет собой методический нормативный документ, определяющий правила создания и функционирования системы управления жизненным циклом ПС, методы постепенного повышения культуры и качества производства. Рост зрелости обеспечивает потенциальную возможность возрастания эффективности и согласованности использования процессов создания, сопровождения и оценивания качества компонентов и ПС в целом. Реальное использование регламентированных процессов предполагает их документирование и поэтапный контроль характеристик качества ПС. На предприятиях, достигших высокого уровня зрелости, формализованные процессы ЖЦ ПС должны принимать статус стандарта, фик-

сироваться в организационных структурах и определять производственную тактику и стратегию корпоративной культуры производства и системы обеспечения качества программного продукта.

В современных автоматизированных технологиях программной инженерии, создания и совершенствования сложных ПС с позиции обеспечения их качества можно выделить ***методы и средства, позволяющие***:

— создавать программные модули и функциональные ***компоненты высокого, гарантированного качества;***

— ***предотвращать дефекты проектирования*** за счет систем обеспечения качества, эффективных технологий и инструментальных средств автоматизации всего жизненного цикла комплексов программ и баз данных;

— ***обнаруживать и устранять различные дефекты и ошибки*** проектирования, разработки и сопровождения программ путем верификации и систематического тестирования на всех этапах жизненного цикла ПС;

— ***удостоверять достигнутые значения качества функционирования*** программных продуктов в процессе их испытаний и сертификации перед передачей в регулярную эксплуатацию пользователям.

Комплексное, скоординированное применение этих методов и средств в процессе создания, развития и применения ПС позволяет исключать многие виды дефектов или значительно ослаблять их влияние. Тем самым уровень достигаемого качества программных продуктов становится предсказуемым и управляемым, ***непосредственно зависящим от ресурсов***, выделяемых на его достижение, а главное, от системы качества и эффективности технологии, используемых на всех этапах жизненного цикла ПС. Эти ресурсы требуются на технологические средства в ЖЦ ПС:

— на приобретение или создание технологии и инструментальных средств, применяемых для обеспечения требуемого качества всего жизненного цикла ПС;

— на эксплуатацию и непосредственное применение технологии в процессе обеспечения ЖЦ ПС;

— на создание технологии и инструментальных средств для испытаний и оценивания характеристик качества программного средства;

— на выполнение измерений достигнутых значений характеристик качества ПС.

Улучшение технико-экономических показателей создания ПС, а также предотвращение ошибок и дефектов обеспечивается примене-

нием современных технологий программной инженерии и систем автоматизированного проектирования. Они представляют собой высокопроизводительные, ресурсосберегающие технологии создания комплексов программ высокого качества и надежности, имеют целью сокращение общих затрат на проектирование, реализацию, сопровождение и совершенствование ПС. Для этого, прежде всего, необходимо применять методы и средства системного анализа и проектирования, обеспечивающие конкретизацию и максимально точное представление целей, назначения и функций с начала ЖЦ ПС и предотвращающие распространение возможных системных дефектов на последующие этапы разработки. Такие ***технологии программной инженерии*** позволяют исключать или значительно снижать уровень системных, алгоритмических и программных ошибок в программных продуктах, передаваемых на эксплуатацию. Кроме того, они эффективны при модификации и сопровождении ПС, а также при изменении конфигурации внешней среды.

Для обнаружения, устранения ошибок и дефектов все этапы разработки и сопровождения ПС должны быть поддержаны ***методами и средствами верификации, а также систематического, автоматизированного тестирования корректности реализованных решений***. На этапах разработки ПС целесообразно применять различные методы, эталоны и виды тестирования, каждый из которых ориентирован на обнаружение, локализацию или диагностику определенных типов дефектов. Непредсказуемость конкретных дефектов и ошибок в программах приводит к целесообразности последовательного, методичного анализа возможности проявления любого типа ошибок и их исключения на наиболее ранних этапах разработки при минимальных затратах. Для тестирования необходимы достаточно полные эталоны, такие как совокупность требований технического задания и поэтапная их декомпозиция в спецификациях. Существенная особенность тестирования сложных ПС состоит в потребности их проверки при ограниченной длительности испытаний. Для этого целесообразно тщательное ***планирование тестирования*** с учетом всех результатов, полученных на этапах жизненного цикла. При планировании основная задача состоит в достижении максимальной достоверности испытаний и определения качества ПС при ограничении допустимых затрат ресурсов.

При применении импортных компонентов в системном проектировании и обеспечении качества программных продуктов следует учи-

тывать, что, в принципе, в них возможны как злоумышленные, так и случайные, непредумышленные дефекты вычислительного процесса, программ и данных, отражающиеся на качестве их функционирования. Злоумышленные вирусы и/или «закладки» хотя и маловероятны в серийных, широко тиражируемых в мире программных продуктах, однако требуются особые методы и средства для целенаправленного их обнаружения и устранения. Зарубежным специалистам свойственно ошибаться, так же, как и отечественным, однако более высокое качество используемых технологий разработки и современная проектировочная культура позволяют значительно снижать уровень случайных дефектов в программных продуктах, поступающих на рынок. Однако ***в любых сложных импортных ПС всегда не гарантировано полное, абсолютное отсутствие случайных ошибок и дефектов***, которые могут быть важнейшими дестабилизирующими факторами проектов. Их применение в критических отечественных системах требует соответствующего дополнительного контроля качества и специальных работ по обеспечению надежности и безопасности при проектировании и эксплуатации.

Комплексирование готовых импортных ПС и компонентов при проектировании конкретной отечественной системы создает ***условия их функционирования, не всегда адекватные*** предусмотренным разработчиками и проверенным при испытаниях, хотя, может быть, и не выходящие за пределы требований эксплуатационной документации. Это способствует проявлению ранее скрытых дефектов и ошибок и вызывает необходимость их устранения. Для этого ответственные и квалифицированные поставщики зарубежных программных продуктов имеют службы сопровождения, регистрации и накопления претензий пользователей и быстрого реагирования для устранения реальных дефектов функционирования. Легальная закупка и использование лицензионно чистых программных продуктов, обеспеченных сопровождением фирмы-поставщика, позволяет в значительной степени снижать влияние на качество функционирования ПС дефектов, не предотвращенных в процессе их создания.

Состояние экономики и промышленности страны все больше зависит ***от качества*** сложных информационных систем и их важнейшей, интеллектуальной части — программных продуктов, применяемых для управления в экономике, социальной сфере, системах вооружения и других областях. В связи с этим ***стратегической задачей*** стало обеспечение

высокого качества отечественных программных продуктов при их массовой разработке и поставке для различных сфер применения в стране и на мировом рынке. Для ***конкурентоспособности в мире сложных программных продуктов*** и возможности их успешного экспорта они должны быть сертифицированы и соответствовать требованиям международных стандартов.

Для ***удостоверения качества, надежности и безопасности применения*** сложных, критических систем используемые в них программные продукты следует подвергать ***сертификации*** аттестованными, проблемно-ориентированными испытательными центрами (см. лекцию 18). Такие испытания необходимо проводить, когда программы управляют сложными процессами или обрабатывают столь важную информацию, что дефекты в них или недостаточное качество могут нанести значительный ущерб. Сертификационные испытания должны устанавливать соответствие комплексов программ документации и допускать их к эксплуатации в пределах изменения параметров внешней среды, исследованных при проведенных проверках. Эти виды испытаний характеризуются наибольшей строгостью и глубиной проверок и должны проводиться специалистами, независимыми от разработчиков и от заказчиков (пользователей).

Основой сертификации должны быть детальные и эффективные Программы и методики испытаний комплексов программ на соответствие требованиям заказчиков, специально разработанные тестовые задачи и генераторы для их формирования, а также высокая квалификация и авторитет испытателей. Применение на предприятиях-разработчиках программных продуктов, сертифицированных ***систем качества и профилей международных стандартов*** на базе требований **ISO 9001:2000** и/или **CMMI:2003** гарантирует высокое, устойчивое управление качеством процессов и продуктов их жизненного цикла, а также позволяет во многих случаях облегчать сертификацию конечного программного продукта. Поэтому заказчики сложных программных проектов должны ***выбирать подрядчиков-исполнителей, имеющих сертификаты, удостоверяющие применение ими систем гарантирования качества*** на основе адаптированных профилей международных стандартов.

Пробелы в обучении методам программной инженерии оставляют широкое поле для произвола специалистов при оценивании качества их труда, а также для появления многочисленных дефектов и ошибок в про-

ектах ПС. Возрастание сложности и ответственности современных задач, решаемых программами, а также возможного ущерба от недостаточного качества их результатов значительно повысило актуальность освоения методов полного, стандартизированного описания требований к характеристикам качества и способов измерения их реальных значений на различных этапах ЖЦ ПС. Резко возросла необходимость знаний специалистами понятий, определений и способов оценивания характеристик качества программных продуктов.

Многие отечественные специалисты в области программных средств привыкли видеть в стандартах рутину, сковывающую их «творчество». Быстрое усложнение и рост размеров комплексов программ приводит к созданию ***крупных программистских коллективов*** с профессиональным разделением труда, в которых необходимо регламентирование координированной деятельности групп специалистов над единым проектом. Обещания разработчиков в контрактах с заказчиками создать высококачественные программы в согласованные сроки во многих случаях не выполняются как вследствие различий в понимании ими требуемого качества, так и вследствие неумения оценить ресурсы, необходимых для достижения высокого качества программ. В результате качество программной продукции зачастую остается низким, не поддающимся достоверной оценке и неконкурентоспособным на международном рынке. Поэтому важнейшей проблемой развития и применения современных систем является ***обучение и воспитание специалистов в области программной инженерии***, использованию международных стандартов, способствующих высокому качеству ПС и достоверному его оцениванию. Необходимо их обучение умению формализовать требования и достигать конкретных значений характеристик качества функционирования и применения сложных комплексов программ с учетом тех ресурсов, которые нужны и доступны для обеспечения и совершенствования этого качества.

ЛЕКЦИЯ 2

ПРОФИЛИ СТАНДАРТОВ ЖИЗНЕННОГО ЦИКЛА СИСТЕМ И ПРОГРАММНЫХ СРЕДСТВ В ПРОГРАММНОЙ ИНЖЕНЕРИИ

2.1. Назначение профилей стандартов жизненного цикла в программной инженерии

При создании и сопровождении сложных, распределенных, тиражируемых ПС требуется гибкое формирование и применение гармонизированных совокупностей базовых стандартов и нормативных документов разного уровня, выделение в них требований и рекомендаций, необходимых для эффективной реализации конкретных функций систем. Для унификации и регламентирования реализации этих функций совокупности базовых стандартов должны адаптироваться и конкретизироваться в программной инженерии применительно к определенным классам проектов, их функций, процессов и компонентов. В связи с этим выделилось и сформировалось понятие «профиля стандартов», как основного инструмента функциональной стандартизации.

Профиль стандартов — это совокупность нескольких (или подмножество одного) базовых стандартов (и других нормативных документов) с четко определенными и гармонизированными подмножествами обязательных и факультативных возможностей, предназначенная для реализации заданной функции или группы функций. Функциональная характеристика (заданный набор функций) объекта стандартизации является исходной для формирования и применения профиля этого объекта или процесса. В профиле выделяются и устанавливаются допустимые факультативные воз-

можности и значения параметров каждого базового стандарта и/или нормативного документа, входящего в профиль. Профиль не может противоречить использованным в нем базовым стандартам и нормативным документам. Он должен использовать факультативные возможности и значения параметров в пределах допустимых, выбранные из альтернативных вариантов. На базе одной и той же совокупности базовых стандартов могут формироваться и утверждаться различные профили для разных проектов и сфер применения. Эти ограничения базовых документов профиля и их гармонизация, проведенная разработчиками профиля, должны обеспечивать качество, совместимость и корректное взаимодействие компонентов системы, соответствующих профилю, в заданной области его применения.

Основными целями применения профилей стандартов при создании и применении ПС являются:

— снижение трудоемкости, длительности, стоимости и улучшение других технико-экономических показателей проектов систем и комплексов программ;

— повышение качества разрабатываемых или применяемых покупных компонентов и ПС в целом при их разработке, приобретении, эксплуатации и сопровождении;

— обеспечение расширяемости ПС по набору прикладных функций и масштабируемости в зависимости от размерности решаемых задач;

— поддержка функциональной интеграции в системах задач, ранее решавшихся раздельно;

— обеспечение переносимости программ и данных между разными аппаратно-программными платформами.

Состояние и развитие стандартизации в области программной инженерии характеризуется следующими особенностями, которые необходимо учитывать ***при формировании и применении профилей***:

— несколько сотен разработанных международных и национальных стандартов не полностью и неравномерно покрывают потребности в стандартизации объектов и процессов создания и применения сложных систем, программных средств и их компонентов;

— большая длительность разработки, согласования и утверждения международных и национальных стандартов (3—5 лет) приводит к их консерватизму, а также к хроническому отставанию требований и реко-

мендаций этих документов от современного состояния техники и от текущих потребностей практики и технологии создания сложных систем;

— стандарты современных ПС должны: учитывать необходимость их построения как открытых систем; обеспечивать расширяемость при наращивании или изменении выполняемых функций; переносимость программных средств и данных систем между разными аппаратно-программными платформами; возможность взаимодействия с другими информационными системами той же проблемно-ориентированной сферы;

— наиболее сложные и творческие процессы создания и развития крупных распределенных ПС (системный анализ и проектирование, интеграция компонентов и систем, испытания и сертификация) почти не поддержаны требованиями и рекомендациями стандартов вследствие трудности их формализации, унификации и разнообразия содержания;

— чем сложнее объекты или процессы, подлежащие стандартизации, тем больше необходимо использовать и формулировать предварительные условия, учитываемые в требованиях и рекомендациях стандарта, которые следует адаптировать и конкретизировать для корректного их применения в определенном проекте;

— пробелы и задержки в подготовке и издании стандартов высокого ранга и текущая потребность унификации и регламентирования современных объектов и процессов в области программной инженерии приводят к созданию и практическому применению многочисленных нормативных и методических документов отраслевого, ведомственного или фирменного уровня.

При практическом формировании и применении профилей ПС в ряде случаев возможно использовать национальные стандарты, стандарты дефакто и ведомственные нормативные документы. Это может быть обусловлено отставанием в разработке некоторых задач в международных стандартах или необходимостью учета конкретных особенностей систем. При применении стандартов и профилей могут быть выявлены пробелы в положениях некоторых стандартов и необходимость модификации или дополнения требований, определенных в них. Некоторые функции, не формализованные стандартами, но важные для унификации построения или взаимодействия компонентов, могут определяться нормативными документами ведомства или предприятия, обязательными для конкретного профиля и проекта.

Применение стандартизированных профилей позволяет заказчику системы освободиться от зависимости от одного поставщика программных или аппаратных средств за счет выбора этих средств из числа доступных на рынке и соответствующих стандартам, нормативным требованиям и рекомендациям профиля. Применение профилей, относящихся к программным комплексам (функциональным частям систем), облегчает повторное использование в проектируемой системе уже разработанных и проверенных программных компонентов. Профили ПС унифицируют и регламентируют только часть требований и характеристик объектов и процессов, выделенных и формализованных на базе стандартов и нормативных документов. Другая часть функциональных и технических характеристик систем определяется заказчиками и разработчиками творчески, без учета положений нормативных документов.

Профиль стандартов ЖЦ ПС (функциональных частей системы) должен ***определять архитектуру программного комплекса*** (модели функций, логические модели данных, внешние интерфейсы) и их структуру (разбиение системы на подсистемы и систем на модули, определение унифицированных интерфейсов взаимодействия между комплексами программ и их компонентами). Жизненный цикл программных средств отражается в профиле стандартов набором процессов, этапов, частных работ и операций в последовательности их выполнения и взаимосвязи, регламентирующим ведение разработки, сопровождение и эксплуатацию, от анализа и подготовки требований до завершения испытаний ряда версий программного продукта и прекращения их использования. Жизненный цикл включает описания исходной информации, способов и методов выполнения операций и работ, устанавливает требования к результатам и правилам их контроля, а также определяет содержание технологических и эксплуатационных документов. Он определяет организационную структуру коллектива специалистов, регламентирует распределение и планирование работ, а также контроль за ходом разработки. Повышение эффективности разработки, качества программного продукта и производительности труда специалистов ***достигается за счет***:

— регламентации организации и порядка проведения работ;

— автоматизации этапов и операций;

— рационального разделения труда между специалистами разной квалификации и проблемной ориентации.

Профиль ЖЦ ПС конкретной системы должен учитывать ее функциональную ориентацию. Он должен содержать ссылки на стандартизированные интерфейсы между комплексом программ и внешней средой, которые описываются в профилях среды системы. Каждый профиль и его параметры для применения в конкретном проекте системы необходимо поэтапно адаптировать и детализировать в соответствии с этапом проекта. Особенности организационных структур, различия в размерах и сложности проектов, в требованиях к системам и применяемым методам их разработки, необходимость преемственности с системами, находящимися в эксплуатации, влияют на организацию разработки, приобретения, применения и сопровождения программных средств. Каждый из выделенных профилей должен для последующего длительного использования пройти стадию формирования, адаптации и параметризации применительно к характеристикам стандартизируемых объектов или процессов.

Для корректного применения ***описания профилей стандартов должны содержать***:

— определение целей, которые предполагается достичь применением данного профиля стандартов;

— перечисление функций продукта или процесса стандартизации, определяемого данным профилем;

— формализованные сценарии применения базовых стандартов и спецификаций, включенных в данный профиль;

— сводку требований к системе или к ее компонентам, определяющих их соответствие профилю и требований к методам тестирования соответствия;

— ссылки на конкретный набор стандартов и других нормативных документов, составляющих профиль, с точным указанием используемых положений, редакций и ограничений, способных оказать влияние на достижение корректного взаимодействия объектов стандартизации при использовании данного профиля;

— информационные ссылки на спецификации тестов проверки соответствия профилю.

В зависимости от области распространения профилей они могут иметь разные ***статусы утверждения***:

— профили конкретной системы, определяющие стандартизированные проектные решения в пределах данного проекта и являющиеся частью проектной документации;

— профили, предназначенные для решения некоторого класса прикладных задач, которые распространяются на все системы и ПС данного класса в пределах предприятия или отрасли и утверждаются как стандарты предприятий, ведомственные или государственные стандарты.

Особенности организационных структур, различия в размерах и сложности проектов, требованиях к системам и применяемым методам их разработки, необходимость преемственности с системами, находящимися в эксплуатации, влияют на организацию разработки, приобретения, применения и сопровождения аппаратных и программных средств. Для ***эффективного применения конкретного профиля необходимо***:

— выделить объединенные единой логической связью проблемно-ориентированные области функционирования систем, где могут использоваться стандарты, общие для одной организации или группы предприятий;

— идентифицировать стандарты и нормативные документы, варианты их применения и параметры, которые необходимо включить в профиль стандартов;

— документально зафиксировать участки конкретного профиля, где требуется создание новых стандартов или нормативных документов, и идентифицировать характеристики, которые могут оказаться важными для разработки недостающих стандартов и нормативных документов этого профиля;

— формализовать профиль в соответствии с его категорией, включая стандарты, различные варианты нормативных документов и дополнительные параметры, которые непосредственно связаны с профилем;

— опубликовать профиль и/или продвигать его по формальным инстанциям для дальнейшего распространения на предприятии или в отрасли.

2.2. Жизненный цикл профилей стандартов систем и программных средств

Профиль стандартов конкретной системы не является статичным, он развивается и конкретизируется (возможно, во взаимодействии с заказчиком) в процессе жизненного цикла и оформляется в составе документации системы. Разработка и применение профилей стандартов являются ***органической частью процессов жизненного цикла, разработки и***

развития систем. Проектированию системы предшествует обследование объекта автоматизации, результатом которой являются его функциональная и информационная модели, определение целей создания системы и состава ее функций. Стандарты, важные с точки зрения заказчика, должны задаваться в спецификации требований на проектирование системы и составлять ее первичный профиль. То, что не задано в требованиях заказчика, остается первоначально на усмотрение разработчика системы, который, руководствуясь требованиями спецификаций, может дополнять и развивать профили, которые согласуются с заказчиком. В профиль конкретной системы включаются спецификации стандартизации компонентов, разработанных в составе данного проекта, и спецификации использованных готовых программных и аппаратных средств, если эти средства не специфицированы соответствующими стандартами. После завершения проектирования и испытаний системы, в ходе которых проверяется ее соответствие профилю, профиль применяется как основной ***инструмент сопровождения*** системы при эксплуатации, модернизации и управлении конфигурацией.

Целесообразно рассматривать ***две группы профилей систем*** (рис. 2.1):

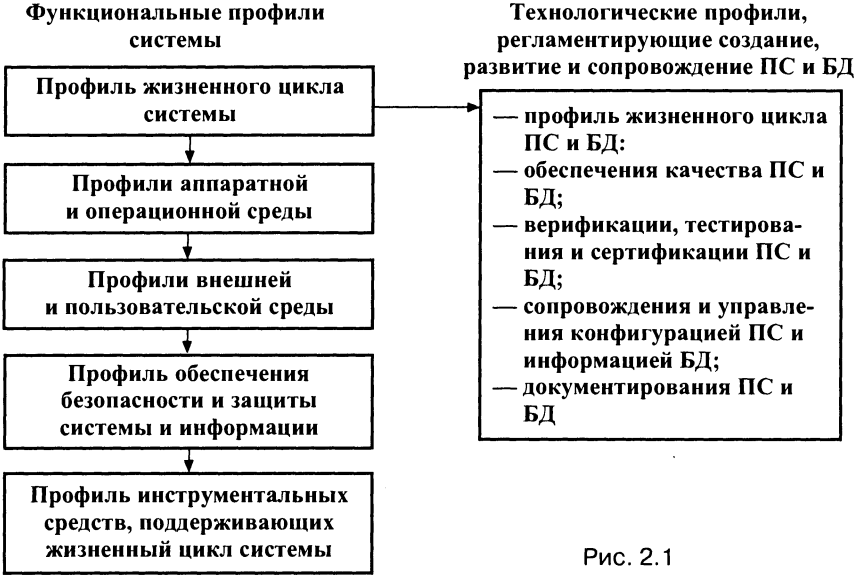

Рис. 2.1

— ***функциональные профили***, регламентирующие архитектуру и структуру объектов системы и ее компонентов; функции, интерфейсы и протоколы взаимодействия, форматы данных;

— ***технологические профили***, регламентирующие процессы проектирования, разработки, применения, сопровождения и развития систем и их компонентов.

На этапах жизненного цикла системы выбираются и затем применяются общесистемные ***функциональные профили***:

— профиль жизненного цикла информационной системы;

— профиль аппаратной и операционной среды системы;

— профиль внешней и пользовательской среды функционирования ПС;

— профиль обеспечения безопасности функционирования и защиты информации в системе;

— профиль инструментальных средств, поддерживающих весь жизненный цикл системы.

При применении функциональных профилей системы следует иметь в виду согласование (гармонизацию) этих профилей между собой. Необходимость такого согласования возникает, в частности, при применении стандартизированных интерфейсов, в том числе интерфейсов ПС и БД со средой их функционирования, интерфейсов со средствами защиты информации. При согласовании функциональных профилей возможны также уточнения профиля внешней среды системы и профиля инструментальных средств создания, сопровождения и развития программных средств.

Детализация ***общесистемных профилей стандартов*** производится по мере декомпозиции структуры системы на составляющие ее компоненты. Выбор и применение этих профилей является органической частью процессов проектирования, разработки, сопровождения и развития сложных систем. Их применение ***включает процессы***:

— выбор аппаратной и операционной среды системы определенного класса;

— определение внешней и пользовательской среды функционирования и применения системы;

— подготовку административного управления системой качества;

— выбор готовых программных и аппаратных средств, соответствующих функциям и профилям системы;

— проектирование и разработка программных средств и баз данных (функциональных частей системы) в соответствии с выбранными профилями, в частности в соответствии со стандартами на интерфейсы;

— разработка требований к методам тестирования компонентов системы на соответствие функциональным профилям, выбор или разработка тестов соответствия;

— тестирование компонентов системы на соответствие профилям или проверка сертификатов соответствия для применяемых готовых программных и аппаратных средств;

— комплексирование компонентов в создаваемой системе на основе последовательного применения профилей и их квалификационного тестирования.

Применение функциональных профилей должны поддерживать основные, ***технологические профили*** (см. рис. 2.1):

— жизненного цикла программных средств и баз данных;

— обеспечения качества программных средств и информации баз данных;

— верификации, тестирования и сертификации ПС и БД;

— сопровождения и управления конфигурацией ПС и информацией БД;

— документирования программных средств и информации баз данных.

Быстро оснащающиеся различными методами и средствами автоматизации этапы системного анализа, моделирования и предварительного проектирования не позволяют стабилизировать основу этих процессов, достаточную для их полной формализации для любых систем на уровне международных стандартов. Поэтому для этих этапов могут создаваться и применяться профили ЖЦ ПС как проблемно-ориентированные совокупности нормативных документов и методических руководств, отражающие как наиболее современные методы, так и фрагменты действующих стандартов, в том числе стандартов «де-факто».

Отдельные внутренние этапы жизненного цикла компонентов и комплексов программ обеспечиваются группами ***стандартов на локальные процессы***, определяющие:

— языки и процессы программирования программных компонентов;

— визуализацию информации для пользователей и обеспечения управления жизненным циклом ПС;

— защиту информационных ресурсов от несанкционированных вмешательств и криптографии;

— телекоммуникацию и взаимодействие с внешней средой.

Эта группа стандартов непосредственно определяет инструментальные средства решения соответствующих задач, и в процессах жизненного цикла ПС обычно стабильны, не изменяются и ***не раскрываются ниже в профилях ЖЦ***.

Учитывая динамику формирования и применения профилей жизненного цикла ПС, по мере детализации структуры системы и ее возможного развития образуется ***жизненный цикл профилей стандартов***. Жизненный цикл профилей ПС целесообразно рассматривать в составе технологических работ проекта отдельно от этапов и работ непосредственной разработки и эксплуатации самих программных средств и баз данных. Создание и применение профилей жизненного цикла ПС можно разделить на ***два крупных процесса*** (рис. 2.2):

— разработка, формирование и адаптация профилей стандартов ЖЦ ПС для использования в конкретном проекте системы;

— непосредственное применение требований и рекомендаций каждого адаптированного профиля стандартов для регламентирования этапов, работ и документов проекта ПС.

При создании ПС профили стандартов развиваются и детализируются параллельно с конкретизацией проекта. Они должны обеспечивать соответствующую часть ***технологической поддержки*** разработки комплекса программ нормативными документами. Таким образом, жизненный цикл профилей в некоторой степени ***подобен жизненному циклу*** самих программных средств и баз данных. Завершение разработки профилей стандартов системы и оформление результатов должно опережать, обеспечивать и подготавливать выполнение соответствующих этапов и работ основного жизненного цикла комплекса программ.

Процессы жизненного цикла, развития системы и ее программных компонентов должны быть поддержаны ***этапами развития и применения комплекта профилей***, которые включают:

— системный анализ объекта информатизации и создания концепции системы, когда производится первичный выбор исходного комплекта стандартов, которым должна соответствовать система; выявляется необходи-

мость разработки и состав дополнительных нормативных документов; оформляются содержание и параметры комплектов документов предполагаемых профилей;

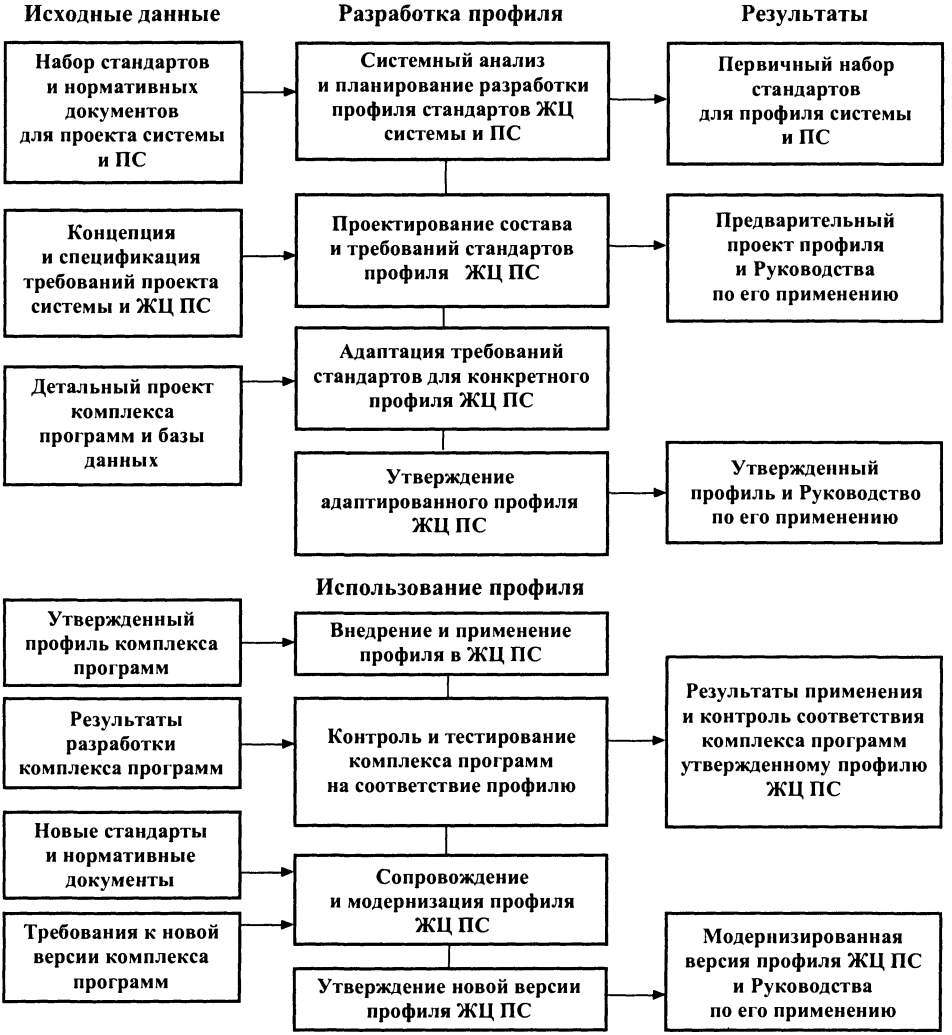

Рис. 2.2

— проектирование системы, когда определяются требования к ее архитектуре и структуре и соответственно уточняются положения, параметры и адаптируются стандарты комплекта профилей; оформляются проекты документов и методических руководств по применению рабочей версии каждого профиля стандартов;

— разработку или приобретение готовых компонентов системы, при этом утверждаются и применяются все положения профиля; производятся контроль, тестирование и испытания компонентов на соответствие требованиям и документам конкретного профиля стандартов;

— сопровождение, актуализацию и развитие системы, когда анализируются положения, параметры и результаты адаптации применяемой версии каждого профиля; выявляются и устраняются дефекты профилей;

— модернизацию профиля, с учетом появления более совершенных технических и программных средств, а также новых стандартов; при необходимости осуществляется формирование, документирование и внедрение новой модифицированной и уточненной версии соответствующего профиля.

В общем случае созданию профилей жизненного цикла системы должно предшествовать ***обследование объекта информатизации***, для которого предполагается создавать систему. Результатами работ на этом этапе являются функциональная и информационная модели, а также спецификации требований, которые служат в качестве исходных данных для проектирования системы и ПС. Целесообразно, чтобы эти модели и спецификации требований были выполнены с помощью формализованных методов их описания, например, с использованием средств описания моделей в известных методологиях структурного проектирования и языков спецификаций. В спецификации должны быть определены требования к жизненному циклу системы, даны ссылки на действующие нормативные документы и ***определена предварительная структура профиля стандартов жизненного цикла***. Следует задать требования к качеству системы и, соответственно, первичный профиль обеспечения качества комплекса программ и данных, функциональные требования к системе — состав решаемых задач и ссылки на нормативные документы, которые регламентируют правила и процедуры выполнения функций и операций.

На этапе ***системного анализа*** при планировании профиля технологической поддержки разработки ПС следует проанализировать набор ба-

зовых международных стандартов, связанных с регламентированием особенностей систем и программных средств. Для поддержки жизненного цикла разрабатываемых ПС необходимо из них выбрать предварительный набор стандартов, в наибольшей степени относящихся к ПС данного класса. Этот набор стандартов может быть дополнен возможными и целесообразными для применения стандартами де-факто и перечнем подлежащих разработке нормативных документов данного проекта. В результате формируется ***предварительный перечень*** стандартов и нормативных документов, который должен стать основой для профилей ЖЦ ПС. Этот перечень должен быть указан в спецификации требований или войти в состав системного проекта комплекса программ.

Одним из преимуществ от разработки и внедрения профиля стандартов для большой организации-пользователя является то, что он обеспечивает совершенствование взаимосвязей, особенно между разными подразделениями, которым необходимы гарантии того, что их системы будут корректно взаимодействовать, а ключевые программные средства и данные будут переносимы между платформами, полученными от разных поставщиков. На ***этапе определения области применения профиля*** должны быть выявлены:

— направления деятельности предприятия, подлежащие учету при построении профиля;

— срок реализации профиля и контрольная дата, когда работа над профилем должна быть завершена;

— технические стратегии, предположения и ограничения проекта системы и ПС;

— опытный и энергичный лидер, который пользуется в предприятии уважением и авторитетом, достаточным для того, чтобы возглавить и довести до конца работу по созданию и утверждению профиля стандартов проекта системы и ПС;

— уровень компетентности коллектива, разрабатывающего профиль, его знания и пригодность к экспертизе проекта и деятельности предприятия.

На ***этапе проектирования профиля ПС*** уточняются жизненный цикл и основные характеристики проекта. Это позволяет селектировать перечень стандартов и нормативных документов, целесообразных для использования в профилях ЖЦ данного ПС, ***провести их адаптацию*** для приме-

нения с учетом характеристик проекта, методологии и технологии создания ПС, а также предполагаемых средств автоматизации разработки, сопровождения и управления конфигурацией комплекса программ. На этом этапе описываются как функциональные, так и технические требования, устанавливаемые в профиле.

В уточненном плане реализации системы должны быть представлены ссылки на состав и содержание документов каждого профиля, выделены компоненты, параметры и ограничения, сформированные в процессе адаптации профиля ЖЦ данного ПС. Для разработчиков и заказчиков на этом этапе должен быть создан проект руководства применения профилей на последующих этапах ЖЦ. В результате на этом этапе формируется ***проект адаптированного набора профилей***. Необходимо провести предварительное обучение разработчиков проекта применению профилей ЖЦ ПС и основным концепциям профилей для данной системы. Конкретизация обеспечения технологической поддержки последующей разработки ПС позволяет ***завершить и утвердить адаптированные профили, поддерживающие ЖЦ ПС***, а также руководства по их применению. Результатом этого процесса является определение стандартов и выбор интерфейсов, которые удовлетворяют требованиям, предъявляемым к системе в целом.

Этап разработки системы и комплекса программ связан, прежде всего, с программированием и тестированием компонентов ПС, которые создаются заново для данной системы. Одновременно создаются функциональные тесты для проверки выполнения компонентами заданных функций. Разработка программных средств и их компонентов производится с помощью инструментальных средств, отвечающих требованиям выбранного ранее профиля методологии и технологии. Системные, аппаратные и программные средства необходимо проверять на соответствие функциональным и эксплуатационным требованиям профилей. Если закупленные продукты или платформы уже прошли у поставщика тестирование на соответствие профилям, процедура тестирования у потребителя может быть несколько сокращена при условии, что нет проблем с несоответствием архитектуры стандартам. Состав и содержание применяемых документов профилей ЖЦ ПС должны быть тесно связаны с планом и перечнем работ, выполняемых на соответствующих этапах. В обязательных документах

должно быть также отражено содержание дополнительных нормативных документов, согласуемых с заказчиком.

На ***этапе внедрения профиля стандартов*** важно иметь план по его применению. Руководители высшего уровня должны установить приоритеты при реализации отдельных частей и требований профиля. Внедрение профиля в соответствии с задачами проекта или предприятия будет упрощено, если ключевые цели обеспечения функциональной совместимости будут четко документированы в профиле. План внедрения профиля должен быть действующим документом и постоянно актуализироваться по мере изменения проекта.

Для обеспечения корректного применения каждого профиля должна быть разработана и утверждена ***методика проверки и тестирования*** для установления степени соответствия комплекса программ утвержденному профилю ЖЦ ПС и БД. Содержание и рекомендации профилей ЖЦ должны быть освоены специалистами, осуществляющими контроль их выполнения и тестирование создаваемого комплекса программ. Отдельные компоненты профиля подлежат тестированию как с точки зрения соответствия необходимым стандартам, так и соответствия требованиям, сформулированным в терминах их характеристик качества. Тестирование на соответствие не гарантирует функциональной совместимости, оно представляет лишь тест на соответствие набору тестовых утверждений, содержащихся в стандарте. Поведение объекта отслеживается и сравнивается с ожидаемым результатом эталонной реализации.

После детального проектирования версии ПС все последующие работы по созданию комплекса программ, вплоть до завершения испытаний и сертификации, ***должны проводиться в соответствии с утвержденными профилями ЖЦ ПС***, руководствами по их применению и проверяться на соответствие профилям по утвержденным методикам тестирования. Для этого должны быть созданы план, перечень и содержание работ, в которых применяются конкретные фрагменты, определенные положения каждого профиля и разделы методики, по которым тестируется соответствие версии ПС данному профилю. Наиболее полная проверка соответствия утвержденному профилю производится в процессе испытаний комплекса программ. В акте по результатам испытаний кроме всех характеристик версии программного продукта должно быть отражено соответствие про-

филям стандартов в той их части, которая непосредственно влияет на характеристики версии программного продукта. Кроме того, должны быть обобщены и представлены результаты применения утвержденных профилей ЖЦ ПС в процессе создания данной версии комплекса программ.

При сопровождении программного продукта и создании его новых версий накапливается опыт применения каждого использованного профиля стандартов ЖЦ, проявляются его некоторые недостатки и появляются предложения по модернизации. На этой стадии профиль продолжает выполнять регламентирующую функцию в качестве инструмента для управления конфигурацией системы. На этапе сопровождения профиль превращается в документ, позволяющий установить план текущих и долгосрочных мероприятий по развитию инфраструктуры предприятия и внедрению новых систем. Кроме того, в течение времени эксплуатации созданной версии программного продукта возможно появление новых стандартов де-юре и де-факто, которые целесообразно учесть в конкретном профиле. Сопровождение и смена версий ПС может привести к необходимости корректировки и модернизации конкретного профиля ЖЦ системы. Такая модернизация профиля может отразиться не только на вновь создаваемых версиях ПС, но потребовать доработок уже эксплуатируемых версий.

Жизненный цикл профиля стандартов ПС при его сопровождении может в некоторой степени повторять ЖЦ системы и/или ПС, созданных с его применением. Для этого следует разработать или выбрать и утвердить ***Руководство по сопровождению, развитию и модификации профиля ЖЦ ПС***, а также методики и план управления конфигурациями версий профиля, включающие:

— правила и процедуры идентификации компонентов и версий профиля стандартов;

— методики сбора, накопления и обработки сообщений о предлагаемых изменениях профиля;

— методики корректировки и извещения пользователей о выполненных изменениях в профиле, влияющих на характеристики качества программного продукта;

— методики и руководства по поддержке сохранности и адекватности документации и средств, реализующих требования и рекомендации профиля;

— руководство по вводу очередной версии профиля стандартов ЖЦ ПС.

При применении профилей следует обеспечить ***проверку корректности их использования путем тестирования, испытаний и сертификации,*** для чего должна быть создана технология контроля и тестирования в процессе применения профиля специалистами. Она должна быть поддержана совокупностью методик, инструментальных средств, составом и содержанием оформляемых документов на каждом этапе обеспечения и контроля корректности применения соответствующей версии и положений профиля. Профили должны определяться таким образом, чтобы тестирование их реализации можно было осуществлять по возможности наиболее полно, по стандартизированной методике. При сертификации сложных систем как специальный вид испытаний ***целесообразно выделять сертификацию на соответствие профилям***:

— ***процессов*** жизненного цикла системы и основных компонентов ПС и БД;

— ***продуктов*** и компонентов системы, подготовленных и рекомендуемых для эксплуатации и сопровождения.

В ряде случаев производится ***перенос разработанного программного продукта*** с инструментальной платформы разработчика системы на реальную — целевую платформу применения ПС. При этом проверяется соответствие реальной платформы требованиям функциональных профилей системы и функционирование ПС на реальной платформе. Этап внедрения предполагает адаптацию и настройку программного продукта на реальные условия эксплуатации, для которых он создавался. Приемочные испытания ПС должны проводиться в условиях реальной эксплуатации на соответствие спецификациям функциональных требований и требованиям полного профиля ПС, который был сформирован в процессе создания системы.

Последующая детализация требований и положений профилей должна проводиться с ориентацией на унификацию конкретных процессов, работ и документов версий программного продукта определенного функционального назначения. Можно выделить следующие ***основные группы специалистов, использующие документы профилей***:

— руководители — менеджеры крупного проекта системы и ее основных, функциональных компонентов программного продукта;

— менеджеры — системные аналитики, создатели спецификаций требований, пилотных проектов компонентов и алгоритмов решения функциональных задач;

— программисты-разработчики программных компонентов, структур и содержания данных;

— интеграторы функциональных программных компонентов, тестирующие и отлаживающие крупные функциональные компоненты или ПС в целом;

— специалисты сопровождения и управления конфигурацией версий программных продуктов;

— испытатели и сертификаторы программных продуктов;

— разработчики технологии, инструментальных средств, методических, руководящих и инструктивных документов, обеспечивающих реализацию профилей стандартов ЖЦ ПС.

Для деятельности перечисленных выше категорий специалистов на базе профилей должен быть создан ***комплект документов***, каждый из которых имеет конкретных пользователей в жизненном цикле ПС. В них должно быть отражено:

— содержание и описание выбранных положений и разделов стандартов и нормативных документов профиля с позиции его конкретного пользователя;

— параметры адаптации разделов стандартов профиля и содержание дополнительных нормативных документов;

— методика и сценарии корректного применения всех обязательных и рекомендуемых положений профиля стандартов;

— требования к содержанию отчетов о результатах контроля и тестирования компонентов системы на соответствие обязательным положениям профиля стандартов в процессе их жизненного цикла.

2.3. Модель профиля стандартов жизненного цикла сложных программных средств

Комплексное, скоординированное применение профилей стандартов и средств в процессе создания, развития и применения ПС позволяет исключать многие виды дефектов или значительно ослаблять их влияние.

Тем самым уровень достигаемого качества ПС ***становится предсказуемым и управляемым***, непосредственно зависящим от ресурсов, выделяемых на его достижение, а главное, от системы качества и эффективности технологии, используемых на всех этапах жизненного цикла ПС.

Процессы жизненного цикла ПС основаны на двух исходных принципах: ***модульности*** и ***ответственности***. Процессы являются модульными в том смысле, что они: строго связаны и взаимоувязаны; свободно соединены. Число интерфейсов между процессами сведено к минимуму. В принципе каждый процесс предназначен для реализации уникальной функции в жизненном цикле и может привлекать другой процесс для выполнения специализированной функции. Для обозначения, определения области применения и структурирования процессов используются правила:

— процесс должен быть модульным, т. е. один процесс должен выполнять одну и только одну функцию в жизненном цикле, а интерфейсы между двумя любыми процессами должны быть минимизированы;

— если функция вызвана более чем одним процессом, тогда функция сама становится процессом;

— должна быть возможность верификации любой функции в модели жизненного цикла ПС;

— каждый процесс должен иметь внутреннюю структуру, установленную в соответствии с тем, что должно им быть выполнено.

Когда организация в целом (или ее часть) заключает договор на программный продукт, то она становится ***стороной***. Организация имеет самостоятельные подразделения, а стороны могут быть из одной или разных организаций. Каждый процесс должен быть рассмотрен с точки зрения ***ответственности (обязанностей) стороны***. Организация может выполнять один или несколько процессов. Сторона, выполняющая процесс, несет ответственность за весь данный процесс, даже если выполнение отдельных задач поручено другим людям. Принцип ответственности в архитектуре и процессах жизненного цикла облегчает применение профилей стандартов для конкретного проекта, в который может быть вовлечено множество лиц.

Общая структура и состав профиля стандартов жизненного цикла системы и крупного программного средства для ***управления проектом*** представлены на рис. 2.3. На этом рисунке выделены и отражены группа

базовых стандартов управления проектами систем и основными процессами сложных программных средств, а также перечень совокупности стандартов, ***детализирующих и поддерживающих процессы их жизненного цикла***. Выделенные базовые стандарты образуют иерархическую группу и тесно связаны между собой концепциями, требованиями, процессам и ссылками. Основу профилей управления проектами составляют ***две группы: стандарты*** менеджмента качества процессов жизненного цикла систем —

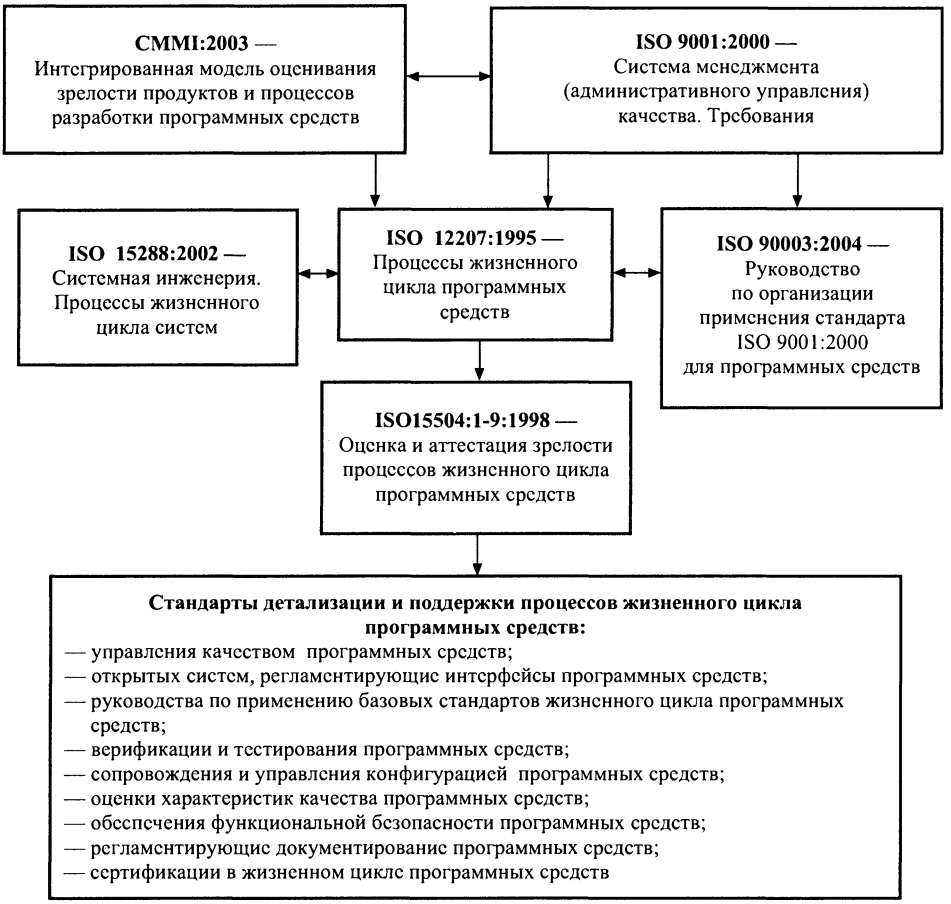

Рис. 2.3

CMMI:2003 и менеджмента (административного управления) системой качества (требования) — **ISO 9001:2000**. Так как эти стандарты имеют много общего и трудно выделить их преимущества, то при реальной разработке крупных проектов целесообразно уделять приоритет одной из групп в зависимости от особенностей конкретного проекта и предшествовавшего опыта специалистов предприятия.

Некоторым преимуществом применения стандарта **ISO 9001** для управления проектами ПС является его развитие и детализация требований в специальном руководстве **ISO 90003:2004** для программных средств. В этом руководстве цитируется каждое требование **ISO 9001**, оно комментируется и снабжается особенностями реализации процессов управления для конкретных проектов программных средств. Кроме того, при описании ряда процессов управления проектом для их уточнения и конкретизации делаются ссылки на основные стандарты, регламентирующие жизненный цикл ПС: **ISO 12207, ISO 15504, ISO 9126**, а в приложении проводится сопоставление требований этого стандарта с процессам управления и с рекомендациями стандарта **ISO 12207**.

Часто создание профилей стандартов крупных программных проектов начинается с определения жизненного цикла системы, процессы которого регламентируются стандартом **ISO 15288:2002**. Положения этого стандарта коррелированны с рекомендациями стандарта **ISO 12207**, которые детализируются в стандарте **ISO 15504:1-9:1998** и в последующей большой группе стандартов (см. рис. 2.3).

Профиль жизненного цикла ПС и БД целесообразно определять на основе подмножества процессов, работ и задач стандарта **ISO 12207**, выбирая их с учетом характеристик проекта конкретной системы. Возможно, что к выбранному подмножеству потребуется добавление дополнительных процессов, работ, задач и нормативных документов, связанных со спецификой данной системы. Это рекомендуется в новых **Приложениях 1** и **2** к этому стандарту, а также в ряде руководств, детализирующих основные процессы стандарта **ISO 12207**. Ряд работ, особенно на наиболее творческих этапах создания программного средства, не регламентируется стандартами. Это не позволяет разрабатывать и применять профили ЖЦ ПС, основанные только на базе стандартов. Иногда целесообразно дополнительно регламентировать такие работы нормативными документами и

спецификациями разработчиков проекта системы или ведомственными нормативными документами.

В стандарте **ISO 12207** и **Приложениях 1** и **2** к этому стандарту изложены ***основы преобразования и адаптации базовой структуры процессов ЖЦ для профиля конкретного проекта ПС и БД***. В них даны общие рекомендации по адаптации процессов ЖЦ, а также конкретные рекомендации по возможным изменениям ряда работ и результирующих документов в зависимости от характеристик конкретного объекта и процесса его разработки. В связи с возрастающей ролью качества сложных ПС целесообразно выделять профиль обеспечения качества ПС и БД конкретной системы, регламентирующий требования к качеству и меры по его обеспечению.

Модель профиля стандартов жизненного цикла сложных программных средств обычно формируется из ***10—12 базовых стандартов***. Их количество зависит от целей, сложности и особенностей проекта, от назначения и области применения модели, а также от возможностей формализации ее компонентов. Для последующего изложения программной инженерии при их выборе и формировании модели профиля стандартов учитывалось наличие международных стандартов (Приложение 1), которые могут использоваться при определении жизненного цикла ПС и объединении их в профиль, пригодный для последующего использования в технологии создания и развития крупного проекта. Поэтому ряд нестандартизированных — творческих процессов явно не отражен в рассматриваемой модели, однако они существенны для реального жизненного цикла ПС.

Сформированный и используемый далее в лекциях ***профиль жизненного цикла ПС*** состоит из ***трех групп стандартов*** — рис. 2.4:

— группы стандартов управления жизненным циклом сложных проектов систем и программных средств, возглавляемой стандартами менеджмента — **CMMI** и **ISO 9000**;

— группы стандартов проектирования, разработки, сопровождения и управления конфигурацией, регламентируемой базовыми стандартами жизненного цикла систем и программных средств — **ISO 15288** и **ISO 12207**;

— группы стандартов оценивания и обеспечения качества, безопасности и документирования в жизненном цикле программных средств, с головными стандартами — **ISO 9126** и **ISO 25000**.

Каждая выделенная группа ***профиля стандартов жизненного цикла*** (см. рис. 2.4) детализирована набором стандартов, которые представлены в Приложении 1. Эта схема далее рассматривается как базовая в программной инженерии, подлежащая конкретизации и адаптации в проектах в соответствии с особенностями развития профиля жизненного цикла программного средства (см. рис. 2.2). Большинство представленных стандартов изложено достаточно детально для практического применения в последующих лекциях.

Рис. 2.4

ЛЕКЦИЯ 3

МОДЕЛИ И ПРОЦЕССЫ УПРАВЛЕНИЯ ПРОЕКТАМИ ПРОГРАММНЫХ СРЕДСТВ

3.1. Управление проектами программных средств в системе — CMMI

Назначение методологии **CMM/CMMI** — ***системы и модели оценки зрелости*** — состоит в предоставлении необходимых общих рекомендаций и инструкций предприятиям, производящим ПС, по выбору стратегии совершенствования качества процессов и продуктов, путем анализа степени их производственной зрелости и оценивания факторов, в наибольшей степени влияющих на качество ЖЦ ПС, а также посредством выделения процессов, требующих модернизации. Для достижения устойчивых результатов ***программной инженерии*** в процессе развития технологии и организации управления жизненным циклом ПС в стандарте **ISO 15504:1-9** рекомендуется использовать эволюционный путь, который позволяет совершенствовать и постепенно повышать качество процессов и продуктов, вскрывать преимущества и недостатки предприятия. В методологии **CMM** ***выделены пять уровней зрелости***, раскрываемые в этом стандарте дефакто (рис. 3.1). Виды деятельности для высоких уровней зрелости в соответствии с **CMM** в стандарте делятся на базовые и общие. Базовые виды деятельности являются обязательными и сгруппированы в пять категорий: контрактная; инженерная; управленческая; вспомогательная; организационная. ***Уровни зрелости характеризуются*** степенью формализации, адекватностью измерения и документирования процессов и продуктов ЖЦ ПС, широтой применения стандартов и инструментальных средств автоматизации работ, наличием и полнотой реализации функций системой обеспечения качества технологических процессов и их результатов.

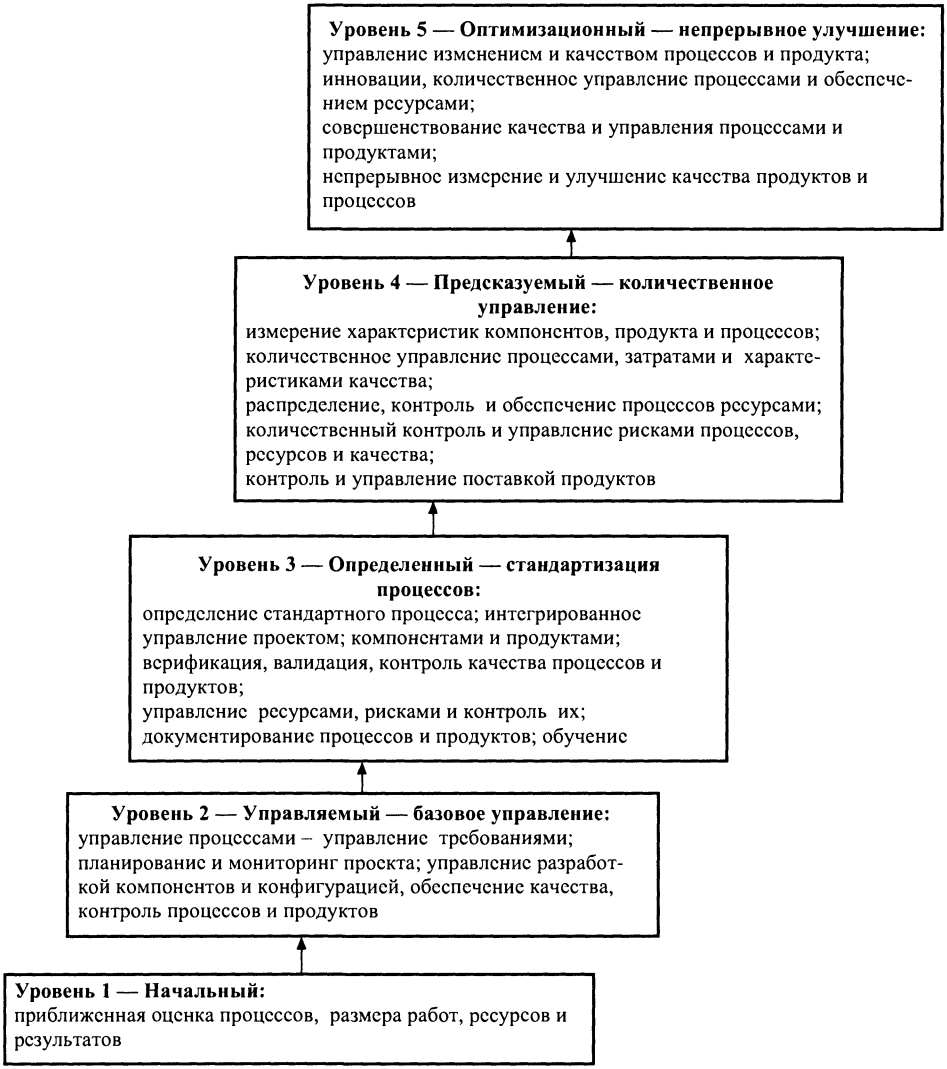

Рис. 3.1

Описание процессов ЖЦ ПС в СММ сфокусировано на поэтапном определении реально достигаемых результатов и на оценивании качества

их выполнения. Качество процессов зависит от технологической среды, в которой они выполняются. ***Зрелость процессов*** — это степень их управляемости, возможность поэтапной количественной оценки качества, контролируемости и эффективности результатов (см. рис. 3.1). Модель зрелости предприятия представляет собой методический нормативный материал, определяющий правила создания и функционирования системы управления жизненным циклом ПС, методы и стандарты систематического повышения культуры и качества производства. Рост зрелости обеспечивает потенциальную возможность возрастания эффективности и согласованности использования процессов создания, сопровождения и оценивания качества компонентов и ПС в целом. Реальное использование регламентированных процессов предполагает поэтапный контроль и документирование их характеристик качества. На предприятиях, достигших высокого уровня зрелости, формализованные процессы ЖЦ ПС должны принимать статус стандарта, фиксироваться в организационных структурах и определять производственную тактику и стратегию корпоративной культуры производства и системы обеспечения качества ПС.

Уровень 1 — Начальный. Массовые разработки проектов ПС характеризуются относительно небольшими размерами программ в несколько тысяч строк, создаваемых несколькими специалистами. Они применяют простейшие неформализованные технологии с использованием типовых инструментальных компонентов операционных систем. Основные процессы ЖЦ ПС на этом уровне не регламентированы, выполняются не совсем упорядоченно и зависят от некоординированных индивидуальных усилий и свойств отдельных специалистов. Успех проекта, как правило, зависит от энергичности, таланта и опыта нескольких руководителей и исполнителей. Процессы на первом уровне характеризуются своей непредсказуемостью по трудоемкости и срокам в связи с тем, что их состав, назначение и последовательность выполнения могут меняться случайным образом в зависимости от текущей ситуации.

Уровень 2 — Управляемый — базовое управление. Для сложных проектов ПС объемом в десятки и сотни тысяч строк, в которых участвуют десятки специалистов разной квалификации, необходимы организация, регламентирование технологии и унификация процессов деятельности каждого из них. Процессы на этом уровне заранее планируются, их выполне-

ние контролируется, чем достигается предсказуемость результатов и времени выполнения этапов, компонентов и проекта в целом. Основной особенностью второго уровня является наличие формализованных и документированных процессов управления проектами, которые пригодны для модернизации, а их результаты поддаются количественной оценке. На этом уровне акценты управления сосредоточиваются на предварительном упорядочении и регламентировании процессов создания, сопровождения и оценивания качества программного средства, однако для крупных проектов ПС с гарантированным качеством риск провала может оставаться еще достаточно большим.

Уровень 3 — Определенный — стандартизация процессов. При высоких требованиях заказчика и пользователей к конкретным характеристикам качества сложного ПС и к выполнению ограничений по использованию ресурсов необходимо дальнейшее совершенствование и повышение уровня зрелости процессов ЖЦ ПС. Процессы ЖЦ ПС на этом уровне должны быть стандартизированы и представлять собой целостную технологическую систему, обязательную для всех подразделений. На основе единой технологии поддержки и обеспечения качества ЖЦ ПС для каждого проекта могут разрабатываться дополнительные процессы последовательного оценивания качества продуктов с учетом их особенностей. Описание каждого процесса должно включать условия его выполнения, входные данные, рекомендации стандартов и процедуры выполнения, механизмы проверки качества результатов, выходные данные, условия и документы завершения процессов. В описания процессов включаются сведения об инструментальных средствах, необходимых для их выполнения, роль, ответственность и квалификация специалистов.

Уровень 4 — Предсказуемый — количественное управление. Для реализации проектов крупных, особенно сложных ПС, в жестко ограниченные сроки и с высоким гарантированным качеством, необходимы активные меры для предотвращения и выявления дефектов и ошибок на всех этапах ЖЦ ПС. Управление должно обеспечивать выполнение процессов в соответствии с текущими требованиями к характеристикам качества компонентов и ПС в целом. На этом уровне должна применяться система детального поэтапного оценивания характеристик качества как технологических процессов ЖЦ, так и самого создаваемого программного продукта

и его компонентов. Должны разрабатываться и применяться методики количественной оценки реализации процессов и их качества. Одновременно с повышением сложности и требований к качеству ПС следует совершенствовать управление проектами за счет сокращения текущих корректировок и исправлений дефектов при выполнении процессов. Результаты процессов становятся предсказуемыми по срокам и качеству в связи с тем, что они измеряются в ходе их выполнения и реализуются в рамках заданных ресурсных ограничений.

Уровень 5 — Оптимизационный — непрерывное совершенствование и улучшение. Дальнейшее последовательное совершенствование и модернизация технологических процессов ЖЦ ПС для повышения качества их выполнения и расширение глубины контроля за их реализацией. Одна из основных целей этого уровня — сокращение проявлений и потерь от случайных дефектов и ошибок путем выявления сильных и слабых сторон используемых процессов. При этом приоритетным является анализ рисков, дефектов и отклонений от заданных требований заказчика. Эти данные также используются для снижения себестоимости ЖЦ особо сложных ПС в результате внедрения новых технологий и инструментария, а также для планирования и осуществления модернизации всех видов процессов. Технологические нововведения, которые могут принести наибольшую выгоду, должны стандартизироваться и адаптироваться в комплексную технологию обеспечения и оценивания системы качества предприятия и его продукции.

В 2003 году американский ***Институт программной инженерии*** (**SEI**) опубликовал ***новую комплексную модель*** **CMMI**, уточняющую и совершенствующую предшествовавшие модели **CMM**, а также частично учитывающую основные ***требования существующих международных стандартов*** в области менеджмента программных средств. Внедрение этой модели акцентировано на улучшении процессов управления проектами ПС, обеспечении их высокого качества и конкурентоспособности, с основной целью — сделать процессы проектов ***управляемыми***, а результаты — ***предсказуемыми***. Значительное внимание в **CMMI** уделяется процессам разработки и учету итераций при изменении требований заказчиков, их прослеживанию к функциям, компонентам, тестам и документам проекта. Концепцию, определяющую функциональную пригодность и каче-

ство продукта, рекомендуется сопровождать проверками возможных сценариев событий при его применении.

В последнее время появилась информация о модернизации американским Институтом программной инженерии **SEI** версии **CMMI-SE/SW/IPPD, V1.1** на основе накопленного опыта и отзывов предприятий. Предполагается выпустить в 2006 году новую, существенно модернизированную, версию модели **CMMI-V1.2,** после чего постепенно должно прекратиться применение версии 1.1. До конца 2007 года должен проводиться переход пользователей на версию **CMMI-V1.2**, а в дальнейшем она станет обязательной для формализованной оценки качества (сертификации) технологии предприятий в области программной инженерии. При этом срок действия сертификата будет ограничен тремя годами. К этим изменениям следует готовиться после официальной публикации Институтом **SEI** версии 1.2.

Два варианта модели **CMMI — 1.1** созданы для обеспечения ***непрерывного*** оценивания комплекса процессов в определенной области создания программных средств или для ***поэтапного*** оценивания и совершенствования зрелости предприятия, а также для организации ЖЦ комплексов программ в целом. Модели **CMMI** представляют помощь специалистам при организации технологии и совершенствовании их продуктов, а также для упорядочения и обслуживания процессов разработки и сопровождения ПС. Концепция этих моделей покрывает управление и оценивание зрелости сложных систем, инженерии программных средств, а также процессов создания интегрированных программных продуктов и совершенствования их разработки. Компоненты непрерывной и поэтапной моделей в значительной степени подобны, могут выбираться и применяться в разном составе и последовательности использования в зависимости от свойств и характеристик конкретных проектов.

Варианты описания моделей построены по единой схеме, которая содержит ***общие разделы***:

предисловие;

1) введение;

2) модель компонентов;

3) терминология;

4) содержание уровней и главные компоненты каждого варианта модели (разработка целей и процедур);

5 раздел — структура взаимодействия процессов. Аннотированы четыре категории процессов раздела 7, их общий обзор и схемы взаимодействия **CMMI** процессов:

- менеджмент процессов;
- менеджмент — управление проектом;
- инжиниринг (технология);
- поддержка;

6 раздел — использование модели **CMMI** — краткие рекомендации для пользователей по применению модели и обучению; отмечается совместимость и соответствие процессов модели с регламентированными процессами предыдущей модели **CMM** в части 2 и 3 стандарта **ISO 15504**.

Последний, ***седьмой раздел*** самый большой в каждом стандарте, он занимает около 500 страниц из полного объема документа, который составляет свыше 700 страниц. В этом разделе представлены ***подробные рекомендации*** для реализации каждого из перечисленных в нем множества процессов, которые учитывают особенности конкретной модели.

Первый вариант (непрерывной) модели отражает документ: ***Capability Maturity Model Integration*** **(CMMI)** ***for Systems Engineering / Software Engineering / Integrated Product and Process Development, Version 1.1, Continuous Representation*** **(CMMI-SE/SW/IPPD,** ***V1.1, Continuous)*** — Интегрированная модель оценивания зрелости инженерии систем / программной инженерии / интегрированных продуктов и процессов разработки — ***непрерывное представление***. В этой модели ***седьмой раздел составляют процессы***:

— ***менеджмент процессов***:

- содержание организационных процессов;
- определение организационных процессов;
- организация обучения;
- организация преобразования (изменений) процессов;
- организация инноваций и расширений;

— ***управление проектом***:

- планирование проекта;
- мониторинг и контроль процессов проекта;
- управление соглашениями с поставщиками;
- интегрированное управление процессами и продуктами проекта;

• управление рисками;
• интеграция команды разработчиков;
• интегрированное управление поставщиками;
• количественное управление проектом;
— ***инженерия*** (технология):
• управление требованиями;
• разработка требований;
• технические решения;
• интеграция продукта;
• верификация;
• валидация (аттестация, утверждение);
— ***поддержка***:
• управление конфигурацией;
• обеспечение качества процессов и продуктов;
• измерение и анализ процессов и продуктов;
• анализ и принятие решений на изменения;
• организация окружения для интеграции;
• анализ причин и разрешение проблем (устранение дефектов).

В пяти приложениях приводятся:

A — состав использованных литературных источников и документов, в котором, однако, не упоминаются стандарты **ISO**;

B — сокращения;

C — глоссарий на основе терминологии **ISO**, применяемой только в четырех стандартах **ISO 9000, ISO 12207, ISO 15504:1-9, ISO 15288**;

D — описания требований и предложений для формирования компонентов модели по уровням зрелости;

E — список участников разработки **CMMI** — проекта.

В этой модели внимание акцентировано на организационных процессах, на планировании, управлении и контроле процессов реализации проектов программных средств, на разработке и управлении требованиями к программным продуктам. Ниже представлены примеры детализации в **CMMI** некоторых из них.

Планирование проекта в этой, так же как и во второй, модели включает:

— оценку возможного размера — масштаба программного продукта;

— оценку сложности функций и характеристик проекта ПС;

— определение модели и этапов жизненного цикла комплекса программ;

— технико-экономическое обоснование проекта — определение стоимости, трудоемкости и длительности ЖЦ ПС;

— разработка поэтапного графика работ и бюджета проекта;

— анализ, идентификация и оценка проектных рисков;

— планирование и управление документированием процессов и продуктов в ЖЦ проекта ПС;

— планирование и распределение технических и людских ресурсов по этапам ЖЦ ПС;

— планирование обеспечения знаний и квалификации коллектива специалистов для реализации проекта;

— обобщение и анализ совокупности планов проекта ПС;

— согласование работ и ресурсов по этапам ЖЦ разработчиком с заказчиком проекта ПС;

— документирование плана работ и утверждение его менеджером разработчиков проекта.

Процессы разработки требований к программному продукту аналогичны процессам в обеих моделях и включают:

— выявление реальных потребностей заказчика и пользователей к функциям и характеристикам программного продукта;

— разработку и согласование между заказчиком и разработчиком исходных, базовых требований к функциям программного продукта;

— определение доступных ресурсов и ограничений проекта комплекса программ;

— декомпозицию базовых исходных требований к функциям ПС в набор требований к компонентам и тестам комплекса программ;

— формализацию требований к интерфейсам между компонентами, с операционной и внешней средой;

— разработку концепции программного продукта и сценариев его использования;

— разработку требований к обобщенным характеристикам функциональной пригодности и использованию функций программного продукта по назначению.

Управление требованиями в обеих моделях включает:

— достижение однозначного понимания требований к проекту ПС заказчиком и разработчиками;

— получение заказчиком от разработчиков обязательств выполнить все его требования к программному продукту;

— согласованное между заказчиком и разработчиком управление изменениями требований к проекту ПС;

— обеспечение прослеживания корректности изменений от общих требований к проекту ПС до требований к компонентам и частным процессам;

— выявление и идентификация несоответствий между процессами разработки проекта и требованиями заказчика.

Второй вариант **CMMI** представлен документом: ***Capability Maturity Model Integration for Systems Engineering / Software Engineering / Integrated Product and Process Development, Version 1.1, Staged Representation*** **(CMMI-SE/SW/IPPD,** ***V1.1, Staged)*** — Интегрированная модель оценивания зрелости инженерии сложных систем / программной инженерии / интегрированных продуктов и процессов разработки — ***поэтапное представление***. Модель базируется на сохранении концепции пяти уровней зрелости **CMM**. Состав процессов практически повторяет приведенный выше для первого варианта модели, в несколько иной последовательности и с относительно небольшими дополнениями. Первый уровень отличается значительной неопределенностью состава и содержания процессов в различных относительно простых проектах, поэтому он в документе не описан и не комментируется. Поэтому при уточнении и детализации содержания процессов в поэтапном варианте **CMMI** рекомендуется ограничиваться ***четырьмя (2-й — 5-й) основными уровнями***:

— ***второй уровень*** — формализует базовое управление проектами:

- управление требованиями;
- планирование проекта;
- мониторинг и контроль проекта;
- управление соглашениями с поставщиками;
- измерение и анализ процессов и продуктов;
- обеспечение качества процессов и продуктов;
- управление конфигурацией;

— ***третий уровень*** — содержит стандартизацию основных процессов:

- разработка требований;
- технические решения;
- интеграция продукта;
- верификация;
- валидация (аттестация);
- содержание организационных процессов;
- определение организационных процессов;
- организация обучения;
- интегрированное управление процессами и продуктами проекта;
- управление рисками;
- интеграция команды разработчиков;
- интегрированное управление поставщиками;
- анализ и разрешение проблем (устранение дефектов);
- организация окружения для интеграции;

— ***четвертый уровень*** — определяет количественное управление:

- организация представления качества процессов;
- количественное управление всем проектом и ресурсами;

— ***пятый уровень*** — оптимизационный, непрерывное совершенствование:

- организация, инновации, количественное управление процессами и обеспечением ресурсами;
- анализ причин дефектов, совершенствование качества и управления процессами и продуктами.

Приложения во втором варианте модели подобны по составу приведенным выше приложениям для первой модели. Рекомендуется на каждом более высоком уровне зрелости применять ***все процессы*** предыдущих нижних уровней. В обоих вариантах модели каждый, выделенный выше базовый процесс комментируется подробными рекомендациями для его практической реализации, которые содержат унифицированные по структуре описания объемом около 20—30 страниц:

— общие цели процесса, которые должны быть достигнуты;

— вводные замечания и общее описание функций процесса;

— специфические цели процесса;

— менеджмент процесса;

— разработка требований к процессу;
— взаимодействие и интерфейсы с другими процессами;
— практические цели — требуемые результаты действий процесса;
— планирование действий в определенном процессе;
— анализ и валидация (утверждение) результатов реализации процесса;
— мониторинг и контроль выполнения процесса.

Эти рекомендации по объему, содержанию и полноте описаний базовых процессов подобны ряду стандартов профиля ЖЦ ПС. Упорядочение и оценка полноты используемых процессов в соответствии с уровнями зрелости позволяет устанавливать производственный потенциал предприятий — разработчиков программных продуктов по прогнозируемому качеству процессов и результатов их деятельности и готовности к сертификации на соответствие определенному уровню зрелости модели **CMMI** — 1.1.

Особое внимание в моделях **CMMI** уделяется процессам менеджмента проекта ПС. Эти ***требования и процессы моделей практически соответствуют*** регламентированным и детализированным рекомендациям в стандартах **ISO 9001:2000, ISO 12207** и в основных компонентах ***профиля стандартов жизненного цикла сложных ПС***. Требованиям к процессам в функциональных разделах 4—8 стандартов **ISO 9001, ISO 9004, ISO 90003** может быть сопоставлен адекватный по содержанию ряд разделов в ***моделях*** **CMMI** — рис. 3.2. Общность процессов и требований состоит в подобии: состава, терминологии, структуры, перечня основных рекомендуемых процессов управления, планирования, учета доступных ресурсов, реализации процессов программной инженерии, оценивания и организации специалистов.

С точки зрения поддержки и регламентирования полного жизненного цикла крупных проектов программных средств к ***недостаткам моделей*** **CMMI** относительно профиля существующих стандартов **ISO** можно отнести следующие:

— не все процессы предусмотрены в составе процессов моделей **CMMI** — 1.1, которые развиваются и детально комментируются для их реализации в стандартах **ISO 9004:2000** и **ISO 90003:2004**, а также в профиле стандартов **ISO**;

— не отражены особенности системной инженерии и международные стандарты, регламентирующие процессы жизненного цикла сложных систем **ISO 15288:2002** и **ISO 19760:2003**;

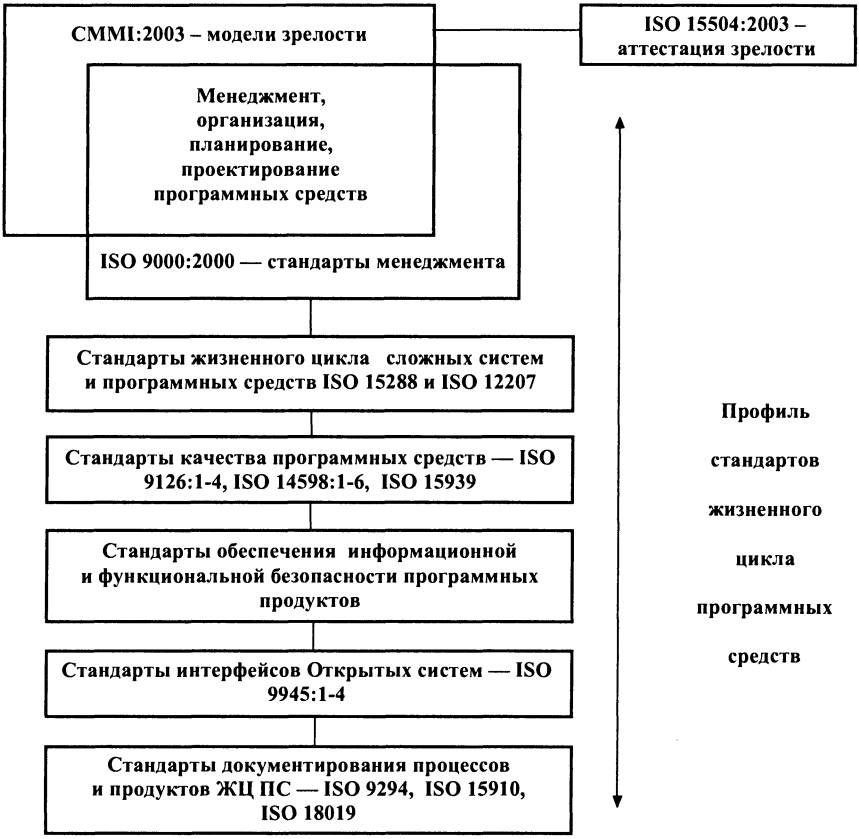

Рис. 3.2

— при анализе процессов обеспечения качества используется ряд традиционных характеристик систем и программных продуктов, которые применяются в сложных проектах, однако не описаны и не комментируются базовые международные стандарты, систематизирующие и регламентирующие качество программных средств — **ISO 9126:1-4, ISO 14598:1-6, ISO 15939**;

— отсутствуют описания характеристик и конкретных процессов обеспечения информационной и функциональной безопасности программных продуктов и ссылки на многочисленные стандарты в этой области;

— не отражены регламентированные интерфейсы Открытых систем на взаимодействие программных компонентов, а также с операционной и внешней средой, в соответствии со стандартами — **ISO 9945:1-4**;

— документирование процессов и продуктов ЖЦ ПС комментируется только по мере их реализации, и не представлены обобщенные требования к технологической и эксплуатационной документации на программный продукт в соответствии со стандартами — **ISO 9294, ISO 15910, ISO 18019**.

Для определения представленных выше уровней зрелости процессов обеспечения жизненного цикла ПС разработан и первоначально утвержден в 1998 году обширный технический отчет **ISO 15504** — Оценка и аттестация зрелости процессов создания и сопровождения ПС и систем, состоящий из девяти частей и множества приложений. В нем изложены модель зрелости **CMM** и восемь базовых принципов программной инженерии на основе стандарта **ISO 9000:2000** (см. лекцию 1). Затем в **ISO** этот документ претерпел коренную переработку, сокращение, упрощение структуры и содержания, при полном сохранении целей и концепции, и утвержден ***как стандарт*** в составе пяти частей (см. Приложение 1). Стандарт **ISO 15504:1-5:2003-2006** регламентирует оценку и аттестацию зрелости процессов создания, сопровождения и совершенствования программных средств и систем, выполняемых предприятиями:

— для установления состояния собственных технологических процессов и их совершенствования;

— для определения пригодности собственных процессов для выполнения определенных требований или классов требований заказчиков;

— с целью его пригодности для выполнения определенных договоров с заказчиками ПС и систем.

Стандарт способствует: самоаттестации зрелости предприятий, обеспечению адекватного управления аттестуемыми процессами, определению профиля рейтингов процессов и подходит к любым сферам применения и размерам ПС и систем. Применение стандарта направлено на выработку предприятиями и специалистами ***культуры постоянного совершенствования зрелости технологий*** обеспечения ЖЦ ПС, отвечающих бизнес-целям проектов и оптимизации использования доступных ресурсов. Аттестация зрелости процессов предприятий обеспечивает возможность их сопоставления и выбора, предпочтительных для определенных проектов:

— для заказчиков, покупателей, пользователей программных продуктов и систем — способность определять текущую и потенциальную зрелость процессов жизненного цикла у предприятия-поставщика;

— для поставщиков и разработчиков — способность определять текущую и потенциальную зрелость собственных процессов жизненного цикла ПС и систем, области и приоритеты усовершенствования процессов;

— для аттестаторов зрелости — основу для проведения и совершенствования процессов аттестации.

Аттестация в стандарте рассматривается в ***двух аспектах***: для усовершенствования процессов ЖЦ ПС и систем конкретного предприятия и для определения соответствия декларированной зрелости процессов обеспечения проекта или предприятия реальным используемым процессам. Это отражено в следующих пяти частях стандарта **ISO 15504:1-5:2003-2006**.

Часть 1 — Концепция и словарь — содержит общую информацию о процессах аттестации зрелости ПС и систем и рекомендации по использованию частей стандарта. Изложены общие требования к аттестации, терминология, структура и область применения остальных частей стандарта.

Часть 2 — Выполнение (производство) аттестации — включает детальные требования к проведению процессов аттестации, как основы для совершенствования и определения уровня зрелости технологических процессов обеспечения ЖЦ ПС и систем. Документ определяет процессы выполнения аттестации, модели рекомендуемых процессов аттестации и верификации процессов, с тем чтобы они были объективными, содержательными и репрезентативными.

Часть 3 — Руководство по производству аттестации — содержит обзор технологии выполнения процессов аттестации зрелости и интерпретации реализации требований. В нем отражены: исполнение аттестации; измерительные средства для определения процессов зрелости; выбор и применение средств аттестации; оценка компетентности аттестаторов; верификация соответствия аттестации декларированным требованиям. Средства аттестации могут использоваться предприятиями при планировании, менеджменте, мониторинге, контроле и усовершенствовании программных продуктов и систем, при их приобретении, разработке, применении и сопровождении.

Часть 4 — Руководство пользователей для процессов усовершенствования и определения зрелости процессов по этим двум аспектам. Рекомендуется ряд шагов, которые включают: применение результатов процессов аттестации; постановка целей аттестации зрелости; определение исходных данных для аттестации; оценка возможного снижения результирующих рисков; шаги по усовершенствованию процессов; шаги по определению уровня зрелости; сравнение результатов анализа аттестации с требованиями.

Часть 5 — Образец модели процессов аттестации на соответствие требованиям, представленным в части 2. Обширный документ (162 стр.) содержит примеры практического применения предыдущих частей стандарта для организации, оценивания и совершенствования аттестации зрелости процессов жизненного цикла для различных областей использования, проектов программных средств и предприятий.

При практической реализации проектов и обеспечении ***жизненного цикла сложных ПС*** разработчикам и поставщикам может быть трудно определить и выделить для применения преимущества моделей **CMMI**. В зависимости от традиций предприятия и особенностей крупного проекта ПС зачастую целесообразно использовать как основной полный ***профиль стандартов*** **ISO**, а для оценивания заказчиками ***уровня зрелости*** менеджмента, организационного и технологического обеспечения проектов ПС применять конкретные рекомендации **CMMI**. Эти рекомендации могут эффективно использоваться при ***сертификации качества процессов*** на предприятиях, обеспечивающих ЖЦ ПС как альтернатива или наряду с сертификацией по комплексу стандартов менеджмента **ISO 9000**, в зависимости от особенностей проекта и требований заявителя на сертификацию программного продукта или технологии обеспечения его жизненного цикла.

3.2. Стандарты менеджмента (административного управления) качеством систем

Серия стандартов **ISO 9000:2000** разработана, чтобы помочь предприятиям всех типов и размеров внедрить и использовать эффективные ***системы менеджмента (административного управления) качества***.

Совместно они образуют комплект согласованных стандартов управления системами качества и могут применяться как ***основа процессов управления в программной инженерии***:

— **ISO 9000:2000** — представляет введение в системы управления качеством продукции и услуг и словарь качества;

— **ISO 9001:2000** — устанавливает детальные требования для систем управления качеством, достаточные в случае необходимости продемонстрировать способность предприятия, обеспечить соответствие качества продукции и услуг требованиям заказчика;

— **ISO 9004:2000** — содержит руководство по внедрению и применению широко развитой системы управления качеством, чтобы достичь постоянного улучшения деловой деятельности и результатов предприятия.

Стандарты серии **ISO 9000:2000** применяют процессный подход в административном управлении системами качества предприятий, а также рассматривают способы быстрого выявления и реализации возможностей для их улучшения. Процессная модель подчеркивает тот факт, что заказчики и другие заинтересованные стороны играют значительную роль в процессе установления исходных требований. После этого по отношению ко всем процессам, необходимым для создания необходимой продукции, применяется управление процессами и проводится проверка «выходов». Измерение степени удовлетворенности заказчика и других заинтересованных сторон используется в качестве обратной связи для оценки и признания того, что требования заказчика были выполнены полностью.

В ***стандарте*** **ISO 9004** детализированы руководящие указания и рекомендации по применению системы менеджмента качества, которые изложены в том же порядке, как требования в **ISO 9001**. Оба стандарта ссылаются на **ISO 9000**, который объясняет используемую терминологию и определения. Структура основных требований и рекомендаций в этих стандартах сведена к четырем объединенным крупным процессам (рис. 3.3):

— ***обязанности и ответственность администрации управления качеством*** (раздел 5);

— ***административное управление ресурсами*** (раздел 6);

— ***процессы жизненного цикла продукции и управления ее качеством*** (раздел 7);

— ***измерения, анализ и совершенствование продукции*** (раздел 8).

Рис. 3.3

Стандарты серии **ISO 9000:2000** рекомендуется применять в деятельности предприятия, начиная от идентификации требований заказчика, и охватывать все процессы системы управления качеством, вплоть до достижения соответствия требованиям. Применение сокращенного, адаптированного варианта требований не освобождает предприятие от ответственности предоставлять продукцию, которая удовлетворяет всем требованиям заказчика. Система качества должна быть внедрена, поддерживаться в

рабочем состоянии и подвергаться улучшениям со стороны специалистов предприятия. Масштаб и глубина процедур должны определяться такими факторами, как размер и тип предприятия, сложность и взаимосвязь процессов, применяемые методы, а также квалификация и степень подготовки персонала, участвующего в выполнении работ. Они ***должны включать***:

— общесистемные процедуры, которые описывают деятельность, необходимую для внедрения и применения системы качества;

— процедуры, описывающие последовательность и внутреннее содержание процессов, необходимых для обеспечения уверенности в соответствии продукции установленным требованиям;

— инструкции, описывающие операционную деятельность и управление процессами.

Для освоения и облегчения применения стандартов в редакции 2000 года в приложении к **ISO 9001:2000** приведены таблицы, отражающие взаимное соответствие требований этого стандарта и требований **ISO 9001:1994**. Таблицы могут использоваться при необходимости подтверждения в настоящее время сертификатов качества, полученных ранее на основании применения стандартов в редакции 1994 года. Это позволяет сохранять и практически использовать всю технологическую документацию и рабочие инструкции, базирующиеся на стандартах качества 1994 года, дополняя их только этими таблицами соответствия требований.

Стандарт **ISO 9001:2000** — Система менеджмента (административного управления) качества. Требования — является базой для Руководства по их реализации в стандарте **ISO 9004:2000** и кратко ***изложены ниже*** (см. рис. 3.2).

Общие требования к системе менеджмента качества. Организация-разработчик должна установить и управлять процессами, необходимыми для обеспечения уверенности в том, что продукция и/или услуга соответствуют требованиям заказчика. В качестве способа внедрения и демонстрации установленных процессов организация должна создать систему менеджмента качества, основываясь на требованиях настоящего международного стандарта. Система менеджмента качества должна быть внедрена, поддерживаться в рабочем состоянии и подвергаться улучшениям со стороны организации. Организация должна подготовить процедуры системы менеджмента качества, которые описывают процессы, необходимые для внедрения системы менеджмента качества. Масштаб и глубина

процедур должны определяться такими факторами, как размер и тип организации, сложность и взаимосвязь процессов, применяемые методы, а также квалификация и степень подготовки персонала, участвующего в выполнении работ.

Высшее руководство предприятия-разработчика должно продемонстрировать свои обязательства заказчику относительно:

— создания и поддержания важности удовлетворения требований заказчика;

— разработки политики качества и целей в области качества, а также планирования качества;

— создания системы менеджмента качества;

— проведения анализа деятельности со стороны руководства;

— обеспечения уверенности в наличии ресурсов.

Требования заказчика — высшее руководство должно обеспечить:

— потребности и ожидания заказчика установлены и переведены в соответствующие требования заказчика;

— требования заказчика полностью поняты разработчиком и могут быть удовлетворены.

Высшее руководство должно разработать ***политику в области качества*** и обеспечить уверенность в том, что она:

— соответствует потребностям организации и ее заказчиков;

— включает обязательства по удовлетворению требований и постоянному улучшению;

— обеспечивает основу для разработки и анализа целей в области качества;

— распространена, понята и внедрена во всей организации;

— анализируется с целью постоянного поддержания ее пригодности.

Планирование — организация должна установить цели в области качества для каждой соответствующей функции и для каждого уровня внутри организации. Цели в области качества должны соответствовать политике и обязательствам относительно непрерывного улучшения качества. Организация должна установить и планировать деятельность и ресурсы, необходимые для достижения целей в области качества. Такое планирование должно отвечать другим требованиям к системе менеджмента качества, а его результаты должны быть документированы. Планирование должно охватывать:

— процессы, необходимые в рамках системы менеджмента качества;

— процессы создания и необходимые ресурсы, а также установленные характеристики качества на различных стадиях, с целью достижения планируемых результатов;

— деятельность по проверке, критерии приемлемости и необходимые отчеты по качеству.

Система менеджмента качества — организация должна создать систему менеджмента качества, как средство реализации ее политики в области качества, достижения своих целей в области качества и обеспечения уверенности в том, что продукция и/или услуга отвечает требованиям заказчика. Роли сотрудников и их взаимосвязи, а также ответственность и полномочия персонала должны быть установлены для того, чтобы способствовать эффективному управлению качеством, и должны быть доведены до соответствующих уровней организации. Высшее руководство должно уполномочить одного (или нескольких) лиц для:

— обеспечения уверенности в том, что система менеджмента качества внедрена и поддерживается в рабочем состоянии в соответствии с требованиями настоящего международного стандарта;

— доклада высшему руководству о функционировании системы менеджмента качества, включая вопросы, связанные с необходимостью ее улучшения;

— обеспечения уверенности в осознании во всей организации требований заказчика.

Организация должна разработать ***Руководство по качеству***, которое должно включать: описание элементов системы менеджмента качества и их взаимосвязей; общесистемные процедуры или соответствующие ссылки на них. Следует установить ***общесистемные процедуры для управления документами***, необходимыми для функционирования системы менеджмента качества, обеспечивающие уверенность в том, что:

— документы проверены на адекватность до их применения;

— документы анализируются, при необходимости уточняются и переутверждаются;

— соответствующие выпуски документов находятся в тех местах, где осуществляется деятельность, имеющая существенное значение для эффективности функционирования системы менеджмента качества;

— устаревшие документы изъяты из всех мест их рассылки и применения или предприняты другие методы управления, предотвращающие их непреднамеренное использование;

— любые устаревшие документы, оставленные для юридических целей или в целях сохранения знаний, должным образом идентифицированы.

Должен быть составлен специальный перечень или эквивалентная процедура управления, идентифицирующая ***статус текущей версии документов***, которая была бы легко доступна в целях предотвращения использования недействительных и/или устаревших документов.

Для демонстрации соответствия требованиям и эффективности функционирования системы менеджмента качества должны вестись подходящие для организации ***отчеты о качестве***. Организация должна создать и поддерживать в рабочем состоянии процедуры общесистемного уровня по идентификации, хранению, восстановлению, обеспечению сохранности, установлению времени и места хранения отчетов о качестве. Анализ со стороны руководства должен через установленные периоды времени проводиться для обеспечения уверенности в сохранении ее пригодности, адекватности и эффективности. По результатам анализа должна проводиться оценка необходимости ***внесения изменений в систему менеджмента качества*** организации, включая политику и цели в области качества.

Управление ресурсами необходимо для создания и поддержания в рабочем состоянии системы менеджмента качества. Организация должна проводить анализ и назначение персонала с целью обеспечения уверенности в том, что те, кто имеет обязанности, определенные системой менеджмента качества, являются компетентными для осуществления своей деятельности на основе соответствующего образования, подготовки, мастерства и опыта, создать и поддерживать в рабочем состоянии общесистемные процедуры по:

— определению потребностей в компетентном персонале и в его подготовке;

— обеспечению подготовки персонала в соответствии с выявленными потребностями;

— оценке через установленный период времени эффективности подготовки кадров;

— ведению соответствующих отчетов об образовании кадров, их подготовке, уровне мастерства и опыта.

Организация должна создать и поддерживать в рабочем состоянии процедуры, обеспечивающие их осознание работниками в соответствующих службах и на соответствующих уровнях:

— важности соответствия политике в области качества и требованиям к системе менеджмента качества важности;

— влияния их деятельности на качество — фактическое или потенциальное;

— выгоды от улучшения работы персонала;

— своей роли и ответственности в достижении соответствия политике в области качества и процедурам, а также требованиям к системе менеджмента качества;

— потенциальных последствий отклонений от установленных процедур.

Организация должна установить ***перечень информации***, которая необходима для управления процессами, а также для обеспечения уверенности в соответствии продукции и/или услуги. Общесистемные процедуры по управлению информацией должны обеспечить уверенность в доступности и сохранности информации.

Организация должна определить, создать и поддерживать ***в рабочем состоянии инфраструктуру***, необходимую для достижения требуемого качества продукции:

— рабочие места и соответствующие помещения;

— оборудование, вспомогательные средства и инструментальное программное обеспечение;

— пригодные способы поддержания работоспособности инфраструктуры.

Процессы, необходимые для ***выпуска требуемой продукции***, их последовательность и взаимосвязи должны быть определены, спланированы и внедрены. При определении таких процессов организация должна учесть результаты планирования качества. Должна быть уверенность в том, что эти процессы осуществляются в управляемых условиях и их результаты соответствуют требованиям заказчика:

— установить для этих процессов соответствующие методы и практику выполнения, необходимые для обеспечения постоянной работоспособности процессов;

— определить и внедрить критерии и методы управления процессами для обеспечения соответствия продукции требованиям заказчика;

— удостовериться, что процессы могут функционировать в той мере, которая позволяет обеспечить соответствие продукции требованиям заказчика;

— обеспечить уверенность в наличии информации и данных, необходимых для поддержания эффективного функционирования и мониторинга процессов;

— фиксировать в виде отчетов качество результатов измерений, осуществляемых в ходе управления процессами, для предоставления доказательств эффективного функционирования и мониторинга процессов.

Организация должна создать ***процесс идентификации требований заказчика***:

— полноту требований заказчика к продукции;

— требования, не установленные заказчиком, но необходимые для применения продукции;

— обязательства по отношению к продукции, включая регламентирующие и законодательные требования;

— требования заказчика относительно пригодности продукции для ее поставок и сопровождения.

Требования заказчика, включая любые предлагаемые изменения, должны быть проанализированы для обеспечения уверенности в том, что:

— требования заказчика к продукции четко определены;

— в случае, когда требования заказчика не оформлены письменно, он подтвердил их до принятия их разработчиком;

— организация располагает возможностями для удовлетворения требований заказчика к продукции.

Организация должна ***планировать и управлять проектированием и/или разработкой продукции***, подготавливать планы проектирования, которые включают:

— этапы процесса проектирования и/или разработки;

— требуемые действия по анализу, проверке и утверждению качества продукции;

— полномочия и ответственность за действия по проектированию и/или разработке.

Входные данные для проектирования и разработки должны включать:

— эксплуатационные требования заказчика или рынка;

— применяемые нормативные и законодательные требования;

— применяемые требования по охране окружающей среды;

— любые другие требования, существенные для проектирования и разработки.

Выходные данные процесса проектирования и/или разработки должны быть зарегистрированы в форме, дающей возможность проверки их по отношению к входным требованиям:

— соответствовать входным требованиям для проектирования и/или разработки;

— содержать или давать ссылку на критерий приемки продукции и/или услуги;

— определять характеристики продукции и/или услуги, которые являются существенными с точки зрения безопасности и правильного использования.

На соответствующих этапах должен проводиться ***систематический анализ проекта*** для: оценки возможности выполнения требований к качеству; идентификации проблем — дефектов и выработки предложений по разработке решений для их устранения. В состав участников анализа проекта должны включаться представители служб, связанных с анализируемым этапом проектирования. Должна быть спланирована и выполнена проверка проекта и/или разработки, обеспечивающая уверенность в том, что выходные данные соответствуют входным требованиям.

Утверждение проекта должно проводиться с целью подтверждения того, что конечная продукция способна отвечать требованиям для конкретных условий использования заказчиком. Когда это возможно, утверждение должно быть спланировано и выполнено до начала поставки или применения продукции. Результаты утверждения и последующих действий должны быть зарегистрированы.

Изменения проекта или модификация должны быть утверждены уполномоченным персоналом и зарегистрированы до их внедрения, следует определить влияние изменений на:

— взаимодействие между элементами проекта и/или разработки:

— взаимодействие между составными частями конечной продукции;

— имеющуюся продукцию и на функционирование ранее поставленной продукции;

— необходимость проведения повторной проверки или утверждения для всех или части выходных данных проектирования и/или разработки.

Организация должна ***управлять процессами закупки*** компонентов для обеспечения уверенности в соответствии закупленной продукции требованиям, установленным заказчиком. Закупочные документы должны содержать информацию, четко описывающую заказанную продукцию.

Организация должна ***спланировать и управлять производственными и сервисными операциями обслуживания***, включая те, которые предпринимаются после первоначальной поставки, посредством:

— предоставления технических условий, определяющих характеристики продукции, которые должны быть достигнуты;

— предоставления четко понимаемых производственных требований или инструкций для тех видов деятельности, где они необходимы для достижения соответствия продукции;

— внедрения надлежащих действий по мониторингу или проверке продукции;

— подходящих методов для выпуска, поставки и/или монтажа продукции.

Меры по утверждению процессов должны быть направлены на: аттестацию процессов до их использования; аттестацию оборудования и/или персонала.

Следует определить, спланировать и внедрить процессы измерений, мониторинга, анализа и улучшения проекта для обеспечения уверенности в том, что система менеджмента качества, процессы и продукция соответствуют установленным требованиям. Эффективность применяемых измерений должна периодически оцениваться. Результаты анализов данных и действий по улучшению должны служить исходными данными для процесса анализа со стороны руководства.

Организация должна определить и установить ***процессы измерения и функционирования системы менеджмента качества***. Удовлетворенность заказчика должна использоваться в качестве одного из измеряемых параметров результатов действия системы. Организация должна применять методы измерения и мониторинга процессов, необходимые для удов-

летворения требований заказчика и для демонстрации постоянной способности процессов удовлетворять поставленным целям. Результаты измерений должны использоваться для поддержания в рабочем состоянии и/или улучшения этих процессов. Данные отчеты должны указывать уполномоченных лиц, ответственных за выпуск продукции.

Следует обеспечить уверенность в том, что продукция, которая ***не соответствует требованиям***, находится под управлением, обеспечивающим предотвращение ее непреднамеренного использования или поставки. Механизм, обеспечивающий уверенность в том, что несоответствующая продукция находится под управлением, должен быть определен в общесистемной процедуре. Продукция, имеющая несоответствия, должна быть, например:

— подвергнута коррекции или исправлению с целью обеспечения соответствия требованиям;

— принята на основании разрешения на отклонение (с коррекцией или исправлением или без них);

— перераспределена для разрешенного альтернативного использования;

— удалена (отбракована) как неприемлемая.

Должны быть определены ответственность и полномочия по проведению анализа и принятию решений по несоответствиям. Там, где это определено контрактом, информация о предлагаемом использовании или ремонте несоответствующей продукции должна направляться заказчику для получения разрешения на отклонение.

Для ***анализа совершенствования процессов*** должна быть установлена общесистемная процедура, направленная на определение эффективности системы менеджмента качества и выявление мест, где могут быть сделаны улучшения. Организация должна собирать данные, появляющиеся в результате действий по измерению и мониторингу, а также из любых других приемлемых источников.

Следует ***постоянно улучшать систему менеджмента качества***, установить общесистемную процедуру, которая определяет использование политики качества, целей, результатов внутреннего аудита, анализа данных, корректирующих и предупреждающих действий и анализа со стороны руководства в целях поддержки постоянных улучшений. Установить

процесс, направленный на сокращение или исключение причин несоответствий для предотвращения повторения несоответствий.

Общесистемная процедура для ***процесса проведения корректирующих действий*** должна определять требования по:

— идентификации несоответствий;

— определению причин несоответствий;

— реализации всех действий, обусловленных необходимостью обеспечения уверенности в том, что несоответствия не повторятся;

— регистрации результатов предпринятых действий;

— анализу эффективности и регистрации предпринятых корректирующих действий.

Следует установить процесс, направленный на ***исключение причин потенциальных несоответствий требованиям*** для предупреждения их появления. Отчеты системы менеджмента качества и результаты, полученные из анализа данных, должны использоваться в качестве исходных данных для предупреждающих действий.

Стандарт **ISO 90003:2004** — Рекомендации по применению стандарта **ISO 9001:2000** для программных средств — предназначен для регламентирования менеджмента при приобретении, поставке, разработке, применении, сопровождении ***сложных программных средств*** и при их обслуживании. Стандарт не содержит ограничений и изменений базовых требований **ISO 9001:2000** и предлагается при установлении соответствия требованиям комплексов программ:

— как части коммерческого контракта с другими организациями;

— при представлении полезного продукта для рынка;

— для использования при поддержке процессов организации проектов ПС;

— для учета при встраивании программных средств в комплексы аппаратуры;

— при организации сервиса программных продуктов.

Полное или частичное применение стандарта **ISO 90003** целесообразно в различных ситуациях, с учетом технологии, модели жизненного цикла, процессов разработки, последовательности действий и организационной структуры предприятия. Его рекомендуется применять как ***поддержку процессов программной инженерии*** в **ISO 9001:2000**, совместно со стандартами **ISO 12207, ISO 15504, ISO 9126, ISO 14598, ISO 15939**.

Первые четыре раздела практически повторяют содержание аналогичных разделов в **ISO 9001:2000**. Структура стандарта **ISO 90003:2004** привязана к разделам и требованиям в **ISO 9001:2000**, которые цитируются в начале каждого раздела. ***Пятый раздел*** определяет ответственность руководства: общие обязанности руководства управления проектом; политику в области обеспечения качества продукции; планирование управления проектом; ответственность и полномочия специалистов; анализ проектирования со стороны руководства. Формулировки в **ISO 90003** повторяют содержание основного стандарта с очень небольшими комментариями.

В ***шестом разделе*** — менеджмент ресурсов — более полно комментируются особенности ***управления ресурсами в области программной инженерии***. Внимание акцентируется: на проблемах обеспечения ограниченными вычислительными ресурсами инфраструктуры проектов; на компетентности, квалификации и подготовке специалистов; на управлении производственной средой. При этом неоднократны подробные ссылки на требования стандартов **ISO 12207, ISO 15504, ISO 9126, ISO 14598** (см. Приложение 1).

Наиболее обширным и специфическим, практически полностью ориентированным на программные продукты, является ***седьмой раздел*** стандарта. В нем изложено планирование и управление процессами и качеством жизненного цикла программных средств с попутными ссылками на рекомендации перечисленных выше стандартов. Рекомендации проектирования и разработки имеют традиционную структуру жизненного цикла ПС: входные и выходные данные процессов; анализ требований; верификация и валидация результатов; управление изменениями и сопровождение; мониторинг и измерение результатов. При этом формулировки и цитаты разделов **ISO 9001** отходят на второй план. Небольшой ***восьмой раздел*** посвящен измерениям, анализу и совершенствованию процессов и результатов управления проектами программных продуктов и почти не связан с требованиями **ISO 9001**.

В стандарте **ISO 90003** имеется ***два оригинальных приложения,*** полностью отличающихся от приложений в **ISO 9001**. Приложение А составляет обширная таблица, в которой процессам стандарта **ISO 90003** сопоставлены полезные детализирующие процессы около 20 основных стандартов жизненного цикла сложных программных средств. В таблице приложе-

ния В сопоставлены рекомендации по планированию программных проектов в стандартах **ISO 90003** и **ISO 12207**, которые весьма близки.

Практически все ***перечисленные процессы и требования стандартов*** в данном разделе конкретизированы в очень подробных рекомендациях процессов (около 500 страниц — седьмой раздел) трех выделенных выше ***уровней модели*** **CMMI** (см. п. 3.1). Они ***соответствуют*** регламентированным и детализированным рекомендациям в стандартах **ISO 9001:2000, ISO 12207** и основных компонентах ***профиля стандартов жизненного цикла сложных ПС*** (см. лекцию 2). Требованиям в функциональных разделах 4—8 стандарта **ISO 9001** могут быть сопоставлены подобные по содержанию разделы в ***поэтапной модели*** **CMMI**. Общность процессов и требований состоит в подобии: терминологии, структуры, рекомендуемых процессов управления, планирования, учета доступных ресурсов, реализации процессов, оценивания и организации специалистов. Процессы, которые развиваются и детально комментируются процессами их реализации в стандартах **ISO 9004:2000** и **ISO 90003:2004**, а также в представленном выше профиле, включающем около пятидесяти стандартов **ISO** (см. Приложение 1), однако, не всегда предусмотрены в рекомендациях и ссылках **CMMI**.

При практической реализации проектов и обеспечении ***жизненного цикла сложных ПС*** разработчикам и поставщикам может быть сложно определить и выделить для применения преимущества моделей **CMMI** или ***профиля стандартов*** **ISO**. В зависимости от традиций предприятия и особенностей проекта ПС зачастую целесообразно использовать как основной полный ***профиль стандартов*** **ISO**, а для оценивания заказчиками ***уровня зрелости*** менеджмента, организационного и технологического обеспечения проектов ПС применять конкретные рекомендации **CMMI**. Эти рекомендации могут эффективно использоваться при ***сертификации качества процессов*** на предприятиях, обеспечивающих ЖЦ ПС, как альтернатива или наряду с сертификацией по комплексу стандартов менеджмента **ISO 9000**, в зависимости от особенностей проекта и требований заявителя на сертификацию программного продукта и/или технологии обеспечения его жизненного цикла.

При ***подготовке системы качества предприятия*** целесообразно учитывать и использовать совокупность рекомендаций ряда стандартов, под-

держивающих и детализирующих базовые стандарты серии **ISO 9000**, в которую входят — **ISO 10005, ISO 10006, ISO 10013, ISO 10011**.

Стандарт* ISO 10006:1997** — Руководство по качеству при управлении проектом — содержит принципы управления качеством различных по содержанию крупных проектов. В нем изложены рекомендации по административному ***управлению качеством процесса проектирования сложных объектов и в том числе программных средств. Приводятся определения процесса, продукции и плана управления проектом, а также общие характеристики организации и фазы проектирования. Обеспечение качества управления проектом представлено группой процессов, для каждого из которых приводятся подробные рекомендации по их реализации и контролю качества выполнения.

В стандарте* ISO 10013:1995** — Руководящие указания по разработке руководств по качеству — изложены рекомендации по подготовке конкретного ***Руководства по качеству, адаптированного к определенным потребностям предприятия и пользователей. Созданное в результате Руководство должно отражать документированные процедуры системы качества конкретного предприятия или проекта в соответствии с общими требованиями стандартов серии **ISO 9000**. В этом документе должны быть изложены: цели конкретной системы качества проекта или предприятия; документированные процедуры этой системы; организация процессов утверждения, изменения и применения данного Руководства. В стандарте предложена подробная типовая структура и рекомендуемое содержание разделов такого документа.

В стандарте* ISO 10005:1995** — Административное управление качеством. Руководящие указания по Программе качества — представлены конкретные рекомендации по структуре и содержанию разделов в ***Программе обеспечения качества продукции, в соответствии с базовыми требованиями стандарта **ISO 9001**, а также примеры документального оформления таких Программ. Для эффективного применения его ***следует адаптировать к характеристикам объектов и среды применения конкретного предприятия или проекта***. Прежде всего, необходимо оставить в рабочей версии этого стандарта разделы и положения, которые целесообразно непосредственно использовать для обеспечения качества конкретных изделий. Адаптация стандарта для конкретных предприятий или проектов может выполняться путем сокращения и конкретизации некоторых

положений. После этого Программа качества должна быть сформирована и оформлена как самостоятельный регламентирующий документ, согласована с заказчиком, утверждена руководством предприятия и доведена до сведения всех участников проекта для практического применения. При сертификации системы качества предприятия должно проверяться наличие и практическое использование всех положений утвержденной рабочей версии Программы качества.

Группа стандартов — **ISO 10011:1-3:1990** — Руководящие положения по проверке систем качества — определяет основные ***требования к процессам и специалистам по оценке систем качества предприятия*** на соответствие стандартам серии **ISO 9000**. В них изложены основные принципы, критерии и методики, а также руководящие положения для разработки, планирования и ведения документации при проверке систем качества предприятия. Определены обязанности и ответственность ***независимых инспекторов по проверке системы качества***, требования к ним, порядок и критерии оценки и выбора инспекторов, их аттестации и условия выдачи свидетельств для допуска к инспектированию. Для независимой и объективной оценки системы качества предприятия или проекта рекомендуется проводить специальный отбор испытателей — инспекторов по сертификации. В соответствии со стандартом **ISO 10011-2** ***кандидаты в инспекторы по проверке систем качества*** должны продемонстрировать способность четко и быстро выражать концепции и идеи в устной и письменной форме. Они должны пройти обучение, дающее им знания и квалификацию, необходимые для проведения испытаний и управления проверками систем качества.

3.3. Стандарты открытых систем, регламентирующие структуру и интерфейсы программных средств

Рядом зарубежных организаций и промышленных фирм под руководством **IEEE** с 1990 года ведется активная разработка последовательных версий стандартов интерфейсов открытых систем **POSIX** (Portable operating system interfaces). Выполнена большая работа по пересмотру, расширению и реорганизации около двадцати базовых спецификаций **POSIX** 1990—1998 годов **IEEE 1003**. Улучшена систематизация и структура стандартов,

усовершенствовано удобство их применения пользователями. В результате подготовлен ***комплексный проект фундаментального международного стандарта из четырех крупных частей*** **ISO 9945:1-4:2003 (IEEE 1003.1 — 2003)**, объемом свыше трех тысяч страниц. Настоящий стандарт — совместная разработка **IEEE** и **The Open Group**, он является одновременно стандартом **IEEE**, стандартом **ISO** и стандартом **Open Group Technical**.

Цель документа — стандартизация в ***программной инженерии*** обеспечения переносимости программ на уровне исходных текстов. В нем определены основные интерфейсы операционных систем и окружения, интерфейсы командного интерпретатора, а также программы общих утилит. ***Три отдельных крупных тома включают***: базовые определения; системные интерфейсы; команды управления и сервисные программы (утилиты). Кроме того, имеется большой четвертый том общего обоснования выбранных решений системы **POSIX**. Важными свойствами разработанных программных интерфейсов являются целостность, модульность их построения и параметризуемость.

Стандарты открытых систем — **POSIX** регламентируют совокупность базовых, системных сервисов для обеспечения унифицированных интерфейсов прикладных программ, специфицированных для языка Си, командного языка и совокупности служебных программ. ***Основная цель*** — сделать программы переносимыми на уровне различных исходных языков. У каждого интерфейса программ существует вызывающая и вызываемая сторона, стандарты **POSIX** ориентированы преимущественно на формализацию вызывающей стороны. Мобильность приложений должна обеспечиваться благодаря применению большого числа стандартизированных системных интерфейсных сервисов и возможности динамического выяснения характеристик целевой платформы и подстройки под них интерфейсов приложений.

При формировании ***концепции стандартов POSIX были поставлены следующие задачи***:

— содействовать облегчению и автоматизации переноса кода готовых прикладных программ на иные платформы;

— способствовать определению и унификации интерфейсов программных компонентов заранее при проектировании программных средств, а не только в процессе их реализации;

— сохранять по возможности и учитывать все главные, созданные ранее, унаследованные и используемые программные средства и компоненты;

— определять необходимый минимум интерфейсов компонентов и комплексов программ для ускорения создания и расширения программных продуктов, а также для анализа, одобрения и утверждения документов;

— развивать стандарты в направлении обеспечения коммуникационных сетей, распределенной обработки данных и защиты информации;

— рекомендовать ограничивать использование объектного кода для программ в простых системах.

Разработчики ***новых версий стандартов*** группы **POSIX** тщательно учитывали их предысторию и наличие множества унаследованных, созданных и развиваемых компонентов и комплексов программ, удовлетворяющих более ранним версиям этих стандартов. В процессе развития стандартов соблюдался принцип обратной совместимости — новые интерфейсы добавлялись так, чтобы они не конфликтовали со старыми. Однако полностью это не удалось реализовать, и некоторые интерфейсы в повторно применяемых программах необходимо корректировать при их использовании в новых программных средствах.

Цель данной версии стандарта состоит в том, чтобы минимизировать дополнительную работу для разработчиков прикладных программ. Тем не менее, поскольку каждая историческая реализация стандарта неизбежно подвергается изменению, пусть даже незначительному, прикладным программам также неизбежно предстоит изменяться, чтобы прийти в соответствие. Концептуально этот стандарт описывает набор фундаментальных услуг, необходимых для эффективной разработки прикладных программ. Доступ к этим услугам обеспечивался определением интерфейса, с использованием языка программирования Си, процессора командного языка, и стандартных сервисных программ (утилит), которые устанавливают стандартную семантику и синтаксис интерфейсов. Этот стандарт предназначен для всех, кто по роду своей деятельности связан с операционными системами, базирующимся на **UNIX**.

Цель состояла также в том, чтобы продвинуть мобильность прикладных программ по всей области применения операционной системы **UNIX**, развивая ясный, последовательный и однозначный стандарт для спецификации интерфейса переносимой операционной системы, основанной на

документации системы **UNIX**. Этот стандарт систематизирует все существующие типовые формулировки операционной системы **UNIX**. Стандарт был разработан для того, чтобы программа, написанная и транслированная для исполнения в одной конкретной версии ОС, могла быть также транслирована для другой версии ОС. Однако этот стандарт не гарантирует, что используемый код (объектный) даст соответствующий результат при другой ОС, нежели той, для которой программа была транслирована, даже при идентичном аппаратном обеспечении.

Разработчики данного стандарта стремились добиться возможности наиболее широкого применения данного стандарта ко всему диапазону уже существующих и потенциальных операционных систем. Однако существовало согласованное мнение по поводу того, какой набор функций, типов, определений (дефиниций) и концепций предназначен для формирования того интерфейса, который является общим для большинства исторически сложившихся типов реализации ПС. Предыдущие стандарты и их модификации выявили множество несогласованных областей, что в основном устранено в данном варианте. Пересмотренное издание стандарта стремится минимизировать количество изменений, касающихся реализаций, которые соответствуют более ранним версиям одобренных стандартов, необходимых для приведения их в соответствие с настоящим стандартом.

Новую версию ***международных стандартов*** **POSIX** составляют аннотированные ниже четыре части стандартов **ISO 9945:1-4:2003** — ИТ. Интерфейсы переносимых операционных систем. Ч. 1. Базовые определения. Ч. 2. Системные интерфейсы. Ч. 3. Команды управления и сервисные программы. Ч. 4. Обоснование.

Стандарт **ISO 9945-1:2003** — содержит основные концептуальные определения и подробные пояснения методов реализации интерфейсов для обеспечения мобильности компонентов и комплексов программ, общее для всех томов оглавление стандарта, в том числе сервисные соглашения и определения заголовков языка Си. В большом разделе — ***концепция*** — изложены: базовые директории; файловая иерархия; сетевое взаимодействие; измерение времени и синхронизация; процессы повторного использования компонентов; политика очередей и применение семафоров. Далее выделены ***разделы***:

— описание синтаксиса и организации данных в файлах для системных интерфейсов и взаимодействия с терминалами;

— характеристики переносимых компонентов, язык кодирования, описание файлов;

— локальные переменные, определения и грамматика;

— описание переменных окружения, интернационализация переменных;

— контекстно-независимый синтаксис представления характеристик переменных компонентов, определения регулярных выражений, генерация требований, базис и грамматика выражений, продленные выражения;

— структура директорий, файлов и устройств, типы терминалов;

— базовые интерфейсы типов терминалов, параметры возможных устройств;

— конвенция и руководство по синтаксису аргументов утилит;

— содержание функций прототипов, описание символьных констант, макросы, препроцессоры, типы определений, форматы входов.

Стандарт* ISO 9945-2:2003** — Системные интерфейсы — уточняет и детализирует: концепцию переносимости и принципы ее обеспечения путем унификации интерфейсов прикладных программ с операционными системами, функции обслуживания, ориентированные на язык программирования Си, функциональные вопросы, в том числе мобильность, обработка ошибок, и устранение ошибок. Он содержит ***три крупных раздела:

— введение;

— содержание основной информации;

— системные интерфейсы.

Документ предназначен для разработчиков приложений, в которых необходимо обеспечить мобильность программ и данных, для разработчиков и пользователей операционных систем, а также для покупателей вычислительной техники и программных средств. Основная цель стандарта — унифицировать интерфейсы приложений и связи с окружением ядра операционной системы путем формализации описаний внутренних и внешних интерфейсов на базовом языке программирования Си. Концептуально представлено описание синтаксиса и семантики взаимосвязей, используемых при проектировании переносимых прикладных программ. Унификация ориентирована на версии UNIX, а также на другие операционные системы, совместимые по интерфейсам с версиями UNIX. Разработчики стандарта

стремились предусмотреть все потенциально возможные функции взаимодействия прикладных программ с ядром ОС на различных аппаратных платформах, в автономных, сетевых и распределенных информационных системах. Для этого представлено содержание 1123 интерфейсов, включающих: описание, типовую структуру, возможные ошибки, примеры и обоснование каждого.

Языково-ориентированные услуги и интерфейсы для языка Си в стандарте состоят из двух частей: ядра, взаимодействующего с ядром операционной системы на языке Си, и расширения для взаимодействия с прикладными программами, реализованными на различных языках. Первоначальная, жесткая ориентация стандартов POSIX на ОС UNIX в последующем изменилась и расширилась на любые операционные среды, обеспечивающие реализацию концепции открытых систем. Отмечается целесообразность применения для взаимодействия с внешней средой концепции и совокупности стандартов, являющихся развитием базовой эталонной модели взаимосвязи открытых систем (ВОС) — **ISO 7498**.

В третьей части ***стандарта*** **ISO 9945-3:2003** — Основные команды управления и сервисные программы (Shell and utilities) — изложено конкретное представление команд операционной системы и утилит, обеспечивающих унифицированное взаимодействие с мобильными прикладными программами, определения для стандартного источника кодового уровня интерфейса командного интерпретатора («shell») и стандартные утилиты для прикладных программ. Стандарт содержит ***четыре раздела***:

— введение;

— описание языка управления;

— пакет обслуживания окружения;

— сервисные программы — утилиты.

Цель этой части стандарта — конкретизировать интерфейсы прикладных программ на уровне команд, для интерпретатора командного языка, и полный набор сервисных программ — утилит. Он специфицирует интерфейсы операционных систем на уровне программных текстов и кодов. Стандартизированный командный язык Shell — средство для подготовки небольших мобильных процедур и их быстрой интерактивной отладки. В развитой среде возможно объединять команды в цепочки с фильтрацией промежуточных результатов. Каждая утилита содержит: описание структуры; применение; операнды; входные файлы; окружение. Факуль-

тативные утилиты расширяют возможности для пользователей мобильных приложений по связи с внешним окружением. В терминах языка Си (стандарт **ISO 9899:1990**) описаны языково-независимые услуги интерфейсов для переносимых приложений и связи с внешним окружением при представлении интерфейсов приложений на различных языках высокого уровня. Команды и утилиты содержат конкретные механизмы на уровне операторов для выполнения операций: сравнения, вывода на печать и на экран файлов, редактирования файлов, вычисления выражений, сортировки данных, определения очередности исполнения сигналов и доступа к информации о среде. Можно выделить:

— утилиты среды взаимодействия программ;

— управляющие утилиты, поддерживающие мобильность программ и связь внешних пользователей с асинхронными терминалами;

— утилиты для взаимодействия комплексов программ;

— языково-независимые системные услуги для прикладных программ на нескольких языках программирования высокого уровня.

Четвертая часть ***стандарта* ISO 9945-4:2003** — Обоснование — содержит группу из ***пяти крупных приложений***:

— обоснование базовых определений — приложение A;

— обоснование системных интерфейсов — приложение B;

— обоснование команд управления и сервисных программ-утилит — приложение C;

— рассмотрение переносимости — приложение D;

— рассмотрение субпрофилей — приложение E.

В перечисленных приложениях подробно разъясняются базовые положения стандартов POSIX, обосновываются формализованные в них решения. Пространные пояснения, которые не слишком вписывались в остальные части документа, содержат исторические комментарии по тексту стандарта, разъяснения причин, по которым те или иные свойства, признаки и компоненты были включены в стандарт или были отвергнуты разработчиками.

Параллельно с подготовкой и внедрением новых четырех стандартов группы **POSIX** действуют и применяются представленные ниже, утвержденные базовые ***международные стандарты***, углубляющие некоторые возможности и облегчающие создание мобильных приложений.

В ***стандарте*** **ISO 14252:1996** — Руководство по **POSIX** окружению открытых систем (OSE) — изложена идеология и модель создания мобильных программных средств, которое детализирует для пользователей модель комплекса стандартов POSIX, а также взаимодействующих с ними стандартов де-юре, де-факто и спецификаций, необходимых для создания переносимых приложений. Модель отражает принципы построения интерфейсов прикладных программ с платформой — операционной системой, через которую осуществляется взаимодействие с компонентами внешнего окружения. Считается, что прикладные программы непосредственно не взаимодействуют с внешним окружением, а связаны с ним только через операционную систему. Разработка приложений предполагается в кросс-режиме, то есть платформа разработки — инструментальная может не совпадать с платформой исполнения (применения) программ (объектной — целевой). Результат компиляции программы на инструментальной платформе может быть перенесен для исполнения на целевую платформу.

Определяющими являются два интерфейса между тремя базовыми компонентами: между прикладными программами и платформой — операционной системой (API) и между платформой и внешним окружением (EEI). Определены общие функции — услуги платформы для этих взаимодействий. Внешнее окружение включает компоненты человеко-машинного взаимодействия с пользователями, компоненты информационного взаимодействия с внешними устройствами ЭВМ и с компонентами, обеспечивающими коммуникацию с внешней средой. Эти интерфейсы и услуги более детально формализованы соответствующими стандартами POSIX.

Стандарт включает также общие принципы и руководство по формированию и описанию сложных согласованных профилей и по обеспечению непротиворечивости их интерфейсов с внешним окружением;

— поддерживающие непосредственное взаимодействие прикладных программ с пользователями, регламентирующие интерфейсы с виртуальными терминалами и графические системы;

— обеспечивающие административное управление файловыми системами и различными, в том числе распределенными, базами данных;

— регламентирующие телекоммуникацию и обмен данными на верхних уровнях эталонной модели ВОС;

— обеспечивающие аттестацию и безопасность применения информационных технологий, программ и данных в соответствии со стандартами ВОС и криптографии.

***Стандарт* ISO 13210:1998** — Методы тестирования для оценки соответствия стандартам **POSIX** — содержит общую методологию тестирования для проверки соответствия интерфейсов прикладных программ стандартам **POSIX**. Представлены принципы формирования наборов тестов и предлагается методика развития системы тестов для контроля создаваемых приложений на соответствие **ISO 9945**. В методе тестирования выделены: подготовка тестов; процедуры тестирования; оформление документов, удостоверяющих соответствие стандартам. Определены и кратко представлены три основных уровня сложности тестирования: исчерпывающий; достаточно полный; оценочный — для установления общих характеристик качества. Подчеркивается, что предлагаемые методы не гарантируют абсолютное соответствие стандартам, так как исчерпывающее тестирование невозможно по техническим и экономическим причинам. Приводятся рекомендации по:

— классификации тестовых сценариев и утверждений;

— подготовке описаний детерминированных тестов и тестов для проверки структурных конструкций;

— кодированию и представлению результатов тестирования во взаимосвязи с исходными тестами;

— формированию удостоверения о соответствии интерфейсов прикладных программ стандартам **POSIX**.

В стандартах POSIX даются ссылки на многие стандарты **ISO, ANSI** и де-факто. В результате полная совокупность стандартов, поддерживающих проектирование ***мобильных программ*** в разных классах открытых систем, возрастает. В конкретных прикладных разработках каждый раз необходима только некоторая, относительно небольшая часть из них, которую целесообразно выделять в соответствующий ***проблемно-ориентированный профиль***.

ЛЕКЦИЯ 4
СИСТЕМНОЕ ПРОЕКТИРОВАНИЕ ПРОГРАММНЫХ СРЕДСТВ

4.1. Цели и принципы системного проектирования сложных программных средств

Комплекс формально организованных мероприятий по достижению единой цели создания сложной системы с требуемыми характеристиками качества при ограниченных ресурсах получил название — ***проект***. ***Цель управления проектом*** — рациональное использование и предупреждение потери ресурсов путем сбалансированного распределения их по частным работам на протяжении всего жизненного цикла объекта с заданным качеством. Управление проектом — это особый вид деятельности, включающий постановку задач, подготовку решений, планирование, организацию и стимулирование специалистов, контроль хода работ и использования ресурсов при создании сложных систем.

Целевое управление проектами возникло из необходимости разрабатывать и реализовывать сложные системы с заданными функциями в максимально короткие сроки при ограниченных ресурсах. Критическим параметром планирования и управления проектами обычно является ***время***. Поэтому в лекциях по программной инженерии большое внимание сосредоточено на конкретном планировании сложных проектов, длительность разработки которых могут составлять несколько месяцев или даже годы. Задачи целевого управления такими работами — сводить воедино усилия исполнителей — специалистов разной квалификации, подрядчиков и субподрядчиков, добиваясь, чтобы они выступали ***как команда*** при создании компонентов систем, а не как разрозненная группа функциональных специалистов. В результате должны обеспечиваться концепту-

альная целостность системы и высокое качество решения главных задач при сбалансированном использовании ресурсов на все функциональные задачи.

Для интеграции усилий специалистов и эффективного использования ресурсов проекта ***должен выделяться лидер — менеджер, управляющий проектом***. Он должен активно участвовать в планировании, организации и контроле основных внутренних и внешних организационных мероприятий, необходимых для достижения основной цели проекта. Все ресурсы и исходные данные, необходимые для эффективного выполнения проекта, управляющий получает от функциональных подразделений и специалистов. Задача менеджера проекта наряду с прямыми воздействиями на подчиненных и координацией их работ — стимулировать и контролировать активность прямых горизонтальных связей между исполнителями частных работ. Для того чтобы процесс достижения целей был рациональным, лицо, принимающее решение (управляющий), должно иметь выбор среди альтернативных действий, ведущих к цели. ***Наличие альтернатив и сомнения*** по поводу того, какая из них лучше, определяют возможность эффективного решения проблем и оптимизации путей их достижения.

Методологической базой целевого планирования и управления проектами ПС является системный анализ, который предполагает:

— обследование объектов и среды проектирования для предварительной формализации целей, назначения и задач проекта ПС;

— исследование и сопоставление альтернативных действий, которые должны приводить к достижению поставленных целей проектирования;

— сравнение альтернатив по величине достигаемого эффекта проекта в зависимости от затрат на его достижение (желательно, по показателю «эффективность/стоимость»);

— учет и анализ влияния неопределенностей характеристик альтернатив, определяющих их выбор, на эффект проекта.

Чтобы найти и проанализировать все разумные альтернативы, обычно недостаточно одного специалиста и ***необходимо участие в системном анализе специалистов разной квалификации***. Не во всех системах и задачах оказывается доступным точный количественный анализ. Во многих, чаще всего особенно сложных, случаях приходится ограничиваться качественным анализом свойств, факторов и их влияния на конечный резуль-

тат и возможный эффект проекта. Поэтому оптимизация решений и выбора альтернатив может ограничиваться оценкой логических суждений экспертов. Базой эффективного управления проектом является план, в котором задачи исполнителей частных работ должны быть согласованы с выделяемыми для них ресурсами, а также между собой по результатам и срокам их достижения.

Основная цель системного проектирования в программной инженерии — подготовить, обосновать и согласовать замыслы и решения заказчика (потребителя) и разработчика (поставщика) о необходимости, направлениях и концепции создания или модернизации существующего ПС и изменениях его качества. Методы и средства системного проектирования должны подготавливать эффективную технологическую базу для обеспечения всего жизненного цикла ПС требуемого качества. Характеристики комплексов программ должны анализироваться и формулироваться в начале их жизненного цикла и определять эффективность всех последующих процессов. Результатом этих работ должны быть ***системный проект, техническое задание и контракт*** на продолжение разработки ПС или решение о ее нецелесообразности и прекращении.

Системное проектирование сложных комплексов программ является ***фундаментом для обеспечения функциональной адекватности требованиям всего жизненного цикла ПС***. От полноты и тщательности системного проектирования зависят эффективность реализации функций системы и степень удовлетворения ожиданий и требований заказчика и пользователей. В последовательности выработки и подготовки к реализации этих требований далее рассматриваются в основном ***три крупных этапа***:

— ***обследование***, системный анализ существующей системы и выявление ее недостатков;

— обобщение результатов системного анализа и создание предварительной ***концепции*** новой или модернизированной системы и ее программных средств;

— разработка ***системного проекта*** комплекса программ и базы данных, определяющего и конкретизирующего цель, назначение и методы дальнейшего детального проектирования и всего жизненного цикла ПС.

На этих этапах при относительно небольших затратах должна определяться экономическая эффективность и рентабельность всех последующих больших затрат ресурсов в жизненном цикле системы и могут быть

предотвращены значительные потери ресурсов вследствие плохого планирования и неопределенностей при реализации проекта. ***Системное проектирование способно остановить нерентабельное развитие проектов*** систем и избежать заказчикам и разработчикам на них крупных затрат. В то же время на базе рекомендуемых при проектировании методов, инструментальных средств и стандартов может и должен быть подготовлен и обеспечен длительный, эффективный жизненный цикл и совершенствование множества версий высококачественных ПС и их компонентов при реализации на различных аппаратных и операционных платформах. Конечный результат системного проектирования должен также положительно отражаться на системах обеспечения качества, безопасности и защиты, на рационально организованных коллективах квалифицированных специалистов, способных обеспечить весь жизненный цикл ПС.

Тщательное и полноценное ***системное проектирование выгодно*** с позиции ускорения разработок, предотвращения ошибок, обеспечения эффективности всего жизненного цикла и высокого качества сложных проектов комплексов программ в условиях ограниченных ресурсов. Инициализация системного проектирования может осуществляться заказчиками, потенциальными пользователями или разработчиками проектов ПС, которые желают получить достаточно четкое представление о возможных функциях и будущем жизненном цикле определенного проблемно-ориентированного комплекса программ и базы данных.

Системное проектирование далее структурировано, и методология его реализации отражает следующие проблемы, процессы и методы:

— цели и принципы системного проектирования сложных программных средств для обеспечения их последующего жизненного цикла в информационных системах;

— подготовку к непосредственному детальному проектированию, разработке и всему жизненному циклу комплекса программ и базы данных для информационной системы;

— основы предварительного структурного проектирования и современных технологий системного анализа и проектирования сложных комплексов программ;

— методы разработки требований к характеристикам качества и распределения ресурсов, необходимых для реализации проектов сложных ПС и баз данных;

— задачи и принципы системного проектирования обеспечения безопасности и защиты комплексов программ от различных угроз;

— планирование жизненного цикла и управление качеством ПС, а также выбор инструментальных средств для поддержки всего их ЖЦ;

— методики системного проектирования сложных ПС и требований к их качеству, а также структура и содержание документов, отражающих результаты выполненных работ;

— организацию и подготовку специалистов, способных обеспечить создание детального проекта и всего жизненного цикла ПС с требуемым качеством.

Непрерывное повышение уровня автоматизации в ряде систем при подготовке и принятии ответственных решений возлагает все большие функции на программные средства с соответствующими базами данных. В результате проблема обеспечения качества и безопасности их функционирования ***сдвигается к лицам, разрабатывающим системные проекты,*** методы и программные средства для автоматизации, подготовки и реализации ответственных решений. Одновременно повышаются требования к уровню ответственности и системной квалификации лиц, реализующих комплексы программ, от которых зачастую зависят ошибки стратегически важных решений, так как некоторые простейшие системные и технические ошибки этих лиц при создании программ могут приводить к большому ущербу или даже катастрофам.

Не предусмотренные при системном проектировании ситуации и возможные дефекты программ являются потенциальными источниками отказов и аварий при применении ряда систем. Массовая практика, когда ***заказчик не может сформулировать*** четкие требования к функциям и безопасности ПС, а ***разработчик не понимает***, что нужно заказчику, приводит к длительному процессу разработки проектов с множеством дефектов и ошибок, на устранение которых расходуются большие ресурсы. В результате многие системы не соответствовали исходному назначению и первоначальным спецификациям, не укладывались в графики и бюджет разработки. Поэтому значительно возросла необходимость освоения всех современных методов и методик ***предупреждения системных дефектов***, обеспечения высокого качества программ с самого начала их разработки. Многие ошибки, обусловленные неопределенностью или некорректностью технических заданий и спецификаций, могут и должны быть выявле-

ны на ранних стадиях системного проектирования, что способствует его ускорению и повышению качества. Обширной практикой доказано, что обнаружение и устранение ошибок и дефектов в комплексах программ на начальных этапах системного проектирования ***в десятки и сотни раз быстрее и дешевле, чем в процессе завершения разработки и испытаний***.

В последние десятилетия быстро возрастает сложность объектов и систем, создаваемых в различных областях индустрии. Для их разработки привлекаются специалисты разной квалификации и большие финансовые и материальные ресурсы. Использование этих ресурсов должно координироваться и объединяться комплексом мероприятий для достижения общей цели — создания соответствующего сложного объекта или системы с заданным качеством в условиях ограниченных ресурсов. Современные сложные системы и соответственно проекты, обеспечивающие их создание, имеют ***ряд важных особенностей***:

— единую цель разработки и последующего функционирования всей системы для определенных пользователей;

— наличие совокупности нескольких, тесно взаимодействующих, компонентов — подсистем, имеющих свои локальные задачи и цели функционирования;

— иерархическую структуру связей и взаимодействия компонентов, обеспечивающую концептуальное единство и устойчивость функционирования всей системы;

— иерархическую совокупность критериев качества функционирования компонентов и системы в целом, обеспечивающих достижение главных целей создания и последующего применения системы пользователями.

Одной из особенностей сложных систем является ***трудность выбора и формализации единого критерия качества*** и оценки эффективности функционирования, адекватно отражающего главные цели системы. Обычно выделяется несколько более или менее равнозначных характеристик, каждая из которых может стать доминирующей для оценки эффективности системы в зависимости от этапа ее проектирования, состояния или некоторых внешних условий. Это обусловлено тем, что каждая сложная система зачастую является частью системы большего масштаба и высшего уровня и подчинена ей.

Для управления проектом системы, прежде всего, должны быть ***адекватно описаны цели и объект проектирования***. Для сложных систем формализация и детализация характеристик объекта разработки происходит одновременно с процессом его проектирования. Последовательно уточняются архитектура объекта, основные функции и их характеристики, требующиеся показатели качества функционирования и методы решения задач. Все эти данные отражаются в концепции, техническом задании, спецификации требований и описании проекта, которые детализируются и конкретизируются по мере развития проекта. Это определяет принципиальную особенность планирования проектов сложных систем, состоящую в наличии влияния на план изменяющихся значений и достоверности знаний разработчиков о требуемых характеристиках объекта разработки. С этим связана необходимость итерационного уточнения планов на всех этапах проектирования, разработки и совершенствования систем.

План проекта должен отражать рациональное сочетание целей, стратегий действий, конкретных процедур и доступных ресурсов, необходимых для достижения поставленной ***основной цели проекта с заданным качеством***. Планирование проектов должно ***обеспечивать компромисс*** между требующимися характеристиками создаваемой системы и ограниченными ресурсами, необходимыми на ее разработку и применение. По мере уточнения исходных данных об объекте разработки, внешней среде применения и ресурсах в процессе системного анализа и проектирования возрастает достоверность планирования.

Первичное прогнозирование характеристик проекта и подготовка плана ***при системном проектировании*** происходит при некоторых фиксированных исходных данных, не полностью учитывающих динамику возможного исполнения плана. На этой стадии отсутствует оперативная обратная связь процесса реального выполнения плана с его первичным вариантом. Важнейшая задача при разработке такого плана — минимизировать число связей и сложность взаимодействия между компонентами проекта, а также между исполнителями отдельных компонентов. Уже при первичном прогнозировании развития системного проекта оцениваются альтернативные характеристики объекта и среды разработки и выбираются наиболее подходящие в соответствии с поставленными целями и имеющимися ресурсами.

После создания системного проекта появляется и действует динамическая обратная связь на план со стороны процесса его исполнения. Реализация проекта зависит от результатов выполнения частных работ и может требовать оперативной корректировки плана. При реализации плана определяющими являются координация, стимулирование и контроль развития проекта. Для этого необходимо следить за ходом исполнения проекта на всем протяжении его жизненного цикла и сравнивать запланированные и фактические результаты работ. Контроль является органической функцией управления и должен иметь средства регулирования поведения отдельных личностей и коллектива проектировщиков в целом. Одновременно обеспечивается наблюдение за состоянием системы и ее характеристиками качества, что позволяет устанавливать частные компромиссы с используемыми ресурсами. ***Объектами контроля при этом являются***:

— технические характеристики реализованных компонентов проекта, показатели качества процессов и результатов выполнения отдельных работ;

— затраты ресурсов на выполнение частных работ и реализацию компонентов проекта (трудоемкость, стоимость, время, материальные ресурсы);

— графики работ, степень их выполнения, наличие и причины отклонений реализации частных работ от планов, угроза нарушения сроков контракта.

Для получения, накопления и применения достоверных данных об объектах управления и альтернативах ***необходима информационная система обеспечения проекта***. Такая система представляет собой комплекс формальных и неформальных каналов обмена информацией между участниками проекта, ее накопления и обработки. Следует учитывать, что любая групповая деятельность связана со сложным комплексом неформальных отношений между исполнителями. Степень формализации может варьироваться от утверждаемых руководителями планов и подробных технических заданий до личных бесед между разработчиками. ***Регулярный обмен информацией позволяет осуществлять***:

— сбор исходных данных о состоянии, достигнутом качестве компонентов проекта и использованных ресурсах;

— диспетчерское управление ресурсами и частными исполнителями работ;

— сравнение текущих результатов частных работ с техническими заданиями, спецификациями и планом;

— корректировку технических результатов работ, сроков и используемых ресурсов в соответствии с изменением требований в процессе развития проекта.

Таким образом, ***целевое системное управление проектами*** позволяет планировать, контролировать и анализировать информацию о состоянии и тенденциях изменения объекта разработки, его качестве и затраченных ресурсах. При этом непрерывно должны ***сохраняться основные цели проекта и главные пути ее достижения***. Это позволяет рассматривать альтернативы технических решений и предотвращает от сосредоточения внимания на частных задачах или вариантах решений, которые кажутся полезными и интересными, но мало отражаются на достижении главной цели проекта ПС.

4.2. Процессы системного проектирования программных средств

Системное проектирование в программной инженерии охватывает период жизненного цикла сложных комплексов программ, начиная от формулирования первичного замысла на создание или модернизацию системы и до начала детального проектирования и разработки ПС. Результатом этого периода работ должно быть согласованное и формализованное разработчиком и заказчиком представление о целях, назначении, функциональных задачах и качестве будущего программного продукта, способного удовлетворить надежды и запросы пользователей. В системном проекте должны быть обобщены и отражены следующие основные ***результаты выполненных системных исследований и разработок*** (рис. 4.1):

— обобщенный анализ проведенного обследования объекта информатизации, функций существующей информационной системы, качества ее основных программных компонентов и базы данных;

— совокупность предварительных исходных требований к функциям и характеристикам комплекса программ;

— оценки имеющихся и потенциально доступных ресурсов (финансовых, вычислительных средств, специалистов) для обеспечения всего жизненного цикла и требуемого качества проекта комплекса программ;

Рис. 4.1

— результаты предварительного анализа возможной архитектуры комплекса программ на основе моделей и прототипов аналогичных систем, позволяющие наметить планы разработки и всего жизненного цикла проекта ПС;

— цели, задачи и функции предполагаемой новой или модернизированной системы, обобщенные в ***концепции*** создания соответствующего программного средства;

— проекты планов жизненного цикла, гарантирования требуемого качества ПС, защиты и обеспечения безопасности его функционирования;

— результаты технико-экономического обоснования целесообразности и основных направлений продолжения проектирования и всего ЖЦ ПС;

— результаты анализа существующей и возможной инструментальной среды разработки, а также системы обеспечения качества, перспективы их развития и совершенствования;

— предварительный план организации работ, требования к составу и квалификации специалистов для выполнения проекта и всего жизненного цикла ПС;

— проект формализованного *технического задания и спецификации требований к ПС*, **а также предложения по его финансированию и обеспечению ресурсами;**

— ***системный проект***, обобщающий проведенные исследования и разработки, позволяющий заключить контракт между разработчиком и заказчиком на финансирование и продолжение проектирования и/или на весь жизненный цикл ПС.

Системное проектирование сложных ПС начинается с ***обследования объекта информатизации***, системного анализа предметной области и выявления потребности в создании или модернизации комплекса программ с определенными функциями и качеством. При создании сложных ПС важно учитывать, что только заказчик и потенциальный пользователь системы вправе и способен корректно формулировать требования и впоследствии судить, насколько успешно проведена разработка соответствующего ПС. Аналитики-консультанты совместно с потенциальными разработчиками и заказчиком или пользователями должны проводить анализ прикладной области и объекта информатизации, разрабатывать стратегию разработки и технико-экономическое обоснование реализуемости выдвигаемых требований.

Разработка исходных требований для технического задания на проект ПС начинается с анализа результатов обследования объекта информатизации и оценки доступных ресурсов для реализации проекта. Эта

деятельность требует специальной организации специалистов наивысшей квалификации и тесной совместной работы представителей заказчика и разработчика. Они должны подготовить исходные данные и документы, в которых содержатся предварительные требования и пожелания к функциональным и конструктивным характеристикам качества программного комплекса. Далее ими должна проводиться сложная работа по предварительному упорядочению, селекции, обобщению и ***ранжированию приоритетов требований*** для их реализации в проекте. Наличие обычно ряда неформализованных, неструктурированных и противоречивых содержательных требований заказчика и разработчика требует их совместной обработки, согласования и корректировки.

Функциональные требования заказчика к процессам и результатам обработки информации необходимо скоординировать с конструктивными требованиями и возможностями их эффективной реализации разработчиками в ***спецификациях требований к комплексу программ*** и его программным и информационным компонентам. Должна быть предусмотрена корректировка, конкретизация и развитие совокупности предварительных требований в процессе системного проектирования и в дальнейшем по мере реализации проекта при тесном взаимодействии заказчика и разработчика. Для крупных проектов ПС целесообразно использовать специальный инструментарий и хранилище решений в процессе отработки требований, которые следует учесть в системном проекте и техническом задании, а также применять для контроля их реализации.

Предварительный анализ и моделирование процессов обработки данных при системном проектировании должны проходить этапы от простого установления базовых отношений между понятиями, через определение интерфейсов доступа и атрибутов, к проекту модели состояний и взаимодействий между реальными объектами и процессами ПС. Эти модели должны служить базой при разработке схем потоков управления и данных, описывающих процессы их обработки, а впоследствии интегрироваться с отработанными моделями процессов для комплексного исследования функционирования прототипов — пилотных проектов ПС в целом. Итеративный характер построения формализованного описания проекта системы предопределен изначально не только потому, что не удается сразу получить непротиворечивое и полное описание из-за неясностей в

исходном описании, но и потому, что сложную систему можно описывать, только начиная с основной части ее предметной области, которая затем постепенно расширяется и детализируется.

Моделирование процессов обработки данных при системном проектировании преследует ***две основные цели***:

— моделирование проблемно-ориентированных процессов и конкретных функциональных задач с целью исследования принципов, методов и характеристик обработки информации и принятия решений для последующего их использования в проектах;

— моделирование архитектуры объектов и процессов, а также их взаимодействия, предполагаемых для применения в конкретном проекте, без акцента на особенности функциональных характеристик компонентов.

При построении формализованного описания системы, выполняемом ее разработчиком, принципиальными являются ***два организационных момента***: специалисты — заказчики или пользователи создаваемой системы должны активно участвовать в процессе анализа и реализации ее описания; каждый шаг описания должен обязательно документироваться. Наглядными и удобными в работе являются ***графические представления описаний проектных решений***, которые позволяют создавать прототипы ПС. Они обеспечивают эффективную визуализацию и обратную связь между разработчиком и потенциальным пользователем с целью оценки реализации требований, корректировки функций и качества компонентов, а также форм пользовательского интерфейса. Для этого разработана целая гамма методологий для моделирования, структурного анализа и визуального проектирования. Современные инструментальные CASE-средства обеспечивают широкие возможности выбора процессов моделирования, автоматизированного анализа системных предложений и выработки первичных требований к предполагаемому проекту ПС. Схемы потоков данных, потоков управления, сущность-связь и другие — составляют комплекс удобных и гибких графических методов и средств описания систем, ***облегчающих взаимопонимание между разработчиками и заказчиками*** на разных уровнях детализации функций, качества и архитектуры ПС.

Концепция создаваемого комплекса программ (см. рис. 4.1) на естественном языке данной предметной области должна включать предварительные требования к ПС, основные понятия и термины. Она является

первым исходным документом, согласуемым с заказчиком для создания комплекса программ. На основе этого описания формируется предварительное техническое задание на систему и ее основные компоненты. При использовании формализованных методов разработки программных средств текстуальное описание системы подлежит переводу на соответствующий, возможно, графический язык. Наряду с разработчиками специалисты-заказчики или пользователи создаваемой концепции ПС должны активно участвовать в процессе анализа и реализации ее описания.

Одним из наиболее эффективных направлений сокращения затрат и повышения качества комплексов программ является активное использование методического, технологического, алгоритмического и программного задела из предшествующих проектов, которое может быть названо ***прототипированием*** в широком смысле слова. Математические модели и прототипы различных компонентов и функций систем обеспечивают возможность применять готовые апробированные решения, а также выделять и исследовать принципиально новые методы и процессы для реализации их в ПС. Прототипирование позволяет наглядно представить заказчику и пользователю функции системы, виды и динамику применения экранов, меню, отчетов и форм запросов, а также откорректировать их для развития ПС на всех этапах ЖЦ. Методами математического моделирования должны создаваться варианты, фрагменты и компоненты прототипа ПС и выделяться возможные методы реализации предполагаемых функций и обеспечения их качества. Для этого следует анализировать и выбирать прототипы комплексов программ, характеристики которых наиболее близки к создаваемой версии ПС и которые позволили бы получить в результате объекты с необходимыми характеристиками. На их основе возможно прогнозировать процессы разработки и достигаемые показатели качества вновь создаваемого ПС. Этим же целям способствует предварительное распределение ресурсов, доступных для создания проекта.

Стратегическое планирование проекта должно содержать долгосрочные цели развития ЖЦ ПС определенного функционального назначения. Планы должны отражать предварительные проекты всего будущего жизненного цикла ПС, обеспечения их качества, защиты и безопасности функционирования, верификации и тестирования, управления конфигурацией и сопровождения. На базе требований к ПС и первичных планов

появляется возможность оценить объем подлежащих разработке компонентов программ и баз данных, а также некоторые дополнительные характеристики возможного объекта и среды разработки. На этом этапе инструментальные средства должны обеспечить наглядное представление каждого плана, оценку возможной трудоемкости и длительности разработки, необходимого числа специалистов и других ресурсов для их реализации. По этим данным руководителем разработки и заказчиком принимается решение о целесообразности продолжения проектирования и осуществляется стратегическое планирование проекта, которое формализуется в системном проекте и в техническом задании на ПС.

Такое постепенное повышение достоверности прогнозов приводит к целесообразности оценки достигаемых значений качества по этапам и к возможности разработки укрупненного, поэтапного плана выполнения всего комплекса работ в ЖЦ ПС. Эти данные позволяют принимать решения по корректировке требований к ПС, по изменению среды разработки или состава коллектива специалистов. Таким образом, последовательное прогнозирование, планирование и системное управление проектом ***обеспечивают рациональное использование ресурсов*** в процессе создания сложных ПС гарантированного качества.

Прогнозы и анализ вариантов технологических процессов проектирования ПС, их технико-экономических показателей и характеристик объекта разработки являются основой для выбора, предварительного планирования и последующего системного анализа всего жизненного цикла ПС. Достоверность планов и прогнозов определяется точностью сведений об объекте разработки, характеристиках технологической среды и прототипов, принятых за основу при планировании. Таким образом, ***производится технико-экономическое обоснование проекта*** (см. лекцию 5), определяются приближенные значения трудоемкости и длительности всей разработки ПС, а также число необходимых специалистов, что позволяет оценить предварительный укрупненный план создания ПС в заданных условиях, ресурсах и сроках. Вследствие творческого характера большинства работ на этом этапе невозможно составить жесткий план их выполнения. Помочь может типовой перечень частных работ, представленный в стандартах ЖЦ ПС, и ориентировочный график, иллюстрирующий их взаимосвязь.

Проведенные таким образом оценки проекта ПС позволяют осуществить ***предварительный выбор основных методов и инструментальных средств*** для проведения последующего детального и рабочего проектирования и поддержки всего ЖЦ ПС. Кроме того, должна подготавливаться адаптация средств автоматизации применительно к особенностям объекта и среды проектирования. Разрабатываются проекты руководств для специалистов, выделяемых на данный проект, и осуществляется их обучение. Интегрированные инструментальные средства служат для формализации знаний заказчика на этапе проведения обследования, анализа и подготовки технического задания, а также для проектирования концептуальной и логической структуры комплекса программ и базы данных. При этом должно активно использоваться ***моделирование и тестирование корректности системных решений***. Благодаря высокому качеству проработки и документирования системного проекта создается основа для снижения трудоемкости тестирования, испытаний, а также сопровождения и модификации ПС.

В процессе системного проектирования должны предварительно определяться состав и структура основных технологических и эксплуатационных документов для поддержки всего ЖЦ ПС. Эти документы должны обеспечивать реализацию процессов жизненного цикла ПС, планирования и управления, регистрировать выполнение требуемых действий, формализовать систему качества. При этом следует подготовить первоначальные ***требования к документации и обеспечить их реализацию***, которая должна быть однозначной — написана в стандартизированных терминах, которые допускают только единственную интерпретацию, уточняемую, если необходимо, соответствующими комментариями.

Системный проект программного средства новой или модернизированной системы должен содержать достаточно полные требования к функциям и характеристикам качества комплекса программ, описание и графическое представление его архитектуры, базы данных и взаимодействия компонентов, предполагаемую модель жизненного цикла, предварительные планы последующих этапов и работ. Кроме того, в него должны входить ***проекты технического задания и контракта*** на детальное проектирование и весь жизненный цикл ПС. Если заказчик удовлетворен результатами системного проектирования, то возможно оформление ***акта***

завершения работ и утверждение системного проекта комплекса программ с требуемыми характеристиками качества новой или модернизированной системы, а также ***контракта (договора)*** на детальное проектирование или на весь жизненный цикл ПС.

4.3. Структурное проектирование сложных программных средств

При разработке структуры программного средства в процессе системного проектирования, прежде всего, ***необходимо сформулировать критерии*** ее формирования. Важнейшими критериями могут быть: модифицируемость, отлаживаемость и удобство управления разработкой ПС, обеспечение возможности контролируемого изменения конфигурации, состава и функций компонентов с сохранением целостности структурного построения базовых версий ПС. В зависимости от особенностей проблемной области критериями при выборе архитектуры ПС могут также применяться: эффективное использование памяти или производительности реализующей ЭВМ, трудоемкость или длительность разработки, надежность, безопасность и защищенность. В соответствии с целями и задачами проектирования на стадии системного анализа и формирования структуры ПС необходимо ранжировать и выделить доминирующие критерии.

Гибкость модификации и расширяемость комплексов программ при их совершенствовании обеспечиваются рядом принципов и правил структурного построения ПС и их компонентов, а также взаимодействия между ними. Эти правила направлены на стандартизацию и унификацию структуры и взаимодействия компонентов ПС разного ранга и назначения в пределах проблемной области. Некоторая часть принципов и правил имеет достаточно общий характер и может применяться практически всегда. Другая часть отражает проблемную и/или машинную ориентированность класса ПС и подлежит отработке при системном проектировании для эффективного применения в соответствующей области. ***Основные принципы и правила структурирования ПС и БД*** можно объединить в группы, которые отражают:

— стандартизированную структуру целостного построения ПС и/или БД определенного класса;

— унифицированные правила структурного построения функциональных программных компонентов и модулей;

— стандартизированную структуру базы данных, обрабатываемых программами;

— унифицированные правила структурного построения информационных модулей, заполняющих базу данных;

— унифицированные правила организации и структурного построения межмодульных интерфейсов программных компонентов;

— унифицированные правила внешнего интерфейса и взаимодействия компонентов ПС и БД с внешней средой, с операционной системой и другими типовыми средствами организации вычислительного процесса, защиты и контроля системы.

Методология структурного анализа и предварительного структурного проектирования ПС начинается с общего обзора функций системы. Далее функции должны детализироваться ***сверху вниз*** в виде иерархической структуры таким образом, чтобы процедуры сбора, хранения и переработки информации, рассматриваемые сначала как нечто единое целое, расчленялись на отдельные элементы данных, компонентов и действия, совершаемые над этими данными. Структурный анализ, исходя из функционального описания системы в целом, позволяет разделить ее на функциональные части, выделить функциональные описания отдельных частей, исследовать в них информационные потоки и формализовать структуры данных.

Разделение общей задачи системы и программного средства на компоненты — это использование принципа здравого смысла, который должен быть применен в системной разработке крупного ПС для преодоления свойственной ему сложности. Очень часто многие решения прочно взаимосвязаны и взаимозависимы. Когда различные проектные решения прочно взаимосвязаны, то бывает полезно, чтобы всеми ими сразу занимались в одно и то же время одни и те же люди, но на практике это обычно невозможно. Единственный способ справиться со сложностью проекта — разделить задачи. Прежде всего, необходимо пытаться изолировать функции, которые менее всего связаны с другими. Затем рассматривать раздельно функции, учитывая ***имеющие отношения между собой*** и детали связанных проблем.

Спецификация требований к предварительной архитектуре комплекса программ формируется в процессе детализации и уточнения спецификации требований к характеристикам ПС в результате проверки последних на непротиворечивость и полноту. Задача спецификации требований архитектуры состоит в том, чтобы представить достаточно ясное и удобное общее описание внешнего поведения системы и свойств, а также ее внутренней структуры и механизмов функционирования. В общем случае ***формализованные методы, применяемые при специфицировании системной архитектуры*** ПС, должны обеспечивать:

— эффективные и удобные средства описания свойств и особенностей структуры создаваемой системы — нотации описания;

— методику последовательного, итерационного уточнения спецификаций требований и проектных спецификаций компонентов, позволяющую проверять их корректность;

— средства и методику для возможности прослеживания реализации требований и тестирования, позволяющего устанавливать соответствие свойств реализуемого ПС и компонентов его спецификациям.

Описания проектных решений должны содержать первичные спецификации крупных функциональных компонентов ПС, подлежащих разработке в детальном проекте создаваемой системы, и спецификаций используемых готовых компонентов, состав которых определяется при декомпозиции общей структуры системы. В ***требованиях спецификации к системной архитектуре*** комплекса программ должно обеспечиваться:

— соответствие функций и структуры ПС аппаратной и операционной среде, их ресурсам и интерфейсам;

— совместимость ПС с другими системами по источникам и потребителям информации;

— соответствие стандартам структурного построения и интерфейсов комплекса программ, функциональных компонентов и модулей;

— предварительная организация информационного обеспечения и структура базы данных;

— состав, структура и способы обмена данными между функциональными компонентами и внешней средой ПС;

— временной регламент и предварительные характеристики процессов реализации функций, интенсивность и объемы потоков информации базы данных;

— контроль, хранение, обновление, защита и восстановление программ и данных;

— стандартизированные, предварительные требования к составу и содержанию технологической документации.

Таким образом, для обеспечения эффективного управления разработкой программ необходимо при системном проектировании стандартизировать и соблюдать ***ряд принципов архитектурного построения ПС***. Эти принципы могут иметь особенности для проектов ПС в различных проблемно-ориентированных областях. Однако их стандартизация обеспечивает значительный эффект в снижении трудоемкости и длительности последующей детальной разработки программного продукта и версий. Частичная потеря гибкости архитектуры ПС, некоторое возрастание ресурсов, необходимых для реализации этих принципов, обычно полностью компенсируются повышением управляемости процесса разработки, а также качества ПС.

Структурное проектирование программных средств ***основано на модульном принципе***. Многоуровневое, иерархическое построение сложных программных комплексов позволяет ограничивать и локализовать на каждом из уровней соответствующие ему компоненты. Нижнему иерархическому уровню представления программ соответствуют программные и информационные модули (модули данных). Эти компоненты объединяются в группы программ определенного функционального назначения с автономной целевой задачей. Несколько групп функциональных программ образуют комплекс программ. В особо сложных случаях возможно создание программного средства из нескольких взаимодействующих комплексов. Всем иерархическим системам (в частности, ПС) ***присущ ряд общих свойств***, важнейшими из которых являются:

— вертикальная соподчиненность, заключающаяся в последовательном упорядоченном расположении взаимодействующих компонентов, составляющих ПС;

— право вмешательства и приоритетного воздействия сверху вниз на компоненты нижних уровней;

— взаимозависимость действий компонентов верхних уровней от реакций на воздействия и от функционирования компонентов нижних уровней, информация о которых передается верхним уровням.

В результате в иерархических структурах ПС образуются ***два потока взаимодействий*** между компонентами разных уровней: ***сверху вниз*** — координирующие и управляющие воздействия верхних уровней и ***снизу вверх*** — информация о состоянии и реализации предписанных сверху функций компонентами нижних уровней. Координируемые компоненты обычно имеют некоторую автономность поведения и подготовки локальных решений. Степень автономности компонентов и интенсивность координирующих воздействий устанавливаются в результате ***компромисса при выделении числа и размеров иерархических уровней***. Взаимодействие компонентов в пределах уровня целесообразно максимально ограничивать, что позволяет упростить общее координирование компонентов и проводить его только по вертикали.

Менее наглядными являются ***иерархия данных, обрабатываемых ПС***, и их взаимодействие с программными компонентами. Функциональная иерархия данных отражается ***расстоянием*** между расчетом или изменением переменной и ее использованием, или условной длительностью хранения неизменяемых значений переменной. Взаимодействие двух программных модулей может осуществляться так, что некоторые переменные используются только этими модулями. Такие обменные переменные имеют более широкую область применения и должны храниться все время, пока не будут вызваны взаимодействующие модули. Ряд переменных и массивов используется многими модулями и группами программ в комплексе — это глобальные переменные. Они характеризуются наиболее широким использованием и соответствуют высшему иерархическому уровню среди данных.

Анализ концепции, требований технического задания и технико-экономических оценок должен позволять выполнить ***предварительное структурное проектирование ПС и оценку вычислительных ресурсов,*** необходимых для решения основных функциональных задач. Повышению эффективности структурирования могут значительно способствовать заимствование из предыдущих проектов спецификаций прототипов, версий и отдельных компонентов ПС. Для обеспечения системного проектирования на этом этапе большое значение имеют графические методы визуализации технических решений и логического контроля проекта.

Характеристики внешней среды применения ПС и особенности реализующей ЭВМ в значительной степени определяют архитектуру и струк-

туру применяемой операционной системы, средств контроля и организации вычислительного процесса. При разработке версий ПС для некоторой прикладной области целесообразно выбирать и унифицировать внешний интерфейс и операционную систему. Это обеспечивает многократное использование одних и тех же организующих программ, дисциплинирует структурное построение версий ПС и способствует унификации межмодульного интерфейса. Переход на новую операционную систему, так же как и переход на реализующую ЭВМ другого типа, может привести к необходимости некоторого изменения структурного построения ПС и базы данных. Это способно повлиять на возможность использования готовых компонентов, а следовательно, и на эффективность всей разработки.

Учет возможности изменений — это принцип, который более всего отличает программное средство от большинства других типов промышленных продуктов. Во многих случаях структура ПС разрабатывается, когда требования заказчика к нему осознаны не полностью. Позже, при детализации и после поставки, ПС должно развиваться на основе откликов заказчика и пользователей, поскольку обнаруживаются новые требования, а старые уточняются. В дополнение к этому программный продукт часто встраивается в среду, которая воздействует на него, и это воздействие генерирует новые требования, которые не были изначально известны. Таким образом, возможности изменений — это принцип, который следует использовать при системном проектировании структуры для достижения способности к ее развитию — ***эволюционности структуры ПС***.

Повторная применимость — является еще одним качеством структуры программного продукта, на которое заметно воздействует принцип возможности изменений. Компонент является многократно применимым, если он может быть непосредственно использован для производства нового продукта или версии. На практике компонент может претерпеть некоторые изменения прежде, чем будет использован повторно. Отсюда многократное использование можно рассматривать как эволюционность системной структуры ПС на уровне компонентов. Если можно предвидеть изменения контекста, в который будет встроен программный компонент, то и компонент можно спроектировать таким образом, что изменения будут им учтены.

4.4. Проектирование программных модулей и компонентов

Сложная система обычно может быть разделена на более простые части — ***модули***. Модульность является важным качеством инженерных процессов и продуктов. Большинство промышленных процессов являются модульными и составлены из комплексов работ, которые комбинируются простыми способами (последовательными или перекрывающимися) для достижения требуемого результата. Главное преимущество модульности заключается в том, что она позволяет применять принцип разделения задач на ***двух этапах***:

— при работе с элементами каждого модуля отдельно (игнорируя элементы других модулей);

— при работе с общими характеристиками групп модулей и отношениями между ними с целью объединить их в конкретный, более крупный и сложный компонент.

Если данные этапы выполняются в последовательности, предусматривающей сначала концентрацию процессов на модулях, а затем — их объединение, то система проектируется ***снизу вверх***. Если сначала систему разбивают на модули, а потом работают над их индивидуальным проектированием, то это — проектирование ***сверху вниз***.

При структурном построении комплексов программ важное значение имеет размер и сложность компонентов для каждого уровня иерархии и соответственно число иерархических уровней для крупных ПС. По принципам построения, языку описания, размеру и другим характеристикам компонентов в структуре ПС ***можно выделить иерархические уровни***:

— программных модулей, оформляемых как законченные компоненты текста программ;

— функциональных групп (компонентов) или пакетов программ;

— комплексов программ, оформляемых как законченные программные продукты определенного целевого назначения.

С повышением иерархического уровня увеличивается размер текста программ, реализующих компоненты этого уровня и количество обрабатываемых переменных. Одновременно совокупности команд все более специализируется и снижается возможность повторного применения компонентов в различных комбинациях для решения аналогичных задач.

Программные модули решают относительно небольшие функциональные задачи, и каждый реализуется 10—100 операторами языка программирования. Каждый модуль может использовать на входе около десятка типов переменных. Если для решения небольшой функциональной задачи требуется более 100 операторов, то обычно целесообразно проводить декомпозицию задачи на несколько более простых модулей.

Функциональные группы программ (компоненты) формируются на базе нескольких или десятков модулей и решают достаточно сложные автономные задачи. На их реализацию целесообразно использовать до десятка тысяч строк текста программы. Соответственно возрастают число используемых типов переменных и разнообразие выходных данных. При этом быстро растет число типов переменных, обрабатываемых модулями и локализующихся в пределах одного или нескольких модулей.

Комплексы программ — программные продукты создаются для решения сложных задач управления и обработки информации. В комплексы объединяются несколько или десятки функциональных групп программ для решения общей целевой задачи системы. Размеры ПС зачастую исчисляются сотнями модулей, десятками и сотнями тысяч операторов. Встречаются ПС, содержащие до двух-трех десятков структурных иерархических уровней, построенных из модулей.

Проектирование модулей включает в себя разработку локальных функций и подробных описаний алгоритмов обработки данных; межмодульных интерфейсов; внутренних структур данных; структурных схем передач управления; средств управления в исключительных ситуациях. С их помощью определяются функции: порядок следования отдельных шагов обработки, ситуации и типы данных, вызывающие изменения процесса обработки, а также повторно используемые функции программы. Программные модули для их многократного использования должны базироваться на унифицированных правилах структурного построения, оформления спецификаций требований и описаний текстов программ и комментариев. Кроме того, целесообразно для каждого проекта директивно ограничивать размеры модулей по числу строк текста с учетом языка программирования, например, 30-ю или 50-ю операторами (см. лекцию 13).

Основная цель такой унификации — облегчить разработку модулей, обеспечение их качества и тестирования, а также упростить управление их функциями и характеристиками. Эти правила в значительной степени стан-

дартизированы в современных языках программирования. Однако при их применении целесообразно выделять типовые ассоциации операторов и ограничения их использования, а также вводить правила описания текстов программ, комментариев, данных и заголовков модулей, ограничения их размеров и сложности. Эти правила наиболее полно должны соблюдаться при разработке основной массы функциональных программ, подлежащих повторному использованию в модифицируемых версиях программных продуктов. Компоненты организации вычислительного процесса, контроля функционирования, ввода-вывода и некоторые другие могут иметь отличия в правилах структурного построения модулей.

Для обеспечения управляемой модификации и развития конфигураций версий программного продукта важно ***стандартизировать структуру базы данных***, в которой накапливается и хранится исходная, промежуточная и результирующая информация в процессе функционирования ПС. Основными компонентами этой структуры являются информационные модули или пакеты данных. В них также целесообразно использовать типовые структуры, ориентированные на эффективную обработку данных в конкретной проблемной области. Объединение информационных модулей позволяет создавать более сложные структуры данных определенного целевого назначения. Иерархия связей между этими компонентами в некоторой степени соответствует процессу обработки и потокам данных и относительно слабо связана со структурой программных компонентов в ПС. Структуры информационных модулей целесообразно координировать между собой с учетом цели и места их использования программными компонентами.

Особое значение для качества модулей и компонентов крупных ПС имеет ***стандартизация структуры межмодульных интерфейсов по передачам управления и по информации***. Эти правила формируются на базе описаний языков программирования или оформляются на основе правил структурного построения программ и базы данных конкретных проектов ПС. В последнем случае соглашения о связях конкретизируются в макрокомандах межмодульного взаимодействия. ***Структурное проектирование*** сложных комплексов программ активно развивается на основе концепции и стандартов открытых систем. Применение стандартов открытых систем следует начинать при создании ***архитектуры исходных модулей*** мобильных ПС и БД, а далее неукоснительно использоваться при всех процессах ЖЦ. Во всех случаях создание архитектуры модулей и компо-

нентов современных сложных систем целесообразно вести с использованием профилей международных стандартов, значительная часть которых обеспечивает мобильность и возможность повторного использования готовых программных средств и баз данных (см. лекцию 3).

Основными целями создания и применения концепции, методов и стандартов открытых систем является повышение общей экономической эффективности разработки и функционирования систем, а также логической и технической совместимости их компонентов, обеспечение мобильности и повторного применения готовых программ и данных. Они реализуют спецификации на интерфейсы, процессы и форматы данных, достаточные для того, чтобы обеспечить:

— возможность расширения ПС, а также переноса (мобильность) программных компонентов и систем, разработанных должным образом, с минимальными изменениями на широкий диапазон аппаратных и операционных платформ;

— совместную работу с другими программными продуктами и системами на локальных и удаленных платформах;

— взаимодействие с пользователями в стиле, облегчающем последним переход от системы к системе (мобильность пользователей).

Для достижения этих целей развиваются и применяются различные проблемно-ориентированные технологии и комплексы средств автоматизации ЖЦ мобильных программ и баз данных, основанные на повторном использовании готовых апробированных модулей, программных компонентов и данных, их эффективном переносе на различные аппаратные и операционные платформы и согласованном взаимодействии в распределенных информационных системах. Профессионалы в области открытых систем акцентируют усилия на поиске и создании гибкой, способной к наращиванию среды, что ***базируется на трех направлениях стандартизации*** в области систем (см. лекцию 2):

— аппаратных и операционных платформ;

— методов и технологии обеспечения жизненного цикла прикладных программных средств и баз данных;

— интерфейсов компонентов и модулей между собой, с операционной и внешней средой.

Методы открытых систем обеспечивают эффективные по трудоемкости и качеству структурное проектирование, расширение и перенос гото-

вых программных средств обработки информации и баз данных на различные аппаратные и операционные платформы. Эти методы можно разделить на ***три части***:

— общая концепция и методы непосредственного обеспечения мобильности компонентов программных средств и баз данных в процессе разработки систем за счет унификации интерфейсов с операционной и аппаратной средой;

— методы, поддерживающие мобильность компонентов и комплексов программ и данных в распределенных системах и совместимость их взаимодействия с внешней средой;

— методы создания текстов программных средств, баз данных и их компонентов на стандартизированных языках программирования высокого уровня, обеспечивающие потенциальную возможность их переноса на различные аппаратные платформы.

Первая группа методов создавалась с ограниченной и определенной целью локализовать и унифицировать интерфейсы программных компонентов между собой, с заранее выделенными операционными системами и с внешней средой. Интерфейсные стандарты являются базой для обеспечения структурного проектирования, свободного перемещения и расширения программных продуктов и компонентов в различные по архитектуре и функциям операционные окружения и аппаратные платформы. Эти методы позволяют разделить функциональную часть комплексов программ и их связи с организационным окружением, обеспечивая технологическую, архитектурную и языковую независимость функциональной части программ и данных. Тем самым они являются базой для реализации концепции открытости в системах и обеспечивают свободу разработчикам при выборе ***методов структурного проектирования*** и языков программирования для создания компонентов программ и описаний данных. Стандарты этой группы включают группу стандартов POSIX, в которой определены концепция и функции интерфейсов переносимых операционных систем, команды управления и сервисные программы, а также расширение для переносимых операционных систем.

Вторая группа методов имеет целью поддержать мобильность программных продуктов в открытых системах путем унификации их интерфейсов с внешней средой. Они обеспечивают совместимость обмена данными в различных файловых системах и базах данных, унификацию адми-

нистративного управления и взаимодействия с пользователями, а также методов обеспечения безопасности и защиты информации. Таким образом, создается стандартизированная по интерфейсам внешняя среда на различных гетерогенных аппаратных платформах, в которую могут эффективно погружаться различные программы, согласованные по интерфейсам с этими стандартами. Стандарты этой группы регламентируют функции и процессы, поддерживающие взаимодействие с внешней средой:

— концепции и архитектуру визуализации информации для взаимодействия с пользователями и ее представления в базовых системах графического отображения;

— архитектуру, интерфейсы, базовые процессы и языки управления файловыми системами и базами данных, обеспечивающими их совместимость и унификацию в сложных системах;

— административное управление локальными и распределенными компонентами систем в процессе решения функциональных задач обработки информации.

Третья группа методов создавалась в значительной степени независимо от первой и второй. Эти методы преследуют цель унифицировать тексты программных модулей, компонентов и описания данных, создаваемых для различных аппаратных платформ при любой их архитектуре, независимо от операционной и внешней среды. Первоначальное огромное число языков программирования (свыше 200), каждый из которых имел несколько диалектов, в результате сократилось до 6—10 массовых языков, ограниченных стандартами, не допускающими диалектов. Создание современных модулей и компонентов ПС и БД поддерживается методами и инструментальными средствами технологий, методами тестирования и аттестации программ и их интерфейсов, а также стандартизированным составом и содержанием документации на программы и базы данных. Эти методы и средства обеспечивают необходимое качество модулей и компонентов сложных ПС и БД. В результате обеспечена мобильность функциональной части текстов программ, однако они могут требовать доработок интерфейсов для сопряжения с новой средой аппаратного и операционного окружения систем.

ЛЕКЦИЯ 5

ТЕХНИКО-ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ ПРОЕКТОВ ПРОГРАММНЫХ СРЕДСТВ

5.1. Цели и процессы технико-экономического обоснования проектов программных средств

Выбор и формирование ***требований к функциональной пригодности ПС*** — наиболее ответственная, стратегическая задача начальных этапов технико-экономического обоснования проекта программного продукта, системного проектирования и всего последующего развития его жизненного цикла. Жизненный цикл ПС можно разделить на ***две части***, существенно различающиеся ***экономическими особенностями*** процессов и ресурсов, характеристиками и влияющими на них факторами. ***В первой части ЖЦ ПС*** производятся системный анализ, проектирование, разработка, тестирование и испытания базовой версии программного продукта. Номенклатура работ, их трудоемкость, длительность и другие экономические характеристики на этих этапах ЖЦ существенно зависят от характеристик объекта, технологии и инструментальной среды разработки. Особенно важно учитывать возможное возрастание суммарных затрат при завышении требований к качеству программного продукта. Как и для других видов промышленной продукции, улучшение качества комплексов программ обычно достигается не пропорциональным, а более значительным возрастанием требуемых для этого ресурсов. Сокращение этой потребности в ресурсах часто возможно только за счет принципиального изменения и совершенствования технологии проектирования и разработки.

Ориентирами для оценивания необходимых ресурсов трудоемкости, длительности и числа специалистов по крупным этапам работ при создании сложных ПС высокого качества могут служить данные, приведенные ниже в п. 5.2—5.4. Максимум трудоемкости и числа специалистов приходится на этапы программирования и тестирования компонентов, когда привлекается основная масса программистов-кодировщиков для разработки компонентов ПС. При активном использовании и совершенствовании технологий системного анализа и проектирования происходит перераспределение всех видов затрат в сторону увеличения трудоемкости начальных этапов разработки. Это дает значительное снижение использования совокупных ресурсов для всего проекта. Менее изучены распределения необходимых ресурсов по этапам работ, с учетом реализации требуемых конкретных характеристик качества ПС. Опубликованные данные и зависимости для различных классов ПС позволяют прогнозировать совокупные затраты и другие основные технико-экономические показатели (ТЭП), планы и графики работ вновь создаваемых проектов программных продуктов.

Вторая часть ЖЦ, отражающая эксплуатацию, сопровождение, модификацию, управление конфигурацией и перенос ПС на иные платформы, в меньшей степени зависит по величине требуемых ресурсов от функциональных характеристик объекта и среды разработки. Номенклатура работ на этих этапах более или менее определенная, но их трудоемкость и длительность могут сильно варьироваться, в зависимости от массовости и других внешних факторов распространения и применения конкретных версий программного продукта. Успех ПС у пользователей и на рынке, а также будущий процесс развития версий трудно предсказать, и он не связан напрямую с экономическими параметрами процессов разработки ПС. Определяющими становятся потребительские характеристики продукта, а их экономические особенности с позиции разработчиков и вложенные ресурсы на очередную версию отходят на второй план (см. первую часть ЖЦ).

Вследствие этого в широких пределах могут изменяться трудоемкость и число специалистов, необходимое для поддержки этих этапов ЖЦ. Это затрудняет статистическое обобщение ТЭП различных проектов и прогнозирование на их основе аналогичных характеристик новой разра-

ботки. Поэтому планы на этих этапах имеют характер общих взаимосвязей содержания работ, которые требуют распределения во времени, индивидуально для каждого проекта. В результате прогнозирование и планирование трудоемкости и длительности этапов приходится производить итерационно на базе накопления опыта и анализа развития конкретных версий ПС, а также ***от их успеха на рынке***.

Для прогнозирования и планирования любых процессов или характеристик объектов (в частности ПС) используются исходные данные ***двух типов***:

— функции и номенклатура характеристик самого прогнозируемого объекта или процесса, для которого необходимо спланировать жизненный цикл;

— характеристики прототипов и пилотных проектов, в некоторой степени подобных планируемому объекту, о которых известны реализованные планы и необходимые экономические характеристики уже завершенных аналогичных процессов или объектов.

Совместная, корректная обработка исходных данных этих двух типов позволяет при проектировании оценивать и получать новые, прогнозируемые характеристики процессов, планов и экономических показателей создания ПС. Исходные данные ***первого типа*** отражают характеристики конкретного создаваемого объекта или процесса, доступные методы и инструментальные средства автоматизации труда при их создании. Эти данные последовательно детализируются и уточняются в процессе проектирования и дальнейшего ЖЦ ПС, что, в частности, позволяет уточнять выбор компонентов аналогичных объектов и их характеристик для исходных данных второго типа.

Второй тип исходных данных для обоснования и планирования разработки ПС составляют обобщенный опыт проектирования и экономические характеристики ***прототипов*** нового программного продукта. Для достоверного планирования необходимо накопление, обобщение и изучение конкретных данных о реализованных планах, затратах и использованных ресурсах завершенных разработок ПС в различных аспектах. Такие ТЭП и факторы, их определяющие, изучены в процессе обработки значительного статистического материала реальных отечественных и зарубежных проектов программных продуктов и использованы ниже в методиках прогнозирования (см. п. 5.2—5.4).

При технико-экономическом обосновании проекта ПС на любом уровне целесообразно применять методы и методики, адекватные целям и этапам его реализации. Следует согласовывать цели оценивания ТЭП с потребностями в информации, способствующей принятию решений для планирования затрат труда и других ресурсов. В общем случае необходимо достигать ***сбалансированного состава целей*** оценивания разных характеристик, которые бы давали примерно одинаковую абсолютную величину уровня неопределенности оценок для всех компонентов ПС. Кроме того, каждая оценка ТЭП должна сопровождаться указанием степени ее неопределенности. По мере разработки проекта их необходимо пересматривать и изменять, когда это становится выгодным.

Привлечение заказчика помогает менее болезненно решать проблемы управления масштабом проекта и реализуемыми функциями с учетом ограничений ресурсов. В зависимости от этапа разработки сложного комплекса программ и достоверности исходных данных о характеристиках и особенностях проекта ПС целесообразно выбирать и применять разные методики и сценарии технико-экономического обоснования проекта и прогнозирования ТЭП. С самого начала работы над проектом ПС важно вести постоянный учет данных о его ***действительной*** трудоемкости, стоимости и развитии затрат и ***сравнивать*** эти данные с реальными оценками характеристик проекта ***по следующим причинам***:

— несовершенство исходных данных при оценивании ТЭП (оценки размера, рейтинги влияния факторов) определяет важность для руководителя проекта пересматривать их оценки, учитывая новую информацию, чтобы обеспечить более реальную основу для дальнейшего управления проектом;

— вследствие несовершенства методов оценивания ПС следует сравнивать оценки с действительными значениями и использовать эти результаты для улучшения методов оценивания ТЭП;

— проекты ПС имеют тенденцию к изменению характеристик и экономических факторов и руководителю проекта необходимо идентифицировать эти изменения и выполнять реалистичное обновление оценок затрат.

При разработке ПС необходимо учитывать, что экономические, временные, вычислительные и другие ресурсы на разработку и весь ЖЦ программ всегда ограниченны и используемые затраты для улучшения каж-

дой характеристики должны учитывать эти ограничения. Для рационального распределения этих ресурсов необходимо знать, ***как отражается изменение затрат на улучшении каждой характеристики качества ПС***. Эта взаимосвязь затрат ресурсов и значений каждой характеристики зависит от назначения, а также от ряда свойств и других особенностей комплекса программ, что усложняет учет влияния таких связей. Тем не менее выявлены основные тенденции такого взаимодействия, которые могут служить ***ориентирами*** при выборе и установлении требований к определенным характеристикам качества в конкретных проектах ПС.

Основными ресурсами у разработчиков при создании сложных комплексов программ являются: допустимые ***трудозатраты (стоимость)*** на разработку ПС с требуемым качеством; ***время*** — длительность полного цикла создания программного продукта; необходимое и доступное ***число специалистов*** соответствующей квалификации. Потребность в этих ресурсах в наибольшей степени зависит от размера — масштаба и сложности разрабатываемого ПС. Уточнения размеров ПС и компонентов могут быть решены последовательно к концу детального проектирования, однако при этом сохраняется неопределенность оценки размера комплекса программ и его трудоемкости порядка 5—10%, связанная с тем, насколько хорошо программисты понимают спецификации, в соответствии с которыми они должны кодировать программу, при этом целесообразно учитывать:

— цели оценивания ТЭП должны быть согласованы с потребностями в информации, способствующей принятию решений на соответствующем этапе проекта ПС;

— достоверность оценок ТЭП должна быть сбалансирована для различных компонентов системы, и величина уровня неопределенности для каждого компонента должна быть примерно одинаковой, если в процессе принятия решения все компоненты имеют одинаковый вес;

— следует возвращаться к предшествующим целям оценивания ТЭП и изменять их, когда это необходимо для ответственных бюджетных решений, принимаемых на ранних этапах и влияющих на следующие этапы.

Достаточно трудно оценить объем трудозатрат, необходимых для выполнения задачи, без достоверной информации относительно ее ***размера***. Таким образом, измерение размера (сложности) предшествует оценке ТЭП,

а эта оценка, в свою очередь, предшествует составлению графика работ. Недостаточно достоверные оценки влекут проблемы взаимодействия разработчика с заказчиком и увеличивают степень риска проекта.

Исходные данные реальных завершенных разработок для оценивания ТЭП собираются, накапливаются и обрабатываются с начала 70-х годов в разных отечественных организациях и за рубежом. Они позволили получать и прогнозировать основные обобщенные ТЭП процессов разработки ПС. При этом обычно рассматривался полный технологический процесс разработки ПС от начала подготовки технического задания до завершения испытаний базовой версии программного продукта. Учитывались все категории специалистов, участвующих в создании программ и обеспечивающих разработку, а также все виды работ, связанные с созданием программного продукта на выделенном интервале времени. Теоретические работы и системный анализ до подготовки требований заказчика в значениях ТЭП не учитывались. В общем случае для оценки технико-экономических характеристик новых проектов ***необходимы исходные данные***:

— обобщенные характеристики использованных ресурсов и технико-экономические показатели завершенных разработок — прототипов ПС, а также оценки влияния на их характеристики различных факторов объекта и среды разработки;

— реализованные и обобщенные перечни выполненных работ и реальные графики проведенных ранее разработок различных классов ПС;

— цели и содержание частных работ в процессе создания сложных комплексов программ и требования к их выполнению для обеспечения необходимого качества ПС в целом;

— структура и содержание документов, являвшихся результатом выполнения частных работ.

В качестве базового варианта целесообразно принять статистические данные ТЭП, перечень работ и документов жизненного цикла создания наиболее ***сложного встроенного комплекса программ реального времени***. На основе этих исходных данных могут быть оценены ТЭП для полного цикла разработки ПС конкретного вида в более простых случаях путем исключения из базового варианта работ и документов, в которых отсутствует необходимость. По оставшимся работам могут быть оценены

ТЭП для анализируемых вариантов, и выбран из них предпочтительный. Для технико-экономического анализа процесса создания программ важнейшей составляющей являются ***совокупные трудовые затраты — трудоемкость*** на непосредственную разработку ПС.

Трудоемкость разработки программных средств наиболее сильно зависит от ***размера — масштаба комплекса программ, выраженного***: числом операторов, строк на языке программирования или функциональных точек (см. стандарт **ISO 14143:1-5:1998-2004**. — Измерение функционального размера). Реальное изменение создаваемых в настоящее время сложных ПС от 10^4 до 10^7 строк (LOC) определяет диапазон трудоемкости разработки таких программ от человеко-года до десятков тысяч человеко-лет. Подтверждена по большому числу проектов высокая статистическая корреляция между размером комплексов программ и трудоемкостью их разработки.

Эффективность затрат при повторном использовании компонентов (ПИК) и сборке ПС в зависимости от их доли зачастую оценивалась путем анализа ***эквивалентной производительности*** труда разработчиков и длительности создания ПС. В ряде случаев особое значение имеет не столько использование готовых программных компонентов, сколько ***перенос баз данных***. Информация о процессах, происходящих во внешней среде, может иметь большой объем и трудоемкость первичного накопления и актуализации, что определяет необходимость ее тщательного хранения. Практически ***всегда необходимо время и трудоемкость*** на:

— первичный системный анализ целесообразности применения ПИК;

— поиск, адаптацию и процессы использования готовых компонентов;

— оценку затрат с учетом стоимости приобретения и адаптации переносимых программ и баз данных;

— интегрирование в новой операционной или внешней среде;

— тестирование и испытания компонентов в комплексе с унаследованными программами.

Для планирования разработки сложных ПС важно знать и использовать экспериментальные статистические распределения основных ТЭП — трудоемкости, длительности и числа специалистов ***по этапам работ и по реальному времени реализации компонентов проектов***. Относительные значения распределения этих величин на интервале реализации крупных

проектов несколько различаются в зависимости от размера и типа комплекса программ, однако наибольший интерес представляют сложные встроенные ПС реального времени размером порядка 500 тысяч строк.

В совокупных затратах на создание полностью новых ПС доминирует трудоемкость непосредственной разработки программных компонентов. ***Распределение необходимой трудоемкости на этапы разработки программ*** сложного ПС реального времени представлено в таблице 5.1. Этап технологической подготовки разработки включен в техническое проектирование, а документирование объединено с комплексной отладкой. В распределении учтены подмножества работ, соответствующие разным категориям специалистов. Все специалисты были разделены на три категории: руководители разработки и системные аналитики; непосредственные разработчики программных компонентов и специалисты по комплексированию; вспомогательный персонал, обеспечивающий разработку и документирование программ. Первая и третья категории специалистов непосредственно не взаимодействуют с текстом программ при их отработке, однако их труд является неотъемлемой частью всего процесса разработки и в крупных проектах составляет около половины затрат на каждом этапе.

Таблица 5.1

Распределение затрат по этапам разработки программных средств реального времени

Этапы разработки	Трудоемкость, %	Длительность, %	Численность специалистов, % от средней
Предварительное проектирование	8	20	40
Детальное проектирование	14	20	70
Программирование	22	16	140
Автономная отладка компонентов	24	16	150
Интеграция и комплексная отладка	24	20	120
Испытания и документирование	8	8	100

Основное содержание, размер и требуемое качество создаваемых ПС практически всегда определяют затраты, связанные с их непосредствен-

ной разработкой. Влияние этой части затрат определяется наиболее сложным творческим процессом создания программ, который зависит от многих факторов. Некоторые из них могут изменять затраты даже в несколько раз, но в большинстве своем изменяют их на десятки процентов. Накопленный опыт создания ПС и обобщение проведенных исследований позволили выделить четыре основные ***группы факторов***, влияющих на оценки затрат при непосредственной разработке программ:

— факторы, отражающие особенности создаваемого комплекса программ, ***как объекта*** разработки, требования к его функциональным характеристикам и к качеству;

— факторы, определяющие ***организацию процесса*** разработки комплексов программ и его обеспечение квалифицированными специалистами;

— факторы, характеризующие ***технологическую среду*** и оснащенность инструментальными средствами автоматизации процесса разработки программ;

— факторы, отражающие оснащенность процесса создания ПС ***аппаратурными вычислительными средствами***, на которых реализуются комплексы программ и базируются инструментальные системы автоматизации разработки.

В представленных четырех группах распределены факторы, которые наиболее важны при анализе основных затрат на проекты ПС. В эти группы включены факторы, которые могут изменять оценку производительности труда при создании ПС не менее чем на 10% в ту или иную сторону. В то же время имеющийся опыт показывает, что ***отсутствуют отдельные факторы или методы***, способные изменять на порядок или более основные ТЭП процесса разработки программ. Большинство факторов изменяет экономические характеристики разработки программ на десятки процентов и не более чем в 1,5 раза. Для оценивания ТЭП ниже в п. 5.2—5.4 последовательно рассмотрены и рекомендуются ***три методики***:

— ***Методика 1*** — экспертного технико-экономического обоснования проектов программных средств при подготовке концепции и технического задания на новый комплекс программ на основе экспертных данных разработки одной строки текста программ-прототипов;

— ***Методика 2*** — оценка технико-экономических показателей проектов программных продуктов с учетом совокупности основных факторов

предварительной модели COCOMO II (см. Boehm B.W. et al. Software cost estimation with COCOMO II. Prentice Hall PTR. New Jersey. 2000);

— ***Методика 3*** — уточненная оценка технико-экономических показателей проектов программных продуктов с учетом полной совокупности факторов детальной модели COCOMO II.2000 (там же).

В качестве основных критериев выбора методик прогнозирования ТЭП разработки ПС целесообразно учитывать возможность их использования как на начальных, так и на более поздних этапах разработки. Для практического применения модели COCOMO II опубликован пакет прикладных программ и руководство по его применению. Оно иллюстрировано формами экранов и несколькими обширными практическими примерами применения для технико-экономического анализа конкретных проектов сложных комплексов программ.

Важнейшим фактором при технико-экономическом обосновании, определяющим создание программных средств, являются люди — ***специалисты***, с их уровнем профессиональной квалификации, а также с многообразием знаний, опыта, стимулов и потребностей. Быстрый рост сложности и повышение ответственности за качество комплексов программ привели к появлению ***новых требований к специалистам,*** обеспечивающим все этапы жизненного цикла ПС. При проектировании ПС различных классов разделение труда специалистов по квалификации при разработке программ и данных, организация коллективов и экономика таких разработок стали важнейшей частью выбора, обучения и подготовки специалистов для обеспечения всего ЖЦ ПС (см. лекцию 9).

В ***детальной модели COCOMO*** значительное внимание уделено ***влиянию организации и взаимодействия коллектива разработчиков*** на трудоемкость создания сложных программных средств. В составе организационных характеристик коллектива рекомендуется учитывать согласованность целей специалистов, участвующих в проекте, их психологическую совместимость и способность к дружной коллективной работе, наличие опыта работы в данном коллективе и другие объективные и субъективные свойства участников проекта. При этом большое значение могут иметь личная мотивация и психологические особенности поведения разных специалистов при комплексной работе над сложным проектом. Эти характеристики могут быть обобщены в качественный показатель влияния слож-

ности взаимодействия специалистов в коллективе, которому сопоставлены коэффициенты изменения трудоемкости разработки ПС. Наилучшим считается продолжительное корректное взаимодействие организованных специалистов с большим опытом работы в данном коллективе при полной согласованности их целей, планов и методов работы. При разработке программ большими коллективами значительно повышается роль ***квалификации руководителей*** разработки, что непосредственно отражается на средней производительности труда всего коллектива. Однако формализовать и учесть влияние руководителя разработки и ведущих специалистов на затраты и ТЭП комплекса программ пока трудно.

Уровень ***квалификации заказчика*** и определенность технического задания на разработку ПС может весьма сильно влиять на суммарные затраты и длительность создания программ. Изменения технического задания и объем переделок непосредственно отражаются на средней производительности труда специалистов, рассчитанной по конечному размеру созданного комплекса программ. Особенно сильно на достоверность технического задания и возрастание затрат влияет попытка заказчика форсировать сроки разработки. При этом первоначальное техническое задание оказывается недостаточно квалифицированным и подвергается в дальнейшем многократным изменениям. Этому же может способствовать различие между заказчиком и разработчиком в квалификации, уровне понимания целей разработки и необходимых затрат на реализацию дополнительных требований.

При проектировании и создании высококачественных комплексов программ, прежде всего, необходимы организация и ***тесное взаимодействие представителей заказчика и разработчиков*** проекта. Взгляды и требования заказчика в основном отражаются в функциональных и потребительских характеристиках ПС. Устремления разработчиков направлены на возможность и способы их реализации с требуемым качеством. Эти различия исходных точек зрения на проект приводят к тому, что некоторые неформализованные представления тех и других имеют зоны неоднозначности и взаимного непонимания, что может приводить к конфликтам.

Затраты и труд специалистов при реализации крупномасштабного проекта ПС можно распределить по ***двум категориям специалистов***: разрабатывающим компоненты и ПС в целом и обеспечивающим технологию

и качество программного продукта (см. лекцию 9). Организационное разделение специалистов, осуществляющих разработку ПС (первая категория), и специалистов, контролирующих и управляющих его качеством в процессе разработки и всего ЖЦ (вторая категория), должно обеспечивать эффективное достижение заданных характеристик, а также независимый, достоверный контроль затрат и качества результатов разработки.

Затраты на технологию и инструментальные программные средства автоматизации разработки ПС обычно являются весьма весомыми при использовании высокоэффективных автоматизированных технологий. При ***технико-экономическом обосновании проекта следует учитывать***, что размер и сложность создаваемого ПС значительно влияют на выбор инструментальных средств и уровня автоматизации технологии, а также на долю этих затрат в общих затратах на разработку. Встречаются ситуации, при которых затраты на технологию достигают 30—50% общих затрат на разработку. Такие затраты могут быть оправданы повышением производительности труда, сокращением сроков разработки и последующим снижением затрат на множество базовых версий ПС. Однако чаще всего эта группа затрат при создании первой версии сложных ПС находится в пределах 30% от суммарных затрат. В первом приближении степень автоматизации разработки программ отражает размер программных средств, используемых в технологических системах. Этот показатель соответствует сложности систем автоматизации разработки программ и пропорционален затратам на их приобретение (или создание) и эксплуатацию.

Стремление уменьшить технологические затраты в период разработки без учета последующего использования ПС, его компонентов и всего жизненного цикла может оказаться мало полезным, а в некоторых случаях привести к значительному увеличению совокупных затрат в ЖЦ. При применении сложных ПС эти затраты исчисляются сотнями человеко-лет, что определяет особую актуальность их снижения. Поэтому необходим системный анализ распределения и использования технологических ресурсов на разработку программ ***с учетом всего их жизненного цикла***, включая сопровождение и возможный перенос на другие платформы.

Уровень автоматизации и качество технологии и инструментальных средств, используемых для поддержки всего жизненного цикла ПС, обычно сильно коррелирован с достигаемым качеством комплексов про-

грамм, а также с качеством средств автоматизации для оценивания этого качества. Поэтому определение уровня зрелости технологической поддержки процессов жизненного цикла, организационного и инструментального обеспечения качества ПС, непосредственно связано с выбором и оцениванием реальных или возможных характеристик качества конкретного комплекса программ. В ***модели COCOMO для оценивания технико-экономических показателей при разработке ПС рекомендуется методология*** сложных программных средств **CMM — *система и модель оценки зрелости*** комплекса, применяемых технологических процессов жизненного цикла ПС (см. лекцию 3). Эти ***уровни зрелости характеризуются*** степенью формализации, адекватностью измерения и документирования процессов и продуктов ЖЦ ПС, широтой применения стандартов и инструментальных средств автоматизации работ, наличием и полнотой реализации функций системой обеспечения качества технологических процессов и их результатов. В модели **COCOMO** приводятся количественные рекомендации ***коэффициентов влияния уровней зрелости CMM на трудоемкость*** реализации сложных проектов ПС. Влиянию технологической зрелости разработки ПС в детальной модели **COCOMO** сопоставлены уровни CMM, для каждого из которых приводятся коэффициенты изменения трудоемкости разработки.

Для ***приближенной оценки влияния на трудоемкость*** некоторых характеристик процессов разработки ПС в детальной модели **COCOMO** выделены небольшая группа показателей и соответствующие им наборы рейтингов. Инструментальные системы, поддерживающие разработку, описаны качественными характеристиками и рейтингами, изменяющими трудоемкость в пределах приблизительно 20% от средней — номинальной. Уровень технологии и комплекса инструментальных средств особенно сильно влияет на ТЭП крупных проектов ПС. Поэтому затраты на их реализацию и применение целесообразно учитывать конкретно с использованием функций и характеристик проекта.

В модели и таблицах отмечено значительное влияние на трудоемкость ПС директивного ***ограничения сроков разработки*** относительно типовых — номинальных. При ограничении сроков сокращению трудоемкости может сопутствовать значительное снижение качества ПС и увеличение рисков реализации проекта. Для уменьшения сроков разработки

есть ряд путей, с помощью которых руководство может добиваться некоторого ускорения разработки за счет увеличения трудоемкости и стоимости проекта, для чего рекомендуется:

— обеспечить детальные структурирование комплекса программ на модули и спецификации интерфейса для обеспечения максимального параллелизма работы специалистов;

— приобрести и освоить технологические, инструментальные средства для более быстрого кодирования, контроля и тестирования и обучить разработчиков их использованию;

— обеспечить дополнительную подготовку программистов и группы тестирования к работе в тематической области функций проекта;

— привлечь дополнительный вспомогательный персонал;

— отложить на время несущественное документирование проекта.

Тем не менее есть предел сокращению сроков разработки с помощью увеличения числа специалистов и приобретения оборудования. При максимально возможном сокращении сроков разработки до 75% от оптимального затраты возрастают на 25%.

При разработке комплексов программ систем реального времени ***большие затраты*** могут потребоваться для обеспечения тестирования и испытаний ПС, которые непосредственно ***не учитываются моделью СОСОМО***. Основные усилия сосредоточивались на процессах программирования и автономной отладки компонентов. На этих этапах выявлялась основная масса дефектов и ошибок, хотя при использовании ПС и сопровождении обнаруживалось некоторое их количество. В последнее время центр тяжести разработки сложных ПС сдвинулся к начальным этапам и внимание акцентировано на ***предотвращение ошибок***, прежде всего путем тщательного системного проектирования ПС из готовых программных компонентов, а также на комплексной отладке и испытаниях в реальном времени.

Этапы комплексной отладки, испытаний и модификации программ имеют много общего, в основе которого лежит широкое применение их тестирования для обнаружения ошибок и удостоверения функциональной корректности ПС. Разработка детерминированных тестов при отладке модулей и некоторых групп программ в большинстве случаев производится вручную. Однако их доля может составлять заметную часть общих затрат на отладку компонентов. Достаточно автономными и локализуемыми обыч-

но являются затраты на стохастические тесты и тесты в реальном времени, используемые при комплексной отладке и испытаниях (см. лекцию 14).

При технико-экономическом обосновании проектов комплексов программ на оценку трудоемкости их разработки может оказать существенное влияние ***ограниченность вычислительных ресурсов специализированных, реализующих ЭВМ реального времени***. Это привело к необходимости разрабатывать методы эффективного использования аппаратных ресурсов ЭВМ. Одним из важнейших и наиболее общих показателей, характеризующих возможность применения таких ЭВМ, является их производительность для конкретных задач реального времени. При этом существенным ограничивающим фактором являются длительность, в течение которой ЭВМ может быть предоставлена для решения данной задачи, или то реальное время, в пределах которого целесообразно получить результаты для их практического использования. При ограничениях ресурсов вследствие требований минимизации весов и габаритов специализированных, реализующих ЭВМ в авиационных, ракетных и космических системах, их ***экономное использование остается актуальным*** и его следует учитывать при обосновании соответствующих проектов ПС.

Быстрый рост в мире масштабов — размеров комплексов программ и баз данных, решающих единую целевую задачу, потребовал создания новых, более эффективных методов разработки сложных систем. Возникла проблема разработки функционально законченных ПС и БД и их компонентов, потенциально готовых к многократному применению в различной внешней и операционной среде, а также в различных сочетаниях их взаимодействия. ***Унификация всегда требует некоторых ресурсов***, которые в данном случае выражаются в дополнительной трудоемкости создания повторно используемых программ и данных, а также в увеличении необходимой памяти и производительности ЭВМ для их реализации. Сохранение и развитие довольно широкого спектра архитектур ЭВМ, естественно, привело к повторному использованию компонентов (ПИК) не только на однотипных платформах, но и к разработке ПС и БД, переносимых на различные аппаратные и операционные платформы. При этом выделились ***две технологические проблемы***:

— создание программных компонентов и баз данных, которые рентабельно повторно применять и/или переносить на различные операционные и аппаратные платформы;

— проблема реализации повторного использования и/или переноса ПС и БД для создания из них новых систем на иных платформах.

Увеличение затрат при решении первой проблемы должно компенсироваться сокращением затрат при создании комплексов программ и баз данных на базе готовых компонентов. Освоение методов и средств решения этих проблем позволило ***качественно изменить*** процессы создания сложных комплексов программ и ***резко повысить производительность труда*** специалистов при их разработке. Это активизировало в последние годы интерес к проблеме мобильности программ и данных во всех отраслях применения вычислительной техники. Создание новых ПС и БД путем переноса их с других аппаратных и операционных платформ стало особенно актуальным для современных административных систем государственного и регионального управления, управления отраслями и предприятиями, а также банковскими системами и в социальной сфере.

На практике при создании нового ПС ***не всегда имеется полный набор готовых, и пригодных для применения программных компонентов***. Тогда при сборке версии ПС может потребоваться доработка отдельных компонентов, их сопряжение в новых сочетаниях и создание новых программ для решения дополнительных задач. Поэтому целесообразно оценивать трудоемкость сборочного программирования с учетом частичных затрат на новые компоненты. Относительное снижение трудоемкости разработки в первом приближении пропорционально доле готовых ПИК. В пределе при создании базовой версии ПС полностью из многократно применяемых готовых компонентов трудоемкость может сократиться в 3—5 раз. В промежуточных случаях, когда готовые компоненты используются частично, оценку изменения трудоемкости можно провести по степени сокращения затрат на программирование и автономную отладку всех необходимых компонентов.

5.2. Методика 1 — экспертное технико-экономическое обоснование проектов программных средств

В ***этой методике*** реализован метод прогноза ТЭП с учетом ***экспертной оценки минимального числа факторов***. Данная методика оценки

ТЭП может применяться, когда определены цели и общие функции проекта ПС, сформулированные в концепции и первичных требованиях с достоверностью около 20—40%. Основная ***цель оценки ТЭП*** — подготовить возможность принять обоснованное решение о допустимости дальнейшего ***продвижения проекта*** в область системного анализа, разработки требований и предварительного проектирования. Если оказывается, что рассчитанные технико-экономические показатели и требуемые ресурсы не могут быть обеспечены для продолжения проекта, то возможны кардинальные решения: либо изменение некоторых ТЭП и выделяемых ресурсов, либо прекращение проектирования данного ПС. Учитывая полноту и достоверность доступных характеристик и требований к проекту ПС, должны быть определены ***цели и возможная достоверность технико-экономического обоснования продолжения проектирования ПС***.

При первичном технико-экономическом обосновании сложных проектов ПС составляется таблица 5.2 — исходными для которой являются ***концепция проекта ПС и комплекс предварительных требований*** к иерархическому набору функций, которые могут быть разбиты на предполагаемые фактические компоненты ПС. В дальнейшем разбиение может детализироваться, формируя упрощенный или более точный уровень абстракции и взаимодействия компонентов. Наиболее низкий и глубокий уровень детализации, как правило, редко формируется ко времени первоначальной экспертной оценки размера ПС.

Таблица 5.2

Бланк для экспертных оценок исходных данных технико-экономических показателей разработки комплексов программ

Экспертные оценки исходных данных	Средние	Оптимистические	Пессимистические
1. Размер — масштаб комплекса программ (тысячи строк текста с указанием языка программирования)			
2. Относительное число строк готовых повторно используемых программных компонентов (%)			
3. Исходная производительность труда при разработке новых программ ПС (число строк на человеко-месяц)			
4. Исходная стоимость разработки одной строки текста программ			
5. Распределение трудоемкости по этапам работ (график или таблица)			

Эти факторы могут быть оценены квалифицированными экспертами на основе имеющегося у них опыта реализации предшествовавших подобных проектов, а также использования опубликованных данных. При наличии необходимых данных важно оценить их достоверность и возможную точность (20—40%). Наименее точный из перечисленных факторов полностью определяет достоверность расчета технико-экономических показателей проекта ПС, поэтому желательно, чтобы значения точности экспертного оценивания перечисленных факторов были сбалансированы.

При наличии перечисленных исходных данных и положительной оценке целесообразности экспертного анализа ТЭП проекта может реализовываться ***методика, состоящая из следующих шагов***:

— определение класса, сложности функций проекта программного средства;

— экспертная оценка размера — масштаба, числа строк предполагаемого текста разрабатываемых программ, с учетом размера повторно используемых компонентов и характеристик возможного языка программирования;

— экспертная оценка возможной средней производительности труда специалистов при разработке программ и/или стоимости (и длительности) разработки одной сроки текста программ проекта ПС;

— расчет возможной полной трудоемкости и длительности разработки проекта ПС, а также среднего числа специалистов, необходимых для его реализации;

— обобщение основных технико-экономических показателей и полной стоимости разработки проекта ПС, анализ результатов и технико-экономическое обоснование рентабельности продолжения проектирования комплекса программ.

Достоверность прогнозов ТЭП зависит, прежде всего, от ***точности экспертной оценки исходных данных***: размера — масштаба ПС и от достоверности экспертной оценки производительности труда специалистов или оценки стоимости разработки одной строки текста программ (см. таблицу 5.2). Кроме того, экспертные оценки зависят от компетенции и объективности экспертов, их ***оптимистичности, пессимистичности***, знания существенных особенностей проекта.

Экспертная оценка размера проекта программного средства наиболее сложная задача в этой методике. Приступая к разработке комплекса

программ, как в любой профессиональной деятельности, необходимо сначала провести реалистическую оценку возможного ***масштаба проекта*** — поставленных целей, ресурсов проекта и выделенного времени. Задача управления масштабом состоит в задании базовых требований, которые включают разбитое на компоненты ограниченное множество функций и требований, намеченных для реализации в конкретной версии проекта. ***Базовый уровень масштаба должен обеспечивать***:

— приемлемый для заказчика минимум функций и требований к проекту;

— разумную вероятность успеха с точки зрения возможностей и ресурсов коллектива разработчиков.

При оценивании масштаба следует определить приоритеты функций для установления состава работ, согласованного между заказчиком и разработчиком, которые ***обязательно*** должны быть выполнены и для определения базового уровня масштаба конкретного проекта ***с допустимым риском*** неуспешной реализации. Сокращение масштаба проекта до размеров, адекватных выделенному времени и ресурсам, может привести к конфликтам заказчиков и разработчиков. Для уменьшения вероятности таких конфликтов целесообразно активно привлекать заказчиков к управлению их требованиями и масштабом проекта, чтобы обеспечить как качество, так и своевременность разработки ПС.

Экспертные оценки удельных затрат на строку текста программ относятся к полному циклу разработки крупных комплексов программ, начиная от создания концепции и требований до завершения испытаний и передачи программного продукта заказчику или пользователям, с учетом всего состава коллектива специалистов всех квалификаций. По мнению некоторых специалистов, несмотря на появление новых методов и инструментальных средств разработки сложных ПС, средняя производительность при их создании за последние двадцать лет осталась почти неизменной и составляет около 3000 строк кода на одного разработчика проекта в год (порядка 250 строк на человеко-месяц). Это отражает то, что уменьшение времени, затрачиваемого на цикл разработки, не может быть достигнуто за счет значительного повышения производительности труда отдельных специалистов. Причем это практически не зависит от усовершенствований языка программирования, организационных усилий со стороны менедже-

ров, от наличия или отсутствия некоторых отдельных видов инструментария и автоматизации работ, хотя значительную роль играет увеличившаяся ***доля повторно используемых компонентов*** (ПИК). На самом деле при достаточно высоком уровне технологии (3—4-й уровень СММ — см. лекцию 3) большое значение имеют ***возросший размер*** и сложность состава функциональных задач комплексов программ, а также значительное ***повышение требуемого качества*** создаваемых ПС.

В ***качестве ориентиров*** при экспертной оценке ТЭП для таблицы 5.2 можно использовать следующие данные ***средней трудоемкости*** разработки сложных комплексов программ. Весьма общие данные опубликованы в виде широких диапазонов производительности труда: для относительно простых ПС — 8 LOC на человеко-день и 4 LOC на человеко-день для достаточно сложных ПС. Также приводятся широкие диапазоны производительности труда при разработке программ на ассемблере — 60—500 LOC на человеко-месяц, и 50—300 LOC на человеко-месяц для языков высокого уровня. Подобные оценки можно использовать ***как ориентиры*** при первичных определениях ТЭП.

Более точные оценки производительности при разработке программ различного размера и классов на основе обобщения статистических данных множества проектов представлены в базовой модели COCOMO:

— для программ административных систем (ИПС) размером порядка 30 тысяч строк оценка производительности составляет около 220 строк на человеко-месяц, а для ПС размером 500 тысяч строк — 160 строк на человеко-месяц;

— для встроенных комплексов программ реального времени размером 30 тысяч строк рекомендуется для оценок использовать производительность около 140 строк на человеко-месяц, а для крупных ПС размером 500 тысяч строк предлагается значение производительности около 80 строк на человеко-месяц.

Эти данные находятся в середине представленных выше диапазонов и их целесообразно использовать при экспертной оценке полной трудоемкости разработки соответствующих новых ПС. При использовании готовых повторно используемых компонентов обобщенная производительность труда возрастает и зависит от доли таких компонентов в комплексе программ. Их также можно использовать для оценки ***полной стоимости***

проекта конкретного ПС. Однако при этом необходимы удельные данные средней стоимости труда одного человеко-месяца специалистов в конкретном предприятии с учетом всех накладных расходов, которые могут различаться в несколько раз. Такие сведения обычно являются коммерческой тайной, и при использовании данной методики для определенного проекта ПС их следует запрашивать у экономических служб конкретного предприятиям. Тем не менее опубликованы ***ориентиры*** стоимости разработки одной строки текста программ реального времени — около 100$ и более, а для административных систем — около 20—50$.

Экспертная оценка длительности разработки сложных ПС (таблица 5.3), может базироваться на формулах модели COCOMO (см. п. 5.3). Основой для расчета длительности целесообразно использовать рассчитанную ранее трудоемкость разработки проекта ПС, от которой ***не линейно*** зависит длительность (месяцы), приблизительно равная трудоемкости (человеко-месяцы) в степени 0,3. Например, крупные проекты ПС реального времени размером около 500 тысяч строк требуют для реализации около 3,5 лет, а небольшие (30 тысяч строк) — около одного года. При этом следует учитывать, что необходимая численность коллектива специалистов изменяется в десяток раз.

Таблица 5.3

Бланк расчетных или экспертных оценок технико-экономических показателей разработки комплексов программ

Экспертные оценки расчетных данных	Средние	Оптимистические	Пессимистические
1. Полная трудоемкость разработки комплекса программ (человеко-месяцы с указанием языка программирования)			
2. Полная длительность разработки комплекса программ (месяцы)			
3. Необходимое среднее число специалистов (человек)			
4. Распределение трудоемкости по этапам работ (график или таблица)			
5. Распределение длительности по этапам работ (график или таблица)			
6. Распределение числа специалистов по этапам работ (график или таблица)			

Экспертная оценка необходимого числа специалистов всех квалификаций рассчитывается путем деления полной трудоемкости разработки ПС на длительность ее реализации. Для примера крупного проекта ПС реального времени, размером 500 тысяч строк, необходимое число специалистов достигает 160 человек, а для относительно небольшого проекта (30 тысяч строк) — в десять раз меньше (16 человек). Аналогично можно получить оценки необходимого числа специалистов на выделенных крупных этапах разработки ПС, что полезно для первичного формирования коллектива и оценки возможности реализации им конкретного проекта ПС (см. таблицу 5.1).

5.3. Методика 2 — оценка технико-экономических показателей проектов программных продуктов с учетом совокупности факторов предварительной модели COCOMO II

В **COCOMO II** для оценки ТЭП представлены две модели — для этапов предварительного и детального проектирования (см. Boehm B.W. et al. Software cost estimation with COCOMO II. Prentice Hall PTR. New Jersey. 2000). Предварительная модель предназначена для анализа общих, архитектурных решений и выработки стратегий последующей разработки при начальных сведениях о содержании предварительного проекта. Детальная модель рекомендуется для применения при разработке наиболее сложных и дорогих систем, когда требуется тщательный учет ряда факторов, влияющих на оценку стоимости на уровне технического проекта ПС. В обеих моделях исходными и определяющими достоверность прогнозов являются размер комплекса программ (с учетом повторно используемых компонентов) и совокупность набора влияющих факторов.

На ***этапе предварительного проектирования комплекса программ***, после утверждения требований и концепции проекта, повышается достоверность данных о функциях, спецификациях, компонентах и структуре разрабатываемого ПС. Это позволяет, прежде всего, более точно оценить размер — масштаб ПС и возможность использования готовых программных компонентов из других аналогичных проектов. Если достоверность

оценки размера ПС достигает 15—20%, то при определении ТЭП целесообразно сбалансированно выделять и учитывать факторы, влияние которых на трудоемкость и стоимость достаточно велико и составляет также около 20%. Таких факторов может быть около 5—10, и их число зависит от конкретных характеристик объекта и среды разработки ПС. При технико-экономическом обосновании проекта ПС на этом этапе состав и номенклатура учитываемых факторов должны выбираться путем включения в анализ тех факторов, которые достаточно влияют на ТЭП конкретного проекта.

Так как ***величина и достоверность определения размера проекта ПС — ключевой фактор*** последующего технико-экономического анализа, то целесообразно применять несколько доступных методов для его оценивания. Конкретизация функций, структуры ПС и состава компонентов проекта позволяет более достоверно определить размеры групп программ и, суммируя их, рассчитать размер всего комплекса программ. Кроме того, на этом этапе повышается возможность выбора и использования аналогов данного проекта, так как становятся более определенными задачи, функции и компоненты разрабатываемого ПС, для которого желательно найти аналоги с известными апробированными характеристиками. Особенно сильно на ТЭП влияет использование готовых компонентов из предшествующих разработок. При анализе аналогов могут быть выделены компоненты, пригодные для повторного применения в новом проекте. Это позволяет оценить возможную долю использования готовых компонентов и тем самым определить эквивалентный размер комплекса программ, подлежащий непосредственной разработке.

Для трех классов комплексов программ в предварительной модели COCOMO II представлены коэффициенты для расчета зависимости трудоемкости разработки программ ***С*** (человеко-месяцы) от их объемов — ***П*** (тысячи строк текста) (таблица 5.4). Для аппроксимации зависимости трудоемкости от размера ПС рекомендуется использовать ***степенную функцию*** вида:

$$C = A \times П^{E} \qquad (5.1)$$

При разработке ПС большого размера должна возрастать сложность разработки по сравнению с ПС малого объема, так как в больших программах существенно усложняются взаимосвязи компонентов по инфор-

мации и управлению, а также становятся более трудоемкими процессы планирования и управления проектом в ходе разработки. Выдвинутая гипотеза о возрастании трудоемкости разработки с ростом размера ПС быстрее, чем по линейному закону, справедлива, если показатель степени в уравнении регрессии $E > 1$. В ряде работ определены коэффициенты A и E в уравнениях степенной регрессии, показывающие характер зависимости трудоемкости от размера ПС. В таблице 5.4 представлены значения коэффициентов регрессии для моделей COCOMO для трех основных классов проектов программных средств. Выражение (5.1) с использованием этих коэффициентов и значений размера ПС — ***П*** в ***тысячах строк ассемблера*** — рекомендуется для прогнозирования трудоемкости полной разработки в ***человеко-месяцах***.

Таблица 5.4

Коэффициенты моделей для оценки трудоемкости разработки программных средств

Коэффициент A	Коэффициент E	Модель и тип программных средств
2,4	1,05	Базовая — COCOMO
		Детализированная модель COCOMO:
3,6	1,20	— встроенный;
3,0	1,12	— полунезависимый;
2,4	1,05	— независимый

Длительность разработки программных средств является важнейшим ТЭП, поскольку часто она определяет общие сроки разработки систем, а значит, быстроту реализации идей в различных областях автоматизации. При определении коэффициентов в таблице 5.5 за начало разработки ПС принят момент начала создания технического задания, а за окончание — завершение испытаний программного продукта в целом. Диапазону размеров современных ПС в три-четыре порядка (до 10 млн строк) соответствуют приблизительно такие же диапазоны изменения трудоемкости и стоимости их разработок. Однако очевидна принципиальная нерентабельность разработки даже очень сложных ПС более 5 лет. С другой стороны, программы даже в несколько тысяч строк по полному технологическому циклу с испытаниями как продукции редко создаются за время, меньшее чем полгода-год. Таким образом, вариация длительностей разработок ПС меньше, чем вариация их трудоемкости, и не превышает десятикратный

диапазон. Длительности разработок — ***T*** ограничены сверху и снизу, и одним из основных факторов, определяющих эти границы, является масштаб комплекса программ — ***П***.

Таблица 5.5

Коэффициенты моделей для оценки длительностей разработки программных средств

Коэффициент *G*	Коэффициент *H*	Модель и тип программных средств
2,5	0,38	Базовая — COCOMO
2,5 2,5 2,5	0,32 0,35 0,38	Детализированная модель COCOMO: — встроенный; — полунезависимый; — независимый

Чтобы сократить ошибки, связанные с неопределенностью измерения размера программ, исследована ***зависимость длительности разработки от ее трудоемкости***. Учитывалась только трудоемкость непосредственной разработки программ ***C*** без затрат на средства автоматизации разработки. Обработка тех же, что выше, наборов данных позволила получить коэффициенты уравнения регрессии, представленные в таблице 5.5. Обобщенные данные длительности разработки — ***T*** по классам комплексов программ ***аппроксимированы уравнениями регрессии*** по методу наименьших квадратов в зависимости от размера ПС и от трудоемкости их разработки:

$$T = G \times C^{H}. \qquad (5.2)$$

Установлено, что длительность разработки ПС меньше подвергается изменениям при автоматизации разработки или другими методами, чем трудоемкость или производительность труда. Необходимость выполнения при разработке ПС определенной совокупности этапов и операций в заданной технологической последовательности остается более или менее постоянной при различных воздействиях на процесс разработки. Исключением является применение повторно используемых компонентов (ПИК), при котором значительно сокращаются этапы программирования и автономной отладки модулей и групп программ, а также в той или иной степени длительность других этапов.. Поэтому зависимость ***T*** от доли ПИК оказывается нелинейной, и заметное сокращение длительности разработ-

ки проявляется только при создании базовой версии ПС практически полностью из готовых компонентов.

Оценка требуемого среднего числа специалистов для конкретного проекта ПС предварительно может быть рассчитана путем деления оценки величины трудоемкости разработки (5.1) на длительность разработки (5.2). Однако рациональное число специалистов, участвующих в проекте ПС, распределяется неравномерно по этапам работ (см. таблицу 5.1). Поэтому целесообразно определять число и квалификацию необходимых специалистов с учетом этапов разработки комплексов программ. ***Обобщенные значения предварительного расчета ТЭП*** целесообразно оформлять в виде таблицы 5.3, где оценки представляются также с учетом пессимистических и оптимистических результатов определения масштаба проекта комплекса программ.

Для учета ***влияния на трудоемкость различных факторов*** удобно пользоваться коэффициентами (рейтингами) изменения трудоемкости (КИТ) — ***M*** (***i***), учитывающими зависимость от ***i***-го фактора на совокупные затраты труда. В них входят факторы процесса непосредственной разработки, факторы программной и аппаратурной оснащенности, а также квалификация специалистов (таблица 5.6). Затраты на разработку ***C*** и объем программ ***П*** могут быть связаны через показатель интегральной ***средней производительности труда*** разработчиков ***P***. Непосредственно затраты на разработку можно представить как частное от размера ПС и производительности труда ***P = 1 / A***, корректируемой произведением коэффициентов изменения трудоемкости (КИТ — ***M*** (***i***)):

$$C = \frac{П^E}{P} \times \prod_i M(i) = A \times П^E \times \prod_i M(i). \qquad (5.3)$$

Средняя производительность труда коллектива специалистов при разработке сложного полностью нового комплекса программ ***P*** в выражении (5.3) может служить ***ориентиром*** для сравнения эффективности труда при создании различных проектов ПС. Эта характеристика, конечно, различается для различных классов, размеров и других параметров комплексов программ, однако диапазон этих различий не столь велик, как изменения размера и требований к качеству. Так, при диапазоне изменения размеров программ реального времени на четыре порядка средняя произ-

водительность труда изменяется только в два раза, что в ряде случаев существенно облегчает упрощенные оценки и прогнозирование ТЭП.

В ***предварительной модели COCOMO II*** при достаточной достоверности определения масштаба ПС рекомендуется в выражении (5.3) учитывать семь факторов ***M*** (***i***), представленных в таблице 5.6. Для каждого из них в таблице 5.7 приведены значения рейтингов (коэффициентов изменения трудоемкости), которые целесообразно использовать при выборе определенного содержания факторов конкретного проекта ПС, а также свободный столбец для выбранных значений. Результаты уточненного расчета трудоемкости проекта ПС по формуле (5.3) следует использовать для последующего определения длительности (выражение 5.2) и требуемого среднего числа специалистов.

Таблица 5.6

Состав и максимальные значения факторов предварительной модели COCOMO II

Фактор	Символ	Макс. значение	Содержание фактора и его составляющие
			Требования к объекту разработки
PCPX	M1	5,55	RELY; DATA; CPLX; DOCU Сложность и надежность программного продукта
RUSE	M2	1,31	RUSE Требование повторного использования компонентов
			Характеристики коллектива специалистов
PERS	M4	4,24	ACAP; PCAP; PCON Квалификация специалистов и стабильность коллектива
PREX	M5	2,56	APEX; PLEX; LTEX Опыт работы по тематике и с инструментарием
			Технологическая среда разработки
PCIL	M6	2,31	TOOL; SITE Уровень инструментальной поддержки и необходимость распределенной разработки
SCED	M7	1,43	SCED Ограничение длительности разработки
			Аппаратурно-вычислительная среда разработки
PDIF	M3	1,00	TIME; STOR; PVOL Ограничения аппаратной платформы разработки и реализации

Таблица 5.7

Уровень оценки

Интегральные факторы	Уровень оценки					
	Очень низкий	Низкий	Номинальный	Высокий	Очень высокий	Сверхвысокий
Сложность и надежность	0,81	0,98	1,00	1,30	1,74	2,38
Требования повторного использования компонентов		0,95	1,00	1,07	1,15	1,24
Квалификация специалистов	1,62	1,26	1,00	0,83	0,63	0,50
Опыт работы	1,33	1,12	1,00	0,87	0,71	0,62
Инструментальная поддержка	1,30	1,10	1,00	0,87	0,73	0,62
Ограничение длительности разработки	1,43	1,14	1,00	1,00	1,00	
Аппаратурно-вычислительная среда		0,87	1,00	1,29	1,81	2,61

5.4. Методика 3 — уточненная оценка технико-экономических показателей проектов программных продуктов с учетом полной совокупности факторов детальной модели COCOMO II.2000

При ***детальном проектировании*** возможно значительное повышение точности определения размера — масштаба проекта комплекса программ. Последовательная детализация и конкретизация проекта ПС позволяет уточнять его будущий размер и привлекать для расчета трудоемкости большее число факторов, способных повысить точность прогноза всех ТЭП. Разработка полного содержания спецификаций функций и структуры программных компонентов, их взаимодействия и интерфейсов, а также архитектуры всего комплекса программ и базы данных обычно позволяют повысить точность определения размера ПС приблизительно на 10%. По-

этому при расчете трудоемкости разработки при прогнозировании целесообразно выбирать и учитывать влияние ряда дополнительных факторов из таблицы 5.8, которые не оценивались в методиках 1 и 2, вследствие их относительно меньшего влияния.

Таких дополнительных факторов обычно может быть выделено около 10, которые целесообразно рассматривать и учитывать при оценках, если он способен изменить трудоемкость разработки конкретного проекта на 5—10%. Анализ, выбор и оценивание коэффициентов влияния ***F (j)*** и ***M (i)***, этих дополнительных факторов, — довольно сложный процесс. Он оправдан, когда совместное влияние совокупности этих дополнительных факторов может изменить оценки трудоемкости на 10—20%. В результате расчет трудоемкости несколько усложняется, однако процессы последующего расчета длительности разработки и необходимого числа специалистов практически не изменяются. В целом процессы методики 3 технико-экономического обоснования проекта ПС с учетом ряда дополнительных факторов практически не отличаются от предыдущей предварительной методики 2, однако требуется более тщательное определение размера комплекса программ и оценивания влияния на трудоемкость разработки большего числа факторов.

Для ***более точного технико-экономического обоснования проектов ПС при детальном проектировании*** обычно целесообразно учитывать влияние ряда дополнительных факторов из четырех групп, что позволяет повысить достоверность прогнозирования технико-экономических показателей ПС до уровня около 5—10%. В ***детальной модели COCOMO II*** влияние на трудоемкость определяют 22 фактора, из которых пять — масштабные факторы, характеризуются множителем ***F (j)*** в значении степени размера ПС, а 17 множителей ***M (i)*** непосредственно изменяют трудоемкость разработки. Перечень, максимальные значения и содержание этих множителей представлены в таблице 5.8. При этом номинальными (средними) ниже принимаются все ***M (i)*** = 1,00, при которых соответствующий фактор практически не влияет на трудоемкость ПС.

Для выполнения ***оценок трудоемкости разработки*** (человеко-месяцы) ***в детальной модели COCOMO II*** предложены выражения, уточняющие зависимости, представленные выше в п. 5.3.

Таблица 5.8

Состав и максимальные значения факторов детальной модели COCOMO II

Фактор	Символ	Макс. значение	Содержание фактора
			Масштабные факторы
PREC	F1	1,33	Новизна проекта
FLEX	F2	1,26	Согласованность с требованиями и интерфейсами
RESL	F3	1,39	Управление рисками и архитектурой проекта
TEAM	F4	1,29	Слаженность работы коллектива
PMAT	F5	1,43	Технологическая зрелость обеспечения разработки
			Факторы, влияющие на затраты разработки
			Требования и характеристики объекта разработки
RELY	M1	1,54	Надежность функционирования
DATA	M2	1,42	Размер базы данных
CPLX	M3	2,38	Сложность функций и структуры
RUSE	M4	1,31	Требование повторного использования компонентов
DOCU	M5	1,52	Полнота и соответствие документации проекта
			Характеристики коллектива специалистов
ACAP	M9	2,00	Квалификация аналитиков
PCAP	M10	1,76	Квалификация программистов
PCON	M11	1,51	Стабильность коллектива
APEX	M12	1,51	Опыт работы по тематике проекта
PLEX	M13	1,40	Опыт работы в инструментальной среде
LTEX	M14	1,43	Опыт работы с языками программирования
			Технологическая среда разработки
TOOL	M15	1,50	Уровень инструментальной поддержки проекта
SITE	M16	1,53	Необходимость распределенной разработки проекта
SCED	M17	1,43	Ограничения длительности разработки проекта
			Аппаратурно-вычислительная среда разработки
TIME	M6	1,63	Ограниченность времени исполнения программ
STOR	M7	1,46	Ограниченность доступной оперативной памяти
PVOL	M8	1,49	Изменчивость виртуальной среды разработки проекта

$$C = A \times \Pi^{E} \times \prod_{i=1}^{n} M(i), \qquad (5.4)$$

где: $A = 2{,}94$; $E = B + 0{,}01 \times \sum_{j=1}^{5} F(j)$; $B = 0{,}91$.

Для прогнозирования ***длительности*** (месяцы) разработки ПС рекомендуются выражения:

$$T = G \times C^{H}, \qquad (5.5)$$

где: $G = 3{,}67$; $H = D + 0{,}02 \times 0{,}01 \times \sum_{j=1}^{5} F(j) = D + 0{,}02 \times (E - B)$;

$D = 0{,}28$.

В модели COCOMO II поддерживаются вероятностные диапазоны оценок, представляющие одно стандартное отклонение на фоне наиболее достоверных оценок. При использовании представленных выражений для прогнозирования ТЭП конкретных проектов следует выбирать набор факторов (***калибровать модель***), имеющих наибольшие значения коэффициентов изменения трудоемкости (КИТ) ***F (j)*** и ***M (i)*** в соответствии с характеристиками конкретного проекта и среды разработки и вставлять ***выбранные значения*** в таблицу 5.8. Значения этих коэффициентов и уровни оценок их влияния на трудоемкость по основным выделенным группам факторов представлены в модели COCOMO II.

Заказчик может заказать разработку специальным образом ***калиброванной версии*** коэффициентов в формулах (5.4) и (5.5), которая должна более точно отражать применяемые технологические процессы, особенности и возможности проекта ПС, чем в методике 2 . При калибровке модели COCOMO II последовательно выполняются следующие процедуры для конкретного проекта:

— выбирается набор факторов ***M (i)*** , оказывающих наибольшее влияние на прогнозируемую трудоемкость проекта ПС;

— устанавливаются значения масштабных факторов ***F (j)***;

— для каждого из выбранных факторов производится оценка коэффициента изменения трудоемкости для анализируемого проекта ПС.

Выбор состава и оценку факторов, влияющих на ТЭП конкретного проекта ПС, предварительно целесообразно проводить по шагам ***при калибровании модели COCOMO II*** на основе совокупности 22 факторов из таблицы 5.8. Первоначально должна производиться оценка коэффициентов влияния пяти групп факторов — ***F (j)***. В выражениях (5.4), (5.5) значения ***M (i)*** отражают коэффициенты влияния ***i***-ых факторов на трудоемкость разработки ПС, которые первоначально (все ***n***) считаются равными единице. Предварительный расчет трудоемкости и длительности разработки ПС при ***M (i)*** = 1 может служить ***уточненным ориентиром***, так

как он базируется на более точном значении размера проекта комплекса программ и более адекватных значениях основных коэффициентов, чем в методике 2.

Выбирать и учитывать следует те факторы, коэффициенты влияния которых на трудоемкость в конкретном проекте имеют достаточную величину, сбалансированную с точностью определения размера комплекса программ или превышают ее. Эти факторы можно разделить на две группы, которые существенно различаются по степени влияния на трудоемкость разработки ПС. В ***первую группу — F (j)*** следует включать, кроме размера и доли повторно используемых компонентов, совокупность факторов, которые способны изменять трудоемкость в несколько (до 3—5) раз:

— новизну проекта комплекса программ;

— необходимую степень согласованности проекта с требованиями технического задания;

— наличие управления рисками и архитектурой проекта;

— уровень обобщенной слаженности и организованности коллективной разработки проекта;

— уровень обеспечения и оснащения технологии разработки по оценке CMM.

Вторую группу — M (i) следует выбирать из совокупности перечисленных в таблице 5.8 семнадцати факторов, таких, которые в конкретном проекте могут повлиять на изменение трудоемкости разработки на 10—20%, соизмеримое с точностью оценок размера ПС:

— требования надежности ПС;

— требования степени соответствия документации программному продукту;

— тематическая квалификация специалистов;

— технологическая квалификация проектировщиков и программистов;

— стабильность состава коллектива разработчиков;

— ограничения ресурсов объектной ЭВМ реального времени;

— стабильность требований заказчика к задачам и функциям ПС.

Остальная совокупность около десяти факторов модели COCOMO II обычно может изменять трудоемкость проекта менее чем на 10%, и их нецелесообразно учитывать в процессе ***детального проектирования***, когда точность оценивания размера проекта ПС может составлять около 10%.

Обобщенные оценки технико-экономических показателей проекта ПС целесообразно представлять в виде таблиц с указанием достоверности оценок результатов расчетов. На основе анализа результатов и оценивания рассчитанных характеристик следует выполнять ***заключительное технико-экономическое обоснование проекта ПС*** и определять:

— целесообразно ли продолжать работы над конкретным проектом ПС в направлении детализации требований, функций и технико-экономических характеристик или следует его прекратить вследствие недостаточных ресурсов специалистов, времени или возможной стоимости и трудоемкости разработки;

— при наличии достаточных ресурсов, следует ли провести маркетинговые исследования для определения рентабельности полного выполнения проекта ПС и создания программного продукта для поставки на рынок;

— достаточно ли полно и корректно формализованы концепция и требования к проекту ПС, на основе которых проводились расчеты ТЭП, или их следует откорректировать и выполнить повторный анализ с уточненными исходными данными;

— есть ли возможность применить готовые повторно используемые компоненты ПС, в каком относительном размере от всего комплекса программ, и рентабельно ли их применять в конкретном проекте ПС или весь проект целесообразно разрабатывать как полностью новый.

ЛЕКЦИЯ 6

РАЗРАБОТКА ТРЕБОВАНИЙ К ПРОГРАММНЫМ СРЕДСТВАМ

6.1. Организация разработки требований к сложным программным средствам

Проекты программных средств различаются по уровню сложности, масштабу и необходимому качеству. Они имеют различное назначение, содержание и относятся к разным областям применения. Поэтому существует потребность в четко организованном процессе, методах формализации и управления требованиями к конкретным программным продуктам. Чаще всего проблемами, с которыми встретились не достигшие своих целей проекты программных продуктов, являются: ***недостаток информации от пользователя или заказчика о функциях проекта, неполные, некорректные требования, а также многочисленные изменения требований и спецификаций***. Это происходит потому, что зачастую разработчики и заказчики считают, что «даже если мы не очень точно знаем, чего хотим достичь, лучше быстрее приступить к реализации проекта, так как мы и так выбились из графика и нам некогда размышлять. Мы можем уточнить требования позднее». Подобный подход приводит к ***хаотическим, неупорядоченным действиям*** при разработке ПС, когда никто не знает, что на самом деле хочет заказчик и пользователь и как в действительности функционирует созданная на данный момент система и/или программный продукт.

Формализация и управление требованиями — это систематический метод выявления, организации и документирования требований к системе и/или ПС, а также процесс, в ходе которого вырабатывается и

обеспечивается ***соглашение*** между заказчиком и выполняющими проект специалистами, в условиях меняющихся требований к системе — рис. 6.1. Развитие программной инженерии, ее обозримое будущее связаны с непрерывным возрастанием сложности, поэтому разработкой ПС должны заниматься ***хорошо организованные*** и обученные коллективы — ***команды*** разработчиков. Каждый член команды в той или иной степени должен привлекаться к управлению и формализации требований к проекту. Команде необходимо выработать профессиональные приемы для понимания потребностей пользователей, управления масштабом ПС, структурой и построением системы, удовлетворяющей эти потребности.

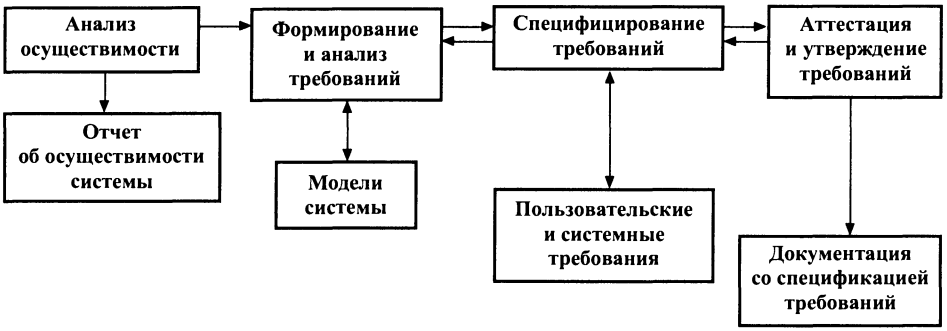

Рис. 6.1

Команда должна применить методы и процессы для того, чтобы ***понять решаемую проблему заказчика до начала разработки ПС***. Для этого следует использовать ***метод анализа, выявления и освоения проблемы и интересов заказчика***:

— достигнуть соглашения между заказчиком и разработчиком по определению проблемы, целей и задач проекта;

— выделить основные причины — проблемы, являющиеся ее источниками и стоящие за основной проблемой проекта системы и ПС;

— выявить заинтересованных лиц и пользователей, чьи коллективное мнение и оценка в конечном итоге определяют успех или неудачу проекта;

— определить, где приблизительно находятся область и границы возможных решений проблем;

— понять ограничения, которые будут наложены на проект, команду и решения проблем.

В результате необходимо ***предложить заказчику возможные решения рассматриваемой проблемы***. Для анализа проблемы можно использовать разнообразные методы. В частности, моделирование бизнес-процессов, специальный ***метод анализа проблем***, который достаточно хорошо зарекомендовал себя для сложных информационных систем, осуществляющих поддержку ключевых инфраструктур. Выявленные на данном этапе прецеденты могут позднее применяться для определения совокупности требований к ПС. Методы системного анализа позволяют осуществить разбиение сложной системы на подсистемы. Этот процесс помогает понять, каким общим целям должно служить ПС. При этом часто оказывается, что в чем-то усложняются требования, создавая новые подсистемы, для которых, в свою очередь, нужно заниматься разработкой и пониманием требований.

Понимание потребностей пользователей — необходимый организационный этап, так как разработчики редко получают совершенные спецификации требований к создаваемой системе, они должны сами ***добывать*** информацию, необходимую им для успешной работы. Термин ***выявление требований*** точно отражает данный процесс, в котором разработчики должны играть активную роль. Чтобы помочь команде решить эти проблемы, лучше понять потребности пользователей и других заинтересованных лиц, целесообразно использовать ***методы***:

— интервьюирования и анкетирования — создание структурированного интервью; проведение интервью с 5—15 пользователями и/или заинтересованными лицами; подведение итогов совокупности интервью, формулирование 10—15 наиболее часто упоминавшихся потребностей заказчика и пользователей;

— совещания, посвященные анализу и синтезу требований — формулирование и определение целей программного продукта; ознакомление с ними всех участников проекта и установление, что они с ними согласны; если это не так, следует остановиться и уточнениями добиться согласия; обязательно убедиться в согласии заказчиков;

— мозговой штурм и отбор идей, чтобы: выявить и/или уточнить функции проекта; отсечь нецелесообразные идеи; провести классифика-

цию функций, чтобы определить приоритеты, риски, трудоемкости реализации функций;

— анализ иллюстративных прецедентов в приложении к концепции требований (или системному проекту), чтобы их функции были наглядны и понятны;

— по возможности выявление или создание временных прототипов на основе первичных требований.

Хотя ни один из методов не является универсальным, каждый из них позволяет лучше понять потребности пользователей и тем самым превратить «неясные» требования, в требования, которые «лучше известны и понятны». Каждый из этих методов эффективен в определенных ситуациях, однако целесообразно отдавать предпочтение совещанию, посвященному требованиям, и мозговому штурму.

Для сложных систем требуются ***стратегии управления информацией о требованиях***. Для этого применяется информационная иерархия; она начинается с потребностей пользователей, описанных с помощью функций, которые затем превращаются в более подробные требования к ПС, выраженные посредством прецедентов или традиционных форм описания и стандартизированных характеристик. Эта иерархия отражает уровень абстракции при рассмотрении ***взаимосвязи области проблемы и области решения***. Концепцию требований, модифицированную в соответствии с конкретным содержанием комплекса программ, необходимо иметь в ***каждом*** проекте. В требованиях к ПС следует указывать, ***какие функции*** должны осуществляться, а ***не то, как*** они могут реализоваться. Они используются для задания функциональных и конструктивных требований, а также ограничений ресурсов проектирования. Нужно использовать набор характеристик для установления и оценки качества ПС и содержащихся в нем компонентов. Если необходимо, документация требований может дополняться одним или несколькими более формальными либо более структурированными методами спецификации требуемого качества.

Без ***лидера руководителя проекта***, который будет утверждать требования к ПС, согласовывать и поддерживать потребности заказчика и команды разработчиков, нельзя быть уверенным в том, что будут приняты необходимые четкие, скоординированные решения. Возможно возникновение флюктуаций требований, задержек, а также принятие неоптималь-

ных решений, вызванное приближением срока окончания проекта. Руководитель должен создать совет по контролю за изменениями, который призван помогать лидеру в принятии действительно сложных решений и гарантировать, что изменения и утверждения требований будут приниматься только после их обсуждения.

Концепция требований к проекту (или системный проект) должна быть «***живым***» документом, чтобы было легче его использовать и пересматривать. Следует сделать концепцию ***официальным каналом*** изменения функций так, чтобы проект всегда имел достоверный, соответствующий современному состоянию документ. Лидер должен нести персональную ответственность за концепцию, регулярно (вместе с командой) оценивать состояние дел и готовить отчеты и запросы, способствующие этим действиям. Процесс отслеживания спецификаций требований к ПС должен помогать убедиться, что концепция надлежащим образом реализуется в детализированных требованиях к компонентам ПС. Руководство процессом контроля за изменениями должно гарантировать, что перед тем, как разрешить внесение существенных изменений в систему, производится оценка рентабельности поступивших предложений (см. лекцию 16).

Проекты, как правило, инициируются с ***объемом функциональных возможностей, значительно превышающим*** тот, который разработчик может реализовать, обеспечив приемлемое качество при заданных ресурсах. Тем не менее необходимо ограничиваться, чтобы иметь возможность предоставить в срок ***достаточно целостный и качественный продукт***. Существуют различные методы задания очередности выполнения (приоритетов) требований и понятие базового уровня — совместно согласованного представления о том, в чем будут состоять ключевые функции системы как продукта проекта — понятие состава требований, задающих ***ориентир*** для принятия решений и их оценки.

Если ***масштаб проекта и сопутствующие требования заказчика превышают реальные ресурсы***, в любом случае придется ограничиваться в функциях и качестве ПС. Поэтому следует определять, что ***обязательно должно*** быть сделано в версии программного продукта при имеющихся ресурсах проекта. Для этого придется вести переговоры. Нельзя ожидать, что данный процесс полностью решит проблемы масштаба и требований к ПС. Но такие шаги окажут заметное воздействие на размеры проблемы,

позволят разработчикам ПС сконцентрировать свои усилия на критически важных подмножествах требований и функций и, в несколько приемов, предоставить высококачественные системы, удовлетворяющие или даже превосходящие основные ожидания пользователей. Привлечение заказчика к решению проблемы управления масштабом и функциями повышает взаимные обязательства сторон, способствует росту взаимопонимания и доверия между заказчиком и разработчиками. Имея достаточное определение функций продукта (концепцию) и сократив масштаб проекта до разумного уровня, можно надеяться на успех в следующих фазах или версиях проекта.

На основе выполненных оценок трудозатрат рекомендуется определять базовый уровень для каждой версии концепции требований, используя атрибут «номер версии», достигнуть ***соглашения с заказчиком относительно масштаба,*** а также принять жесткие решения по масштабу версии. Итеративная разработка и управление изменениями, используя базовый уровень, позволяет контролировать, что все предложенные функции записаны и ничего не потеряно. Целесообразна ***система управления запросами на изменения требований,*** чтобы фиксировать все изменения и гарантировать, что они поступают через эту систему в совет по контролю за изменениями. На всех этапах развития спецификации требований к ПС следует задавать полный набор характеристик функционального и конструктивного поведения программного продукта и подробные прецеденты для основных функций системы.

В требованиях должны быть полно и сжато зафиксированы потребности пользователей в таком виде, чтобы разработчик мог ***построить удовлетворяющее их ПС и его компоненты***. Кроме того, требования должны быть достаточно конкретными, чтобы можно было определить, когда они удовлетворены. Зачастую именно разработчики обеспечивают эту конкретизацию. Существуют различные возможности организации и документирования этих требований. Вся разработка должна вытекать из требований, а все спецификации ПС и компонентов должны найти отражение в процессах и продуктах разработки. Сложный программный комплекс следует пересматривать и обновлять на протяжении всего жизненного цикла проекта.

Проектирование и реализация корректной (правильной) системы, адекватной требованиям, — весьма трудоемкая задача. Один из полез-

ных методов состоит в использовании прототипов требований и прецедентов в качестве основы архитектуры и реализации проекта. Постоянно отслеживать эволюцию функций и требований, а также их реализацию позволяет ***верификация*** (см. лекцию 13). Ее поддержка осуществляется путем использования методов ***трассировки,*** что позволяет связывать друг с другом части проекта, проводить трассировку требований к прецедентам и функциям и обратно. С помощью трассировки можно удостовериться в том, что:

— все элементы требований проекта учтены;

— все реализованные элементы проекта служат заданной цели и требованиям.

Хотя верификация является аналитическим методом, рекомендуется помнить о важности размышлений и интуиции. Нельзя ограничиваться механическим применением верификации. Основное внимание при ее проведении уделяется тестированию и использованию трассировки для отбора компонентов системы, нуждающихся в тестировании. ***Методы проверки корректности требований*** призваны гарантировать, что:

— все элементы требований могут надлежащим образом и достаточно полно тестироваться;

— все тесты служат цели проекта и не являются избыточными.

Чтобы принять решение о том, какая часть системы нуждается в верификации и проверке корректности и в каком объеме, целесообразно применять ***анализ и оценку рисков.*** Это позволяет определить, для каких элементов неправильная реализация требований недопустима, а также разработать план действий по верификации и проверке правильности, основываясь на результатах этих оценок. На этом этапе следует привлекать к управлению требованиями и анализу их корректности специалистов по тестированию, подключая их к планированию тестов с самого начала проекта (см. лекцию 13). Группа тестирования должна разработать тестовые процедуры и сценарии, которые трассируются к прецедентам, а также функциональным и конструктивным требованиям.

Построение ***корректных требований к системе*** во многом зависит от надлежащей обработки изменений. Изменения — неотъемлемая часть жизни ПС, это нужно учитывать ***при создании планов,*** а также необходимо разработать процесс, с помощью которого можно управлять изменени-

ями требований. Управление изменениями дает уверенность в том, что создаваемая система ***является корректной*** и, более того, ***будет правильной*** и в дальнейшем. Специалистам по независимой гарантии качества должна отводиться роль в отслеживании и оценке процесса управления требованиями и плана их тестирования, а также периодических испытаний по окончании значительных этапов, чтобы проверить правильность результатов работы и обеспечить постоянное участие заказчиков.

6.2. Процессы разработки требований к характеристикам сложных программных средств

При планировании, разработке и реализации ***требований к характеристикам качества ПС*** необходимо в первую очередь учитывать следующие основные факторы:

— функциональную пригодность (функциональность) конкретного проекта ПС;

— возможные конструктивные характеристики качества комплекса программ, необходимые для улучшения функциональной пригодности;

— доступные ресурсы для создания и обеспечения всего жизненного цикла ПС с требуемым качеством.

Эти факторы целесообразно применять последовательно, итерационно на каждом из основных этапов ЖЦ ПС. При первоначальном определении требований к функциональной пригодности и к конструктивным характеристикам качества заданные заказчиком ограничения ресурсов не всегда могут учитывать ряд особенностей проекта, что обусловит недопустимое снижение (или завышение) требований к некоторым характеристикам качества ПС. Кроме того, возможно, что некоторые характеристики противоречивы или принципиально нереализуемы в данном проекте. В результате ***несбалансированные требования*** к качеству и доступные ресурсы проявятся как риски — ущерб в виде потерь в качестве или в перерасходовании ресурсов. Для устранения или снижения рисков до допустимых пределов потребуется изменение требований к функциональной пригодности и/или к конструктивным характеристикам. Таким образом, целесообразно анализ и разработку требований к качеству ПС проводить в ***два этапа***: предварительно максимизируя функциональную пригодность

и конструктивные характеристики качества, а затем минимизируя риски снижения требуемого качества или используемых ресурсов.

Разработку и утверждение требований к характеристикам и атрибутам качества без учета рисков целесообразно проводить ***итерационно на этапах системного и детального проектирования ПС***. Полная и однократная формализация требований к характеристикам качества в начале жизненного цикла сложного ПС обычно невозможна, прежде всего, из-за разных представлений заказчика и разработчиков о деталях его назначения, функций и возможностей реализации при доступных ресурсах. Чем крупнее и сложнее проект ПС и соответственно выше его стоимость, тем тщательнее следует разрабатывать требования к его характеристикам качества и распределять ресурсы на их реализацию (см. лекцию 12). Поэтому при средней и относительно невысокой сложности ПС во многих случаях можно удовлетвориться подготовкой требований к качеству с подробностью анализа, соответствующей системному проектированию. Для крупномасштабных и особо сложных проектов необходимы более детальный анализ факторов при разработке требований и их оптимизация по критерию качество/затраты. Поэтому ***на этапах проектирования последовательно должны определяться требования***:

— при проектировании концепции — предварительные требования к назначению, функциональной пригодности и к номенклатуре необходимых конструктивных характеристик качества ПС;

— при предварительном проектировании — требования к шкалам и мерам применяемых атрибутов характеристик качества с учетом общих ограничений ресурсов;

— при детальном проектировании — подробные требования к атрибутам качества с детальным учетом и распределением реальных ограниченных ресурсов, а также, возможно, их оптимизация по критерию качество/затраты.

В зависимости от сложности проекта окончательным результатом работ при предварительном или детальном проектировании должны быть детализированные и утвержденные требования к функциям, свойствам и значениям атрибутов качества ПС, которые в обоих случаях достаточны для его полноценного рабочего проектирования и последующей, эффективной эксплуатации (без учета рисков). В реальных проектах при подго-

товке требований к качеству ПС могут исключаться этапы проектирования, однако содержание и последовательность работ по анализу и детализации требований к характеристикам и атрибутам качества целесообразно сохранять. Эти ***требования закрепляются в контракте и техническом задании***, по которым разработчик впоследствии должен отчитываться перед заказчиком при завершении проекта или его версии. Однако на последующих этапах жизненного цикла и при конфигурационном управлении требования могут изменяться по согласованию между заказчиком и разработчиком, которые чаще всего приурочиваются к подготовке новой версии программного продукта. Для этого необходим мониторинг требований и реализаций характеристик качества в течение всего ЖЦ ПС.

Выбор требований к характеристикам и атрибутам качества при проектировании программных средств начинается с ***определения исходных данных***. Для корректного выбора и установления требований к характеристикам качества, прежде всего, необходимо ***определить особенности проекта***:

— класс, назначение и основные функции создаваемого ПС;

— комплект стандартов и их содержание, которые целесообразно использовать при выборе характеристик качества ПС;

— состав потребителей характеристик качества ПС, для которых важны соответствующие атрибуты качества;

— реальные ограничения всех видов ресурсов проекта.

Этап разработки концепции проекта целесообразно начинать с формализации и обоснования набора исходных данных, отражающих общие особенности класса, потребителей и этапов жизненного цикла проекта ПС, каждый из которых влияет на выбор определенных характеристик качества комплекса программ. Из конструктивных характеристик и атрибутов качества, прежде всего, следует выделять те, на которые в наибольшей степени воздействуют ***класс, назначение и функции ПС***. Для этого первоначально целесообразно использовать классификацию программных средств по стандарту ISO **12182** и всю базовую номенклатуру характеристик и субхарактеристик качества, ***стандартизированных в*** **ISO 9126** (см. лекцию 11). Их описания желательно предварительно упорядочить по приоритетам с учетом особенностей назначения и сферы применения конкретного ПС. Обычно наиболее сильное влияние функции ПС оказывают

на требования к атрибутам характеристик: защищенность, надежность и эффективность. В то же время, характеристики сопровождаемость и мобильность относительно слабо связаны с назначением и конкретной функциональной пригодностью ПС для оперативных пользователей. Их меры и шкалы определяются не столько конкретными функциями комплекса программ, сколько его архитектурой и приспособленностью интерфейсов к модификации и переносу на иные операционные и аппаратные платформы.

Требования стандартов к функциональной пригодности должны выполняться для любых классов и назначения ПС. Однако номенклатура учитываемых требований к конструктивным характеристикам качества существенно зависят от назначения и функций комплексов программ. Так, например, при проектировании:

— систем управления объектами в реальном времени наибольшее значение имеет защищенность, корректность и надежность функционирования стабильного комплекса программ и менее важно может быть качество обеспечения сопровождения и конфигурационного управления, способность к взаимодействию и практичность;

— административных систем, кроме корректности, важно обеспечивать практичность применения, комфортное взаимодействие с пользователями и внешней средой и может не иметь особого значение эффективность использования вычислительных ресурсов и обеспечение мобильности комплекса программ;

— операционных систем наиболее жесткие требования предъявляются к корректности, эффективности использования вычислительных ресурсов и защищенности, и не всегда могут учитываться мобильность и унификация способности к взаимодействию ее компонентов;

— пакетов прикладных программ для вычислений или моделирования процессов возможно не учитывать их мобильность, защищенность и временную эффективность, но особенно важна корректность.

Далее необходимо ***выделить и ранжировать по приоритетам потребителей***, которым необходимы определенные показатели качества ПС с учетом их специализации и профессиональных интересов. Широкая номенклатура характеристик, представленная в стандарте **ISO 9126** определяет разнообразные требования, из которых следует селектировать и выбирать те, которые необходимы ***с позиции потребителей этих данных***:

— заказчиков, для которых важно регламентировать и оценивать ПС по значениям требований к характеристикам, заданным и утвержденным в контракте, техническом задании и спецификациях требований и определяющих, прежде всего, назначение, функции и сферу применения программного продукта;

— пользователей, для которых необходима функциональная пригодность, корректность, надежность и другие показатели качества при оперативном использовании комплекса программ по основному назначению;

— разработчиков, для которых особенно важны: ясность и конкретность описаний требований к функциям и характеристикам ПС, его возможная архитектура и интерфейсы между компонентами и с внешней средой;

— специалистов, сопровождающих и модифицирующих ПС, которые отдают приоритет характеристикам, поддерживающим сопровождение и конфигурационное управление версиями комплекса программ и его компонентов;

— лицам, ответственным за инсталляцию и реализацию программного продукта в различных аппаратных и операционных средах, для которых наиболее важны характеристики мобильности.

Таблица 6.1

Пример ранжирования важности характеристик программного средства для различных категорий специалистов

	Функциональные возможности	Надежность	Эффективность	Практичность	Сопровождаемость	Мобильность
Заказчики	Высокая	Высокая	Высокая	Высокая	Средняя	Средняя
Пользователи	Высокая	Высокая	Высокая	Высокая	Низкая	Низкая
Разработчики	Высокая	Высокая	Средняя	Средняя	Средняя	Средняя
Сопровождающие	Средняя	Средняя	Средняя	Высокая	Высокая	Низкая
Специалисты по переносу	Высокая	Средняя	Высокая	Средняя	Низкая	Высокая

В таблице 6.1 представлен ***пример ранжирования*** по степени важности на три уровня (высокая, средняя, низкая) основных стандартизированных характеристик качества ПС для разных категорий специалистов. Первые две группы потребителей характеристик качества заинтересованы преимущественно в реализации функций, проявляющихся в процессе использования конечного, готового программного продукта. Для этих потребителей при выборе важно выделять и по возможности формализовать внешние, эксплуатационные характеристики на завершающих этапах ЖЦ версий ПС. К ним относятся, прежде всего, высокие приоритеты для надежности, эффективности и практичности. Для заказчиков приоритетными могут быть также сопровождаемость и мобильность, которые для оперативных пользователей ПС обычно являются второстепенными.

Последние три группы потребителей интересуют преимущественно характеристики ПС на промежуточных этапах ЖЦ, на которых проявляются в основном внутренние структурные, технологические свойства комплекса программ, влияющие также на сопровождаемость и мобильность. Их можно не представлять в составе эксплуатационной документации для оперативных пользователей и отражать только в технологической документации разработчиков, специалистов по сопровождению и переносу программ и данных, а также поставлять заказчикам по специальному запросу. Они должны формализоваться для обеспечения возможности контроля системой качества предприятия или проекта, а также для технологической поддержки реализации этих характеристик качества в течение всего ЖЦ ПС. Для этих потребителей надежность и практичность отходят на второй план, однако ресурсная эффективность может оставаться высоко приоритетной.

Приоритеты потребителей при выборе требований к качеству отражаются не только на выделении важнейших для них критериев и ранжировании приоритетов других характеристик, но также на возможности исключения из анализа некоторых характеристик качества, которые для данного потребителя не имеют значения. Представленное в таблице 6.1 ранжирование может детализироваться и изменяться в зависимости от функций ПС и реальных ресурсов, доступных для обеспечения их жизненного цикла. Эти приоритеты должны обобщаться и учитываться заказчиком проекта при подготовке контракта и технического задания. После определения

назначения и функций ПС подготовка исходных данных и концепции проекта должны завершаться ***выделением номенклатуры приоритетных конструктивных атрибутов характеристик качества***, имеющих достаточное влияние на функциональную пригодность ПС для определенных потребителей.

На ***этапе предварительного проектирования***, после фиксирования исходных данных и приоритетов характеристик для конкретного проекта и его потребителей, может начинаться выбор требований к свойствам и значениям, а также установление и утверждение конкретных мер и шкал характеристик и атрибутов качества. Такой анализ должны проводить заказчик и некоторые потенциальные пользователи совместно со специалистами, обеспечивающими ЖЦ комплекса программ и реализацию установленных требований к показателям качества. Этими специалистами, для каждого из выбранных атрибутов, должна быть установлена и согласована мера и шкала оценок характеристик и их атрибутов для конкретного проекта и потребителя результатов анализа. Там, где кроме оценивания наличия номинальных свойств возможны количественные измерения, при оценке реализации требований к качеству может быть полезен выбор и установление допусков на отклонения от величин, требуемых спецификациями. Для показателей, представляемых качественными свойствами и признаками их наличия, желательно определить и зафиксировать в спецификациях описания допустимых условий, при которых следует считать, что данная характеристика может или должна быть реализована в проекте ПС.

Принципиальные и технические возможности, точность реализации свойств и измерения значений характеристик качества, а также ***общие ресурсы конкретного проекта*** всегда ограниченны в соответствии с их содержанием и возможностями заказчика и разработчиков. Это определяет рациональные диапазоны значений каждого атрибута, которые могут быть выбраны для проекта ПС ***на основе требований заказчика***, здравого смысла, а также путем анализа пилотных проектов и прецедентов в спецификациях требований реализованных проектов. В пределах этих диапазонов целесообразно определение реальной достоверности оценивания и масштаба мер для описания соответствующего атрибута требований.

Требования к значениям характеристик качества должны быть предварительно проверены разработчиками на их реализуемость с учетом дос-

тупных ресурсов конкретного проекта и при необходимости откорректированы по составу и значениям с учетом рисков. При ограниченности ресурсов проекта ПС распределение приоритетов должно становиться более строгим и могут снижаться приоритеты характеристик, для реализации которых ресурсов недостаточно. В результате формируется полный ***набор требуемых характеристик, свойств и атрибутов, их мер и значений качества для определенных потребителей в ЖЦ ПС***.

Входные проектные данные и требования к ПС, включая установленные законодательные и регламентирующие нормативные требования, должны быть оформлены документально, а их выбор проанализирован поставщиком на адекватность. Неполные, двусмысленные или противоречивые требования должны быть предметом урегулирования с лицами, ответственными за их предъявление. Спецификацию требований должен представить потребитель-заказчик. Однако по взаимному согласию ее может подготовить поставщик-разработчик в тесном сотрудничестве с потребителем для предупреждений разногласий путем, например, уточнения определений терминов, объяснения предпосылок и обоснования требований. Исходные требования могут быть представлены и согласованы в виде спецификации всей системы. Если программный продукт нуждается во взаимодействии с другими программными или аппаратными продуктами, то в спецификации требований должны быть оговорены непосредственно или при помощи ссылок интерфейсы между разрабатываемыми и другими применяемыми продуктами. В этом случае должны быть разработаны процедуры, обеспечивающие четкое распределение системных требований между аппаратными и программными компонентами, а также соответствующие спецификации интерфейсов с внешней средой. При заключении контракта спецификация требований может быть определена не полностью, она может быть доработана в ходе реализации проекта.

Выходные проектные данные должны быть документально оформлены и выражены в требованиях так, чтобы их можно было проверять в ЖЦ ПС и подтвердить соответствие входным проектным требованиям. Выходные проектные данные должны содержать критерии приемки проекта заказчиком или ссылки на них, а также идентифицировать те характеристики проекта, которые являются критическими для безопасного и надежного функционирования ПС. К составу выходных проектных данных могут относиться:

— спецификация структурного проектирования;

— описание результатов и спецификации рабочего проектирования компонентов;

— исходные тексты программ и программы в объектном коде;

— комплект эксплуатационной документации и руководств для пользователей;

— комплект технологической документации для обеспечения возможности модификации и сопровождения ПС.

Результаты системного анализа и выбора номенклатуры и мер характеристик проектов ПС средней или относительно невысокой сложности должны быть документированы в спецификациях требований, согласованы с их потребителями и утверждены заказчиком проекта для последующей реализации. Для разработчиков особенно важно ***формализовать требования к качеству и согласовать их с заказчиком при утверждении контракта и технического задания на проект ПС***. Требования к характеристикам, утвержденные после предварительного проектирования, могут быть закреплены в техническом задании ***как обязательные для детального и рабочего проектирования***. Эти данные могут использоваться при последующем оценивании качества и его сопоставлении с требованиями в процессе квалификационных испытаний или сертификации ПС. Однако для крупных и дорогих проектов может потребоваться уточнение требований к качеству при детальном проектировании с позиции улучшения соотношения значений качества и затрат ресурсов, которые необходимы или допустимы для их реализации в ЖЦ ПС.

При детальном проектировании может быть целесообразно дополнительное уточнение совокупности требований выбранных характеристик и атрибутов качества сложных ПС с учетом соотношения качества и затрат ресурсов, которые могут быть весьма велики. Для заказчика и пользователей может иметь значение не только определение функциональной пригодности, но и потенциального спроса на рынке конкретного программного продукта, а также его конкурентоспособности с другими аналогичными по функциям ПС с учетом его качества и стоимости. Это обстоятельство может определять необходимость уточнения требований к отдельным характеристикам качества не только для их реализации разработчиками в ЖЦ ПС, но также для оценивания интегрального качества готового программного продукта, поставляемого на рынок.

Каждая характеристика качества и затраты ресурсов первоначально ***анализируются независимо***, что может использоваться в качестве исходных данных для их сопоставления с отдельными характеристиками аналогичных ПС или для представления, как составляющей вектора в многомерном пространстве стандартизированных атрибутов характеристик качества. Обычно заказчики и разработчики первоначально устанавливают требования к каждой характеристике без учета относительных затрат на их достижение, а также без детального анализа их совместного влияния на полную функциональную пригодность у потребителей. Это может приводить к значительным перекосам и ***несбалансированным значениям требований*** к отдельным, взаимосвязанным характеристикам качества, на которые нерационально используются ограниченные ресурсы ЖЦ ПС, или к неадекватно низким их значениям. В проектах крупных ПС это может угрожать значительным повышением стоимости и/или снижением конкурентоспособности создаваемого программного продукта из-за недостаточного уровня отдельных показателей качества.

Атрибуты качества ПС имеют различные меры и шкалы, вследствие чего они в большинстве своем непосредственно ***несопоставимы между собой***. Они предварительно выбираются и согласовываются с заказчиком при последовательном, почти независимом анализе каждого атрибута качества в соответствии с их мерами и шкалами, для последующего использования в контракте и техническом задании. Для обобщенного оценивания качества ПС необходим учет относительного влияния каждого атрибута на функциональную пригодность. При этом не всегда учитываются ресурсы для их реализации в конкретном ПС. Это часто приводит к выдвижению ряда нерациональных требований, которые значительно отличаются: либо по степени влияния на функциональную пригодность, либо по величине ресурсов, необходимых для их реализации. Для целенаправленного эффективного управления качеством сложного ПС при проектировании целесообразно иметь механизм объединения разнородных характеристик в некоторый интегральный показатель, отражающий их совокупное влияние на его функциональную пригодность. Таким образом, при разработке требований к характеристикам качества выявилась проблема анализа системной эффективности программного средства и обобщения его характеристик, а также ***оценивания совместного влияния различных характе-***

ристик и атрибутов качества на функциональную пригодность ПС с учетом затрат на их реализацию (см. лекцию 12).

6.3. Структура основных документов, отражающих требования к программным средствам

При разработке требований к проектам программных средств кроме основных целей, назначения и функций важно учесть и сформулировать содержание достаточно полного множества характеристик, каждая из которых может влиять на успех проекта программного продукта. Для уменьшения вероятности случайного пропуска важного требования заказчикам и пользователям целесообразно иметь типовые проекты перечней (шаблоны) наборов требований, которые можно целеустремленно сокращать и адаптировать, обеспечивая ***целостность требований для конкретных проектов ПС***. Ниже представлены ***примеры*** состава требований на двух этапах жизненного цикла сложных ПС: на этапе формирования концепции ПС и на этапе детального проектирования комплекса программ.

Состав концепции основных требований к программному средству:

— описание обобщенных результатов обследования и изучения существующей системы и внешней среды;

— описание целей, назначения программного продукта и потребностей заказчика и потенциальных пользователей в заданной среде применения;

— перечень базовых стандартов предполагаемого проекта программного продукта;

— общие требования к характеристикам комплекса задач ПС:

- цели создания программного продукта и назначение комплекса функциональных задач;
- перечень объектов среды применения ПС (технологических объектов управления, подразделений предприятия и т. п.), при управлении которыми должен решаться комплекс задач;
- периодичность и продолжительность решения комплекса задач;
- связи и взаимодействие комплекса задач с внешней средой и другими компонентами системы;

• распределение функций между персоналом, программными и техническими средствами при различных ситуациях решения требуемого комплекса функциональных задач;

— требования к входной информации:

• источники информации и их идентификаторы;

• перечень и описание входных сообщений (идентификаторы, формы представления, регламент, сроки и частота поступления);

• перечень и описание структурных единиц информации входных сообщений или ссылка на документы, содержащие эти данные;

— требования к выходной информации:

• потребители и назначение выходной информации;

• перечень и описание выходных сообщений;

• регламент и периодичность их выдачи;

• допустимое время задержки решения определенных задач;

— описание и оценка преимуществ и недостатков разработанных альтернативных вариантов функций в концепции создания проекта ПС;

— сопоставительный анализ требований заказчика и пользователей к программному продукту и набора функций в концепции ПС для удовлетворения требований заказчика и пользователей;

— обоснование выбора оптимального варианта требований к содержанию и приоритетам комплекса функций ПС в концепции;

— общие требования к структуре, составу компонентов и интерфейсам с внешней средой;

— ожидаемые результаты и возможная эффективность реализации выбранного варианта требований в концепции ПС;

— ориентировочный план реализации выбранного варианта требований концепции ПС;

— общие требования к составу и содержанию документации проекта ПС;

— оценка необходимых затрат ресурсов на разработку, ввод в действие и обеспечение функционирования ПС;

— предварительный состав требований, гарантирующих качество применения ПС;

— предварительные требования к условиям испытаний и приемки системы и ПС.

Спецификация требований к системе и к комплексу программ на этапе детального проектирования:

— требования проекта системы к комплексу программ, как к целому в общей архитектуре системы;

— требования к унификации интерфейсов и базы данных комплекса программ;

— требования и обоснование выбора проектных решений уровня системы, состава компонентов системы, описание функций системы и ПС с точки зрения пользователя;

— спецификация требований верхнего уровня комплекса программ, производные требования к компонентам ПС и требования к интерфейсам между системными компонентами, элементами конфигурации ПС и аппаратуры;

— описание распределения системных требований по компонентам ПС с учетом требований, которые обеспечивают заданные характеристики качества;

— требования к архитектуре системы, содержащей идентификацию и функции компонентов системы, их назначение, статус разработки, аппаратные и программные ресурсы;

— требования совместного целостного функционирования компонентов ПС, описание и характеристики их динамических связей;

— требования анализа трассируемости функций компонентов программного средства к требованиям проекта системы;

— требования для системы или/и подсистем и методы, которые должны быть использованы для гарантии того, что каждое требование к комплексу программ будет выполнено и прослеживаемо к конкретным требованиям системы:

- к режимам работы;
- к производительности системы;
- к внешнему и пользовательскому интерфейсу системы;
- к внутреннему интерфейсу компонентов и к внутренним данным системы;
- по возможности адаптации ПС к внешней среде;
- по обеспечению безопасности системы, ПС и внешней среды;
- по обеспечению защиты, безопасности и секретности данных;

- по ограничениям доступных ресурсов проекта ПС;
- по обучению и уровню квалификации персонала;
- по возможностям средств аттестации результатов и компонентов, включающих в себя демонстрацию, тестирование, анализ, инспекцию и требуемые специальные методы для контроля функций и качества конкретной системы или компонента ПС.

Представленный состав спецификации требований на этапе детального проектирования может использоваться как компонент для уточнения технического задания и контракта с заказчиком на проект ЖЦ ПС и служить базой для формирования комплекса отчетных требований, утверждаемых и проверяемых заказчиком при приемке готового программного продукта. Состав стандартизированных характеристик качества программных средств и процессы выбора требований к ним в конкретных проектах представлены в лекциях 11 и 12. Эти требования должны быть отдельным, обязательным разделом в общей спецификации требований, итерационно формируемыми на этапах концепции и проектирования ПС и контролируемыми при испытаниях программного продукта.

ЛЕКЦИЯ 7
ПЛАНИРОВАНИЕ ЖИЗНЕННОГО ЦИКЛА ПРОГРАММНЫХ СРЕДСТВ

7.1. Организация планирования жизненного цикла сложных программных средств

Цель планирования жизненного цикла программного средства состоит в выборе и определении способов создания и совершенствования ПС, которые способны удовлетворить требованиям технического задания, спецификаций и контракта, а также обеспечить уровень качества, соответствующий заданным требованиям. В современных стандартах подчеркивается, что ***эффективное планирование — определяющий фактор высокого качества всего ЖЦ программного средства***, удовлетворяющего требованиям заказчика. В стандартах **ISO 12207** и **ISO 16326** рекомендуется определить администратора, который должен подготовить планы для выполнения процессов ЖЦ ПС. Планы, связанные с выполнением процесса, должны содержать описания соответствующих работ и задач, обозначения создаваемых компонентов программного средства и охватывать ***следующие задачи***:

— установление графиков своевременного решения частных задач и всего ПС;

— оценки необходимых трудозатрат на задачи и проект в целом;

— определение ресурсов, необходимых для выполнения задач и проекта;

— распределение задач по исполнителям;

— определение обязанностей исполнителей;

— определение критических ситуаций, связанных с задачами или процессами ЖЦ ПС;

— установление используемых в процессах ЖЦ ПС критериев управления качеством;

— определение затрат, связанных с реализацией каждого процесса;

— обеспечение условий и определение инфраструктуры выполнения процессов ЖЦ ПС.

Должны быть определены ***обязанности специалистов по подготовке и утверждению (согласованию) планов***. Следует установить модель жизненного цикла программного средства, задачи, распределение задач, их блокировку и соответствующие ресурсы. В программном проекте должен быть определен один основной график работ, а все вспомогательные графики должны быть связаны и согласованы с основным графиком. С помощью структуры классификации работ можно эффективно проверять ход процессов и обеспечивать контроль этих процессов и продуктов. План должен быть применен так, чтобы обеспечить управление программным проектом на всех уровнях его детализации с использованием соответствующих технологий в зависимости от объема, сложности, критичности и риска проекта. ***Оценки проекта***, используемые при планировании, должны охватывать:

— стоимость реализации соответствующих процессов;

— инфраструктуру обеспечения реализации процессов;

— потребности в ресурсах, включая соответствующее управление и контроль;

— оценку и контроль качества реализации процессов;

— управление риском результатов процессов;

— обеспечение среды программной инженерии проекта ПС;

— задания, выполняемые в каждом процессе и (или) работе.

Администраторы каждого программного проекта должны стремиться по возможности использовать существующую ***организационную инфраструктуру предприятия***. Если существующая инфраструктура не удовлетворяет потребностям конкретного проекта, тогда она должна быть соответствующим образом адаптирована или дополнена. Для устранения несовершенства (неполноты) существующей инфраструктуры может потребоваться использование субподрядных работ.

Планирование является постоянной, повторяющейся работой, при выполнении которой оценивают, уточняют и, при необходимости, коррек-

тируют ход проекта. Администраторы программного проекта должны использовать соответствующие процессы, способствующие проведению перепланирования и уточненных оценок в жизненном цикле программного проекта. В каждом программном проекте имеется множество взаимосвязей, поэтому для уточнения исходного плана управления проектом обычно требуется несколько итераций. При необходимости внесения в план управления информации, содержащейся в других планах, в данном плане может быть приведена ссылка на эти планы.

В стандарте **ISO 15504** расширены, детализированы задачи и виды деятельности, которые следует отражать в ***плане управления проектом ПС***:

— выбрать модель жизненного цикла программных средств, соответствующую назначению, функциям, величине и сложности проекта;

— определить работы, которые необходимо выполнить по проекту, и возможность достижения целей проекта в рамках существующих ресурсов и ограничений;

— оценить варианты достижения целей проекта и определить, на основе анализа рисков, какая стратегия целесообразна;

— количественно оценить сложность работ и ресурсы, необходимые для их выполнения, рассматривая варианты достижения целей проекта и принимая во внимание существующие риски и возможности, чтобы весь жизненный цикл ПС удовлетворял требованиям заказчика;

— выявить и выбрать элементы человеческих и материальных ресурсов, необходимые для обеспечения и реализации стратегии проекта;

— установить график (исполнители, сроки, ресурсы) выполнения проекта, основываясь на распределении работ, оценках и элементах инфраструктуры;

— выявить конкретных лиц и группы, дающие требуемый вклад в проект, определить им конкретные зоны ответственности и обеспечить то, чтобы обязанности были поняты и приняты, профинансированы и достижимы;

— идентифицировать интерфейсы между элементами проекта, а также с другими проектами и организационными единицами системы;

— определить инструментарий для обеспечения того, чтобы планы проекта были формально разработаны, реализованы, поддержаны и дос-

тупны лицам, вовлеченным в проект, обеспечить публикацию планов для специалистов, к которым они относятся;

— использовать упорядоченные методы для того, чтобы регулярно оценивать степень выполнение проекта, принимать меры для корректировки отклонений от плана и предотвращения повторения проблем, выявленных в проекте.

Поставщик-разработчик ПС должен подготовить ***планы по каждому виду деятельности.*** Следует установить и поддерживать в рабочем состоянии документированные процедуры, гарантирующие разработку программного продукта в соответствии с заданными требованиями и согласно плану развития ЖЦ. Такие планы должны описывать виды деятельности или содержать ссылки на разделы стандартов и определять ответственность за их осуществление.

Планирование жизненного цикла ПС должно определить действия по анализу требований, проектированию, программированию, интегрированию, тестированию, установке и поддержке программных средств с целью их принятия и применения заказчиком. Жизненный цикл следует оформить документами, которые необходимо проанализировать на реализуемость и утвердить. Они должны актуализироваться по мере развития проекта. В планах необходимо определить, каким образом следует управлять проектом, контролировать и анализировать выполнение работ, а также установить вид и частоту отчетов для руководства, заказчика и других заинтересованных сторон, принимая во внимание все конкретные требования заказчика. В планах должны определяться организационные подразделения и специалисты, которые будут исполнять различные виды работ. Для этого ***планы должны содержать следующую информацию***:

— роли и обязанности соответствующих субъектов проекта;

— подлежащие выполнению работы и задачи;

— перечень всех проектных результатов (продуктов), подлежащих поставке и определенных в структуре классификации работ;

— критерии завершения соответствующей деятельности, работ, задач;

— состав окончательных отчетных документов;

— отчетные документы по стоимости и графикам проведения работ;

— содержание средств организации работ по управлению, выпуску продукта и/или синхронизации работ;

— периодичность и средства выдачи отчетных документов;

— отчетные материалы по проблемам-дефектам или выполнению деятельности;

— требования к ресурсам и их наличие.

Стандартами **ISO 16326** и **ISO 90003** рекомендуется в процессе планирования ЖЦ ПС ***подготовить и утвердить содержание следующих планов*** (рис. 7.1):

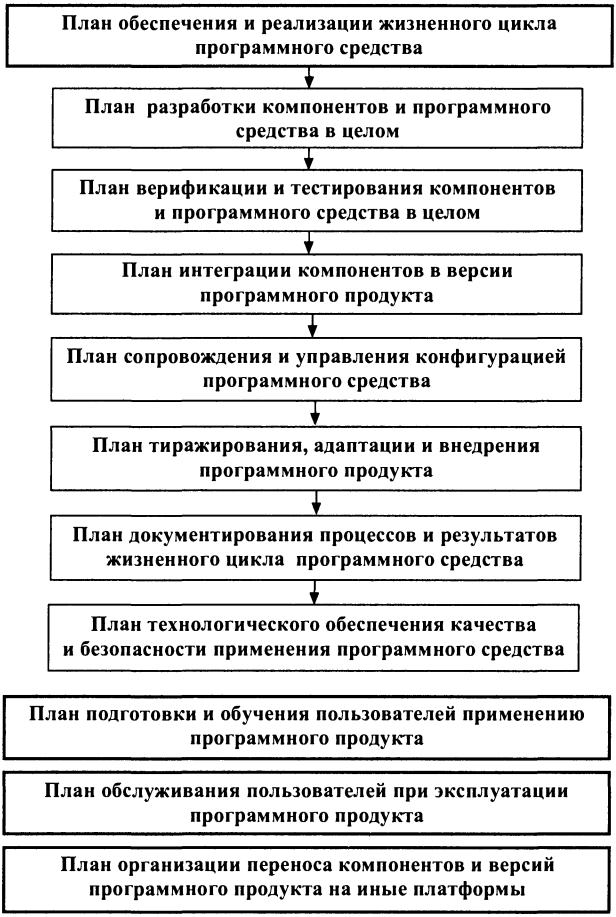

Рис. 7.1

— разработки компонентов и всего ЖЦ ПС, который должен определять используемую модель жизненного цикла комплекса программ и его компонентов, а также внешнюю технологическую среду проектирования;

— верификации и тестирования, который определяет методы и средства, способные удовлетворить последовательные цели процесса устранения дефектов и контроля качества ПС и его компонентов;

— реализации процессов интеграции компонентов в версии комплекса программ;

— сопровождения и управления конфигурацией ПС, который должен устанавливать методы и средства, при помощи которых будут удовлетворяться цели процесса совершенствования, управления изменениями и корректировками программ;

— тиражирования, адаптации и внедрения версий ПС для конкретных пользователей, включая их подготовку и обучение;

— документирования процессов и результатов жизненного цикла ПС, создания и выпуска технологической и эксплуатационной документации;

— управления и обеспечения качества ПС, определяющих методы и средства, при помощи которых будет гарантировано требуемое качество комплекса программ.

Соответствующие планы должны быть разработаны администраторами вспомогательных процессов, так как эти процессы обычно являются частью проекта. Данные планы должны быть ***привязаны к базовому плану управления жизненным циклом программного проекта*** и обеспечивать его реализацию; они могут быть оформлены в виде отдельных планов или включены в общий план. Планы должны быть согласованы (утверждены) менеджером проекта программного средства и подлежат контролю при внесении изменений в проект. Менеджером-администратором планирования программного средства должна быть определена отчетность по вспомогательным процессам (либо непосредственная, либо через управление организацией). Должны быть представлены отчеты о проблемах-дефектах и исключительных ситуациях для анализа их влияния на стоимость проекта, график работ по нему, область управления проектом и его качество. Должен быть определен механизм для разрешения или преодоления конфликтных ситуаций между администратором планирования ЖЦ программного средства и администраторами вспомогательных процессов на соответствующем уровне их полномочий по организационному управлению.

Когда определенные контрольные точки проекта и результаты, установленные требованиями для этих точек, зависят от выходных результатов вспомогательного процесса, важно, чтобы отчетные материалы по ним были представлены точно и своевременно в соответствии с установленными планами. Это положение является общим для всех контрольных точек, связанных с выполнением договорных обязательств по вспомогательным процессам, поэтому необходимы синхронизация соответствующих планов и своевременное уведомление администратора программного средства о всех затруднениях, возникающих при выполнении соответствующих задач вспомогательных процессов. Синхронизация всех планов может быть затруднена при наличии субподрядных соглашений и заданий, но упрощена при наличии единого базового плана.

7.2. Задачи планов для обеспечения жизненного цикла сложных программных средств

План управления жизненным циклом ПС включает регламентированные стандартами процедуры анализа, проверки и оценки состояния проекта для реализации организационных и управляющих воздействий на компоненты и ПС в целом. При этом должны использоваться техническое задание и спецификации требований, в которых определяющую роль играют согласованные с заказчиком входные и выходные данные проекта ПС. Потребитель-заказчик может иметь определенные обязанности согласно контракту при формировании ЖЦ ПС. Особого внимания потребителя требует ***сотрудничество с разработчиком-поставщиком***, своевременное предоставление ему нужной информации и решение оперативных вопросов для обеспечения качества ПС. Если представитель заказчика обладает соответствующей компетентностью, то он может представлять конечного пользователя продукта, а также административное руководство проектом и иметь полномочия заниматься контрактными вопросами. К ним относятся определение и уточнение требований спецификаций потребителя к поставщику и к показателям качества ПС, одобрение и утверждение предложений разработчика, а также заключение дополнительных соглашений с поставщиком. Целесообразно планировать либо регулярно проводить разработчиком и заказчиком ***совместные анализы состояния***

проекта либо такие анализы в случае значительных проектных событий, чтобы охватить:

— состояние и развитие выполняемых поставщиком работ по разработке и модификации программного средства;

— соответствие результатов разработки, согласованной спецификации требований заказчика;

— состояние работ, касающихся подготовки и обучения конечных пользователей разрабатываемой системы;

— результаты проверок текущего состояния проекта и приемочных испытаний.

План разработки компонентов и ПС в целом (см. рис. 7.1) должен включать назначение, стандарты и описание фрагментов жизненного цикла, которые следует использовать в процессе разработки:

— предварительное описание процессов стандартизированного ЖЦ ПС и его компонентов, которые должны использоваться для формирования конкретного жизненного цикла данного проекта, включая критерии перехода между этапами разработки;

— идентификацию фрагментов стандартов: на требования к ПС; описание проекта; кодирование программ для данного проекта; а также ссылки на стандарты для ранее разработанных компонентов, включая используемые готовые апробированные компоненты ПС;

— обоснование выбора инструментальной среды разработки ПС в аппаратной и программной части, которые будут использоваться, включая выбор языков программирования, средств кодирования, компиляторов, редакторов связей и загрузчиков.

План верификации и тестирования ПС является предварительным описанием организации процедур тестирования, удовлетворяющих цели достижения заданной корректности программ. Данный план должен включать:

— описание методов, которые будут использоваться на каждом этапе, а также для обеспечения независимости верификации и тестирования;

— распределение организационной ответственности внутри процессов тестирования и интерфейсы с другими процессами жизненного цикла ПС;

— описания оборудования для анализа и тестирования, инструментальных средств, а также руководств по применению этих средств и аппаратного тестового оборудования;

— описание методов идентификации компонентов, на которые оказывает воздействие модификация ПС, и измененные части исполняемого объектного кода.

План сопровождения и управления конфигурацией ПС устанавливает методы, используемые для сопровождения программных средств и их компонентов в течение всего жизненного цикла. Этот план должен включать:

— описание процессов управления конфигурацией в жизненном цикле ПС, которые обеспечат выполнение задач сопровождения;

— описание среды управления конфигурацией, которую следует использовать, включая процедуры, инструментальные средства, методы, стандарты, организационные соглашения и интерфейсы;

— идентификацию отчетов о дефектах и ошибках программного продукта и процессов жизненного цикла; метод закрытия отчетов об ошибках и взаимодействия с контролем изменений;

— архивацию версий; применение методов и средств формирования версий и обеспечения их сохранности.

Цель ***планирования технологической среды жизненного цикла ПС*** состоит в том, чтобы определить методы, инструментальные средства, процедуры, языки программирования и аппаратные средства, которые будут использоваться и совершенствоваться для разработки, верификации, управления и подготовки документации программного средства. План должен включать стандарты, методы предотвращения ошибок и обеспечения отказоустойчивости, которые ограничивают возможность внесения ошибок, и такие методы тестирования, которые гарантируют их обнаружение. Цель методов обеспечения отказоустойчивости состоит в том, чтобы включить в проект такие средства обеспечения качества и безопасности применения ПС, которые могут гарантировать, что программное средство будет адекватно реагировать на ошибки входных данных, предотвращать выдачу ошибочных данных и контролировать возможность проявления ошибок.

Кроме того, в составе перечисленных планов или автономно может быть полезной ***разработка ряда вспомогательных планов*** (см. рис. 7.1):

— плана подготовки и обучения пользователей для квалифицированной эксплуатации версий ПС;

— плана обслуживания пользователей в процессе эксплуатации ПС;

— плана организации переноса и установки версий ПС на различные аппаратные и операционные платформы пользователей.

Все перечисленные ***планы должны иметь в своем составе*** предварительные графики, идентифицирующие: этапы работ; входные, выходные данные и описания решаемых задач; необходимые ресурсы, длительность и сроки выполнения; взаимосвязи этапов и работ. Должно быть установлено организационно-техническое взаимодействие между различными группами специалистов, которые вносят свой вклад в процессы и обеспечение качества ЖЦ ПС, а необходимая информация должна документироваться и регулярно анализироваться. В планах работ поставщиков и субподрядчиков при проектировании следует четко определить границы ответственности за каждую часть программного средства и за способ обмена технической информацией между всеми сторонами проекта. При установлении этого взаимодействия следует обратить внимание на вспомогательных специалистов, помимо потребителя и поставщика, которым необходимы входные данные для проектирования, установки, обслуживания и подготовки программ и данных.

Анализ состояния и результатов проекта следует официально планировать и регулярно проводить на соответствующих этапах ЖЦ ПС. В состав участников каждого анализа должны включаться представители всех служб, заинтересованных в результатах анализируемого этапа проектирования. Степень формальности и жесткости действий по осуществлению анализа должна соответствовать сложности ПС и степени риска, связанного с областью применения и использованием программного средства. В процедуры анализа проекта могут входить действия, которые необходимо предпринять до и во время его проведения, включая используемые методики и руководящие указания для всех участников проверок. Если оговорено в контракте, поставщик должен проводить совещания по анализу результатов и состояния проекта вместе с заказчиком. Обе стороны должны согласовать результаты подобных анализов. Проверку проекта на соответствующих этапах ЖЦ ПС надо проводить, чтобы удостовериться, что выходные данные на этих этапах соответствуют входным требованиям.

В проверку проекта могут входить: анализ выходных проектных данных; демонстрации прототипов и моделей, а также результатов их тести-

рования. Эти проверки следует проводить согласно документированным процедурам и планам. Перед тем как предложить продукт потребителю на испытания и приемку, поставщик должен оценить его качество согласно назначению и спецификациям требований, например, на этапе окончательного внутреннего контроля и испытаний разработчиком. При разработке программного средства важно, чтобы результаты оценки и любые последующие действия, необходимые для выполнения заданных требований, были зафиксированы и проверены по завершении этих работ.

7.3. Планирование процессов управления качеством сложных программных средств

При проектировании ПС ***планирование процессов управления качеством ПС целесообразно отделять*** от непосредственного планирования и управления процессами создания и совершенствования крупных комплексов программ. Это позволяет сосредоточить внимание выделенных специалистов на совокупности мероприятий, гарантирующих качество конечного продукта. Чтобы такие гарантии достигались при минимальных затратах, необходимы целенаправленное, координируемое планирование и управление для предотвращения ошибок проектирования, а также для выявления и устранения дефектов проекта на самых ранних этапах разработки. Поэтому мероприятия планирования, обеспечивающие качество программ, должны охватывать не только завершающие испытания, но и весь жизненный цикл ПС.

Соответственно должны выделяться и применяться методы и средства автоматизации процессов планирования и управления качеством по этапам разработки компонентов и ПС в условиях реальных ограниченных ресурсов. Следует учитывать ***глубокую взаимосвязь*** плана управления и применения процедур ***обеспечения качества ПС*** с планом управления ***непосредственной разработкой программ***, с процессами тестирования, испытаний и сертификации ПС. Эта связь выражается в аналогичном поэтапном представлении планов и в наличии в них значительной части близких по содержанию процессов и документов. При планировании процессов обеспечения качества ПС целесообразно учитывать и использовать совокупность рекомендаций ряда стандартов, в которые входят —

ISO 10005, ISO 10006, ISO 10013, поддерживающие базовые стандарты менеджмента качества серии **ISO 9000** (см. лекцию 3).

В соответствии со стандартами процедуры управления качеством должны применяться к различным документам и данным, включая: контрактные документы; процедурные документы, описывающие процессы системы качества; состояние работ поставщика-разработчика; взаимодействие поставщика и заказчика; документы и данные, которые описывают конкретный программный продукт. В составе документов должны содержаться методики, инструкции и описания по использованию конкретных инструментальных средств и выполнению частных работ и операций в ЖЦ ПС. Для этого в общем случае должны быть выделены руководители и коллектив специалистов, которые должны планировать, создавать, утверждать и сопровождать комплекты документов. Они должны стимулировать разработчиков ПС осуществлять непрерывное, регламентированное документирование процессов и результатов своей деятельности, а также контролировать полноту и качество результирующих и отчетных документов. Стандарты и базовые нормативные документы ЖЦ ПС должны служить верхним уровнем иерархической системы технологических документов, регламентирующих и конкретизирующих все этапы, работы и документы проекта.

Организационной основой управления качеством ПС на базе стандартов ЖЦ является ***план обеспечения заданных характеристик качества*** на всех этапах жизненного цикла комплекса программ (см. лекцию 11). Для этого до начала разработки в процессе формирования требований технического задания следует сформулировать основные положения методики обеспечения качества, поэтапных испытаний компонентов и определения достигнутых значений характеристик, допустимых для продолжения работ на следующих этапах. Такой план целесообразно создавать для сложных проектов ПС на этапах системного анализа, разработки требований технического задания и проектирования. На этих этапах оформляются первичные требования к характеристикам и качеству ПС и, соответственно, должна планироваться совокупность мероприятий, обеспечивающих их достижение в процессе последующего проектирования. ***В плане управления качеством ПС*** должны быть отражены:

— цели управления качеством, номенклатура и требования к значениям характеристик качества, область действия требований и условия их применения;

— методы управления и достижения заданных значений качества; процедуры, которые должны выполняться для каждого процесса и на протяжении всего жизненного цикла ПС; действия, связанные с отчетностью об ошибках, с трассировкой и системой корректирующих действий;

— организация разработчиков и технология создания ПС; утвержденные обязанности специалистов по обеспечению качества, их ответственность и полномочия на утверждение программных компонентов и документов;

— ресурсы, базовые документы и стандарты, используемые для обеспечения качества на всех этапах разработки;

— средства автоматизации разработки, обеспечивающие достижение и измерение заданных свойств и значений характеристик качества;

— структура и содержание отчетных документов, удостоверяющих достижение определенного качества компонентов и ПС в целом на последовательных этапах разработки, а также их соответствие стандартам и требованиям заказчика.

Реальные ограничения ресурсов, используемых в процессе разработки, квалификация специалистов, изменения внешней среды и требований заказчика объективно приводят к отклонениям процессов и реализации плана от предполагавшегося. Величина таких отклонений в значительной степени зависит от принятой технологии разработки, от уровня и характеристик средств автоматизации создания программ. Для своевременного обнаружения отклонений от плана необходимо регулярно регистрировать результаты выполненных работ и их качество. Для реализации таких изменений целесообразно предусмотреть и согласовать с заказчиком специальный документ, регламентирующий ***правила корректировки плана обеспечения качества ПС***, а также состав и содержание поддерживающей его документации. Подобные изменения должны оформляться протоколами и доводиться до сведения всех специалистов, к которым они относятся.

Одна из трудностей достижения высокого качества ПС состоит обычно в отсутствии полной совокупности достоверных требований к значениям характеристик качества на начальных этапах проектирования и разработки, а также итерационный процесс конкретизации требований в течение всего жизненного цикла ПС. В результате первично сформулированные

требования к ***характеристикам качества сложных ПС последовательно уточняются и корректируются в процессе взаимодействия заказчика и разработчика*** с учетом объективно изменяющихся особенностей развивающегося проекта ПС. Для этого необходимо контролировать требуемые характеристики в процессе разработки, анализировать их адекватность целям проекта и управлять их изменениями в нужном направлении. Активное взаимодействие разработчиков с заказчиком или потенциальными пользователями на всех этапах разработки позволяет уточнять, корректировать и детализировать совокупность спецификаций требований и атрибуты качества в соответствии с развитием концепции и ***возрастанием понимания задач проекта как заказчиком, так и разработчиком***.

Управление качеством комплексов программ предполагает формализацию технологии обеспечения их жизненного цикла, а также выделение в специальный процесс поэтапного измерения и анализа текущего качества программных компонентов и проекта в целом. В общем случае ***в процессе планирования и управления качеством ПС*** следует учитывать:

— анализ контракта и спецификаций требований заказчика к ПС, выделение и ранжирование приоритетов характеристик и атрибутов качества конечного продукта;

— декомпозицию требований к характеристикам качества по контролируемым этапам и компонентам разработки и создание разделов по детальным требованиям к атрибутам качества в спецификациях на программные компоненты и ПС в целом;

— выбор или создание методов, технологии и инструментальных средств автоматизации разработки, обеспечивающих создание ПС и его компонентов с требуемыми характеристиками качества;

— разработку методик контроля соблюдения стандартов, правил технологии проектирования и системы обеспечения качества жизненного цикла программных средств;

— создание методов, методик и средств объективного измерения свойств и/или значений атрибутов характеристик качества программных компонентов на этапах их создания и всего ЖЦ ПС для испытаний заказчиком и эксплуатации пользователями;

— организацию, обучение и стимулирование коллектива специалистов на создание компонентов и ПС в целом, в максимальной степени

удовлетворяющих требования заказчиков и пользователей к характеристикам качества.

Чтобы решать эти задачи, коллективы специалистов в процессе планирования проекта должны играть активную роль и осуществлять их выполнение для обеспечения качества на последующих этапах жизненного цикла. Они должны действовать в соответствии с полномочиями, ответственностью и независимостью, чтобы гарантировать удовлетворение целей и требований к качеству ПС. Качество объектов формируется и документируется при выполнении частных работ каждого этапа ЖЦ и окончательно удостоверяется при их завершении. Измерения объектов разработки сводятся к регулярной, поэтапной регистрации и документированию характерных для данного объекта показателей качества, а также к сопоставлению их с заданными требованиями.

В стандарте **ISO 15504** (раздел **MAN.3**) ***процесс планирования и управления качеством ПС*** концентрируется на мониторинге качества продуктов и процессов как проекта, так и предприятия в целом. В результате успешной реализации процессов в проекте должно быть обеспечено:

— на основе явных и неявных требований потребителя необходимо определить цели по качеству продукта и процессов, которые следует оценивать, предпочтительно количественным образом, для различных контрольных точек в жизненном цикле ПС;

— определить общую стратегию на уровне проекта ПС и предприятия, помогающие аттестовать достижение соответствующих целей по качеству, определяя количественную меру результатов деятельности по проекту и критерии их приемки;

— для каждой цели по качеству ПС идентифицировать виды деятельности по контролю и обеспечению качества, помогающие достижению этой цели и ее мониторингу как на уровне проекта, так и на уровне предприятия, и интегрировать эти виды деятельности в жизненный цикл ПС;

— проводить идентифицированные виды деятельности по контролю и обеспечению качества и подтверждать их выполнение;

— по ходу реализации проекта в контрольных точках жизненного цикла ПС применять заданные метрики качества для аттестации достижения соответствующих целей по качеству;

— в случаях, когда заданные цели по качеству не достигаются, применять корректирующие или предотвращающие мероприятия, которые мо-

гут включать либо исправление продукта, либо изменение запланированного набора видов деятельности для лучшего достижения целей по качеству, либо изменение спецификации требований продукта или определения процесса для предотвращения повторения дефектов.

Утверждение и выпуск документации о качестве должны планироваться и контролироваться уполномоченным персоналом на предмет их адекватности. Следует планировать и поддерживать в актуальном состоянии ***процедуры управления документами***, которые идентифицируют текущий статус корректировки и пересмотра документов, с тем чтобы предотвратить использование недействующих и/или устаревших документов:

— наличие актуальных изданий соответствующих документов на всех участках, где проводятся работы, от которых зависит эффективное функционирование системы качества;

— немедленное изъятие недействующих и/или устаревших документов из всех пунктов их рассылки или применения либо принятие других мер по предотвращению их непреднамеренного использования;

— идентификацию любых устаревших документов, составленных для юридических целей и/или для сохранения полезной информации.

Изменения документов должны планироваться и утверждаться теми же службами и/или организациями, которые проводили первоначальный анализ и утверждали эту документацию. Архивные данные по проекту являются основным источником для анализа и осмысления возможностей и эффективности процесса планирования в организации, рабочих характеристик данного проекта и обобщения полученного опыта. Эти архивные данные, образующие общую базу данных предприятия, следует постоянно использовать для усовершенствования процессов жизненного цикла ПС. Та же самая база данных должна обеспечивать реализацию процесса управления по отдельным программным проектам.

ЛЕКЦИЯ 8

ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОЕКТИРОВАНИЕ ПРОГРАММНЫХ СРЕДСТВ

8.1. Задачи и особенности объектно-ориентированного проектирования программных средств

Международные стандарты ***программной инженерии*** (Приложение 1) регламентируют жизненный цикл программных средств широкого класса в различных областях их применения. Они ориентированы в основном на структурное проектирование и разработку процессов, функций и данных в различных комплексах программ, в которых доминируют сложные алгоритмы обработки, относительно небольшой совокупности данных. Объектно-ориентированное проектирование (ООП) предназначено организовывать программные системы с большими базами данных на ***основе описаний объектов*** реального мира, важных для пользователей. Этот подход существенно отличается от структурного проектирования, которое акцентировано ***на сложные функции и процессы*** обработки данных. Объекты реального мира, существующие внутри области действия ООП программных проектов, определяются в терминах целей, характеристик и ответственностей поведения соответствующих данных (атрибутов) и отношений с другими многочисленными объектами. Все ***функции при этом скрыты внутри деталей описаний объекта***.

Объектно-ориентированное проектирование представляет собой ***стратегию***, в рамках которой программная система состоит из взаимодействующих объектов, имеющих собственное локальное состояние и способных

выполнять определенный набор операций, определяемый ***состоянием объекта***. Объекты скрывают информацию о представлении состояний и, следовательно, ограничивают к ним доступ. Под процессом ООП подразумевается проектирование классов объектов и взаимоотношений между этими классами. Объектно-ориентированные системы можно рассматривать как совокупность автономных и в определенной степени независимых объектов. Изменение реализации какого-нибудь объекта или добавление новых функций не влияет на другие объекты системы.

Общий процесс объектно-ориентированного проектирования состоит из ***нескольких крупных этапов***:

— определение рабочего окружения системы и разработка моделей ее использования;

— проектирование архитектуры программной системы;

— определение и идентификация основных объектов системы;

— разработка модели архитектуры комплекса программ;

— определение и документирование интерфейсов объектов.

Процесс ООП нельзя представить в виде простой схемы (как при структурном проектировании), в которой предполагается четкая последовательность этапов. Фактически все перечисленные этапы в значительной мере можно выполнять параллельно, с учетом взаимного влияния друг на друга. Как только разработана архитектура системы, определяются объекты и интерфейсы. После создания моделей объектов отдельные объекты можно переопределить, а это может привести к изменениям в архитектуре системы.

Главное преимущество ООП программных средств состоит в том, что оно ***упрощает задачу внесения изменений в системную архитектуру***, поскольку представление состояния объекта не оказывает на нее влияния. Изменение внутренних данных объекта не должно влиять на другие объекты системы. Более того, так как объекты слабо связаны между собой, обычно новые объекты просто вставляются без значительных воздействий на остальные компоненты системы.

Основные ***понятия*** ООП включают:

— при объектно-ориентированном проектировании основные компоненты программной системы представляются как объекты со своими состояниями и операциями;

— объекты предоставляют сервисы (методы) другим объектам и создаются в реальном времени на основе определения класса объектов;

— объекты могут быть реализованы последовательно и параллельно, параллельный объект может быть пассивным, у которого состояние изменяется только через его интерфейс, или активным, который может изменять свое состояние без вмешательства извне;

— в процессе объектно-ориентированного проектирования возможно создание ряда различных моделей, которые можно разделить на статические (модели классов, модели обобщения, модели агрегирования) и динамические (модели последовательностей, модели конечного автомата);

— важным преимуществом объектно-ориентированного проектирования является то, что он упрощает процесс модификации системы.

Одна часть общей системы занимается сбором данных, другая обобщает данные, полученные из различных источников, третья выполняет архивирование данных и, наконец, четвертая создает результаты. Система представляет собой многоуровневую архитектуру, в которой отражены все этапы обработки данных в системе, сбор и обобщение данных, архивирование данных и создание результатов. Такая многоуровневая архитектура вполне годится для проектирования, так как каждый этап основывается только на обработке данных, выполненной на предыдущем этапе.

Использование методов ООП строго ***регламентировано***, поэтому:

— возрастает производительность труда разработчиков благодаря переходу к высокоэффективному методу — на базе предварительного анализа проекта;

— запросы и объекты реального мира проще моделируются путем концентрации внимания на классах, а не на алгоритмах их функционирования;

— компоненты системы легко изменяются и применяются повторно;

— требования проще отслеживаются;

— поддерживается эффективное прототипирование;

— разработка проекта отличается непрерывностью в представлении объектов — одни и те же типы диаграмм применяются как при анализе, так и на этапе разработки;

— работа по проектированию может осуществляться с помощью универсальных технологических инструментов.

ООП — только ***часть объектно-ориентированного процесса разработки системы***, где на протяжении всего процесса создания ПС используется объектно-ориентированный метод. Этот подход подразумевает выполнение ***трех этапов***:

— объектно-ориентированный ***анализ*** — создание модели предметной области приложения ПС, где объекты отражают реальные объекты-сущности, а также определяются операции, выполняемые объектами;

— объектно-ориентированное ***проектирование*** — разработка модели системы ПС и системной архитектуры с учетом системных требований, в которой определение всех объектов подчинено решению конкретной задачи;

— объектно-ориентированное ***программирование*** — реализация архитектуры (модели) системы с помощью объектно-ориентированного языка программирования (например, Java), непосредственно выполняющего отражение определенных объектов и предоставляющего средства для определения классов объектов.

Этапы могут «перетекать» друг в друга, т.е. могут не иметь четких рамок, причем на каждом этапе обычно применяется одна и та же система нотации. Переход на следующий этап приводит к усовершенствованию и конкретизации результатов предыдущего этапа путем более детального описания определенных ранее классов объектов и определения новых классов. Так как данные скрыты внутри объектов, детальные решения о содержании данных можно отложить до этапа реализации системы. В некоторых случаях можно также не спешить с принятием решений о расположении объектов и о том, будут ли эти объекты последовательными или параллельными. Если необходимы функциональные изменения, они производятся внутри объекта, что приводит к незначительным изменениям в оставшейся внешней части его процессов. Для обновления или добавления функций оставшиеся объекты поддерживаются с помощью интерфейсов. Объектно-ориентированное проектирование отображает предметную область задачи и ответственности системы, но задерживает определение подробностей реализации объектов, переносит их на более поздний этап разработки и минимизирует влияние изменений в функциональных требованиях.

При ООП основное внимание уделяется тому, ***что*** следует делать, ***каким образом*** добиться цели, а процесс ее достижения целиком зависит

от этапа разработки. Объектная декомпозиция дает возможность создавать программные комплексы визуально меньшего размера путем использования общих механизмов, обеспечивающих необходимую экономию выразительных средств. Использование объектного подхода повышает уровень унификации разработки и ***пригодность для повторного использования*** не только программных компонентов, но и больших комплексов программ, что ведет к созданию унифицированной среды разработки и переходу к ***сборочному*** созданию программных продуктов. Системы зачастую получаются более компактными, чем их структурные эквиваленты, что означает не только уменьшение объема программного кода, но и удешевление проекта за счет использования компонентов из предыдущих разработок. Однако структурный подход сохраняет свою высокую значимость и широко используется на практике. Взаимосвязью между структурным и объектно-ориентированным подходами является общность ряда категорий и понятий жизненного цикла ПС.

Объектная декомпозиция существенно отличается от функциональной, поэтому переход на новую технологию связан как с ***преодолением психологических трудностей***, так и с дополнительными финансовыми затратами. Кроме того, диаграммы, отражающие специфику объектного подхода, гораздо ***менее наглядны*** и хуже понимаемы непрофессионалами, объектно-ориентированный подход обычно не дает немедленной отдачи. Эффект от его применения начинает сказываться после разработки нескольких проектов и накопления повторно используемых компонентов, отражающих типовые проектные решения в данной области.

В программных средствах при ООП рекомендуется выделять ***три уровня***:

— уровень ***интерфейсов***, который занимается всеми взаимодействиями с другими частями системы и предоставлением внешних интерфейсов системы;

— уровень ***сбора данных***, управляющий сбором информации из внешней среды и обобщающий данные перед отправкой их в систему построения обобщенных результатов;

— уровень ***объектов***, в котором представлены и описаны все объекты, используемые в процессе сбора исходных данных.

В общем случае рекомендуется структурировать систему на части так, чтобы архитектура была как можно проще. Согласно хорошему ***прак-***

тическому правилу модель архитектуры должна состоять не более чем ***из семи-восьми основных объектов***. Перед выполнением проектирования должны быть сформированы представления относительно основных объектов проектируемой системы. Вместе с тем требуется определить и документировать все другие внешние объекты системы. Определения объектов на данном этапе проектирования отражают ***классы объектов,*** и структура системы описывается в терминах этих классов. Классы объектов, определенные ранее, получают более детальное описание, поэтому иногда приходится возвращаться на данный этап проектирования для переопределения классов. Первый этап в любом процессе проектирования состоит в выявлении взаимоотношений между проектируемым ПС и его окружением. Выявление этих взаимоотношений помогает решать, как обеспечить необходимую функциональность системы и как структурировать систему, чтобы она могла эффективно взаимодействовать со своим окружением.

Язык **UML** представляет ***набор разработанных инженерных задач*** практического профиля, которые успешно испытаны при моделировании крупных и сложных систем. Поддерживается метамодель для диаграмм классов и набор семантических и синтаксических правил, определяющих суть элементов и их отношений. Модель поддерживается большим количеством автоматизированных инструментов. Основные ***цели*** использования языка **UML** на этапе разработки проекта ПС:

— поддержка на уровне пользователей готового к применению, выразительного визуального языка моделирования, применяемого для разработки и обмена моделями;

— поддержка спецификаций, независимых от определенных языков программирования и процессов разработки;

— поддержка формального базиса для представления языка моделирования;

— поддержка высокоуровневых понятий, таких как компоненты, элементы сотрудничества, каркасы и шаблоны;

— интеграция наилучшего опыта.

Разработчики **UML** установили, что даже при использовании современных передовых технологий, при сборе требований, анализе и разработке продолжается борьба с неадекватными и ***постоянно изменяющимися требованиями*** заказчиков (см. лекцию 6). Варианты использования пред-

назначены для уточнения динамических требований и выработки более четкого представления возможных изменений в поведении системы. Они позволяют реализовать функциональные усовершенствования, которые отражаются и адресуются в организованной форме. Классы и объекты языка **UML** демонстрируют гибкость и управляемость при использовании статично и динамично несовершенных описаний. Для оживления общения в команде специалистов ООП позволяет реализовать в группе программистов интегрированный подход с помощью общего словаря и языка; инкапсуляция данных и алгоритмов позволяет членам команды работать над компонентами в параллельном режиме. Также в этом случае можно ограничить влияние изменчивости требований; наследственные свойства функций и атрибутов являются эффективным дополнением к функциональным возможностям.

8.2. Основные понятия и модели объектно-ориентированного проектирования программных средств

Класс представляет собой абстракцию в предметной области приложения, в общем случае выраженную существительным. Абстракция может быть концептуальной или физической: она отражает возможности системы по хранению информации о ней, по взаимодействию с ней или же оба эти фактора. Класс включает: атрибуты (данные, описывающие объект, а также тот объем информации об объекте, который хранится в системе); отношения с другими классами и операциями (поведение данного объекта, которое и описывает соответствующие ответственности). Диаграмма класса отражает содержимое классов (набор декларативных, статичных элементов модели), и их взаимоотношения. Класс определяет интерфейсы соответствующих экземпляров объектов (таблица 8.1).

Классы должны отличаться идентичностью структуры, иметь четко определенные ответственности и поддерживать системные функции, взаимодействуя с другими объектами посредством сообщений. Классы описываются с помощью: атрибутов (данные, свойства), операций (службы, функции, поведение, процесс, методы), жизненного цикла разработки ПС (состояние, идентичность, независимость существования) и ассоциаций

(отношения, связи, соединения). Классы имеют свойства, структуру, поведение и отличаются независимостью существования. Класс может применяться для определения подклассов, которые могут быть проиллюстрированы примерами. Однако класс нельзя проиллюстрировать непосредственно, опираясь только на него самого. Классы обладают поведением, которое также называется операцией, службой, функцией или методом. Эти термины используются в спецификации **UML**, где операция описывает класс и объект, а метод — реализацию операции.

Таблица 8.1

Основные компоненты объектно-ориентированного проектирования программных средств

Компоненты	Содержание и функции
Классы	Объединения однородных объектов, имеющих одинаковые атрибуты, структуру и поведение
Объекты	Реальности (сущности), описываемые границами, индивидуальными состояниями и поведением
Атрибуты	Характеристики класса, отражающие идентификацию, состояние и поведение конкретного объекта этого класса
Ассоциации	Отношение между классами, отражающее связи между объектами этих классов
Интерфейсы	Множества операций взаимодействия между объектами
Сообщения	Взаимодействия объектов, имеющие стимулы, отправителя и получателя
Операции	Возможные воздействия объекта на другой объект того же класса с целью вызвать отклик
Состояние	Ситуация, в течение которой объект выполняет деятельность или ожидает события
Инкапсуляция	Выделение и сокрытие части информации об объекте
Наследование	Совместное использование атрибутов и поведения объектов в пределах иерархической структуры для построения новых классов
Диаграмма последовательности	Диаграмма взаимодействия объектов, упорядоченных по времени их проявления

Существует ряд подходов к ***определению классов объектов***:

— использование грамматического анализа естественного языкового описания системы, объекты и атрибуты — это существительные, операции

и сервисы — глаголы, такой подход реализован в иерархическом методе объектно-ориентированного проектирования, который широко используется;

— использование в качестве объектов ПС событий, объектов и ситуаций реального мира из области приложения (например самолетов, ролевых ситуаций менеджера) для реализации таких объектов, могут потребоваться специальные структуры хранения данных (абстрактные структуры данных);

— применение подхода, основанного на сценариях, в котором по очереди определяются и анализируются различные сценарии использования системы; группа, отвечающая за анализ, должна идентифицировать необходимые объекты, атрибуты и операции; метод анализа, при котором аналитики и разработчики присваивают роли объектам, отражают эффективность подхода, основанного на сценариях.

Для описания классов можно использовать информацию, полученную из разных источников. Объекты и операции, первоначально определенные на основе неформального описания системы, могут служить отправной точкой при ООП. Затем для усовершенствования и расширения описания первоначальных объектов можно использовать дополнительную информацию, полученную из области применения ПС или анализа сценариев. Дополнительную информацию также можно получить в ходе обсуждения с пользователями разрабатываемой системы или анализа имеющихся систем.

Объект является отдельным (особенным) экземпляром класса. Объекты — это исполняемые сущности с атрибутами и сервисами класса объектов. Объекты представляют собой реализацию класса, на основе одного класса можно создать много различных объектов. Обычно при разработке объектных моделей основное внимание сосредоточено на классах объектов и их отношениях. Во всех случаях применяется общее правило, согласно которому объект инкапсулирует данные о своем внутреннем строении.

Объект — это сущность, способная пребывать в различных состояниях и имеющая определенное множество операций. Состояние определяется как набор атрибутов объекта. Операции, связанные с объектом, предоставляют сервисы (функциональные возможности) другим объектам (клиентам) для выполнения определенных вычислений. Объекты создаются в соответствии с определением класса объектов, которое служит шаблоном

для создания объектов. В него включены объявления всех атрибутов и операций, связанных с объектом данного класса. Нотация, которая используется для обозначения классов объектов, определена в **UML**.

Все объекты из класса имеют значения атрибутов, соответствующие атрибутам из полного дескриптора классов. Эти объекты будут также поддерживать операции, описываемые дескриптором классов. Объекты являются экземплярами класса и совместно используют свойства (атрибуты и операции) данного класса. Каждый объект отличается собственной идентичностью и имеет характерный набор значений для атрибута. Он является сущностью, инкапсулированной в двух воплощениях — ***состояния и поведения***. Состояние представлено с помощью атрибутов и связей; операции и механизмы состояния представляют поведение. Состояние сохраняет эффекты от производимых некоей сущностью операций. Диаграммы объектов отображают объекты, их ассоциации и отношения (связи) в их развитии во времени. Объектом (если он подобен сущности) ***может быть***:

— материальный предмет (или индивидуум);
— выполняемая роль;
— событие;
— взаимодействие (контракт);
— операционная процедура (обзор);
— организационная единица;
— место (банк);
— структура.

Объект является экземпляром, который структурирован и функционирует в соответствии со своим классом. Все объекты, порожденные одним и тем же классом, структурированы одним способом, хотя каждый из них имеет свой собственный набор ***связей атрибутов***. Каждая связь атрибута имеет ссылку на экземпляр, обычно на значение данных. Объект может порождаться несколькими классами. В этом случае объект обладает всеми свойствами, которые объявлены во всех этих классах, как структурными, так и поведенческими. К объекту можно добавлять новые классы, а старые классы отделять. Это значит, что свойства новых классов динамически добавляются к данному объекту, а свойства, объявленные ранее в классе, удаляемые из объекта, динамически также удаляются из объекта.

Интерфейсом называется набор операций, характеризующих поведение элемента. Интерфейсы можно использовать для установки атрибу-

та, возвращения значения атрибута и для запроса объекта с целью выполнения операции. Важной частью любого процесса ООП является ***специфицирование интерфейсов*** между различными компонентами системы. Интерфейсы необходимо определять так, чтобы объекты и другие компоненты можно было проектировать параллельно. Определив интерфейс, разработчики других объектов могут считать, что интерфейс уже реализован. Один и тот же объект может иметь несколько интерфейсов, причем каждый из них предполагает свой способ поддержки методов. Проектирование интерфейсов объектов связано со спецификацией интерфейса в объекте или группе объектов. Интерфейсы можно определить подобно диаграмме классов. По мере усложнения интерфейсов такой подход оказывается более эффективным, так как для обнаружения ошибок и противоречий в описании интерфейса можно воспользоваться средствами проверки синтаксиса языка программирования.

Сообщения — содержат объект назначения (включающий операцию, предназначенную для выполнения), название выполняемого оператора и параметры, которые необходимы для выполнения операции. Эти интерфейсы определяют средства взаимодействия между объектами. Каждому объекту пересылаются сообщения других объектов. Интерфейсы относятся к общедоступным методам, поскольку они имеют отношение к другим объектам. Полезно определять интерфейсы заранее, т.е. в начале жизненного цикла ПС. В этом случае команда программистов может быть разделена с учетом границ между объектами, что может быть предпочтительнее разделения по функциональным границам. После того как интерфейсы и набор ответственностей четко определены, они могут функционировать независимо, объединяясь лишь позднее, в цикле по тестированию модели.

Сообщение является спецификацией по транспортировке информации из одного экземпляра объекта в другой, причем ожидается, что эта деятельность имеет результат (сообщение может указывать на формирование сигнала или на вызов операции). ***Стимул*** определяет передачу информации из одного экземпляра объекта в другой, например, формирование сигнала или вызов операции; получение сообщения означает обработку стимула, который передается из экземпляра отправителя. ***Получатель*** (объект) является объектом для обработки стимула, который передается из экземпляра отправителя. ***Отношение*** представляет семантическую связь

между элементами модели (примеры отношений включают ассоциации и обобщения). ***Ответственность*** является контрактом или обязательством создателя классов; для пересылки сообщения стимул передается из экземпляра объекта отправителя к экземпляру получателя. ***Отправитель*** (объект) представляет собой объект, передающий стимул в объект получателя.

Атрибут представляет собой описание набора значений, которые могут вызываться экземплярами объектов данного класса. Эта информация является для объекта внутренней и имеет следующие особенности:

— представление с помощью существительного;

— описание объекта в терминах реального времени;

— возможность служить индикатором состояния;

— обладание типом данных;

— идентичность для объектов одного класса — может отличаться по значению, но не по сути;

— возможность получения значения, определенного с помощью домена пересчета (набор определенных значений).

Операция представляет собой услугу, которая может запрашиваться из объекта для оказания влияния на его поведение. Операции можно описывать согласно следующим параметрам:

— инкапсулирование внутри объекта;

— отклик на стимул (сообщение);

— возможность действия, выполняемого объектом с помощью другого объекта;

— возможность преобразования, которому подвергается объект.

Метод является специфической реализацией операции. Операция (метод) является средством, с помощью которого объект может реализовать свою ответственность. Операция для объекта вызывается с помощью сообщения, пересылаемого из другого объекта. Все объекты имеют методы, применяемые для их инициализации и выявления в рамках модели, а также возможности для приобретения атрибутов и избавления от них. Методы представляют собой способы взаимодействия объектов между собой, сообщений для вызова (стимулирования) определенной деятельности (поведения) в пределах объекта получателя. В ходе проведения анализа для этого к каждому объекту, имеющему ссылку на другие объекты, добавля-

ется внешний ключ. Метод представляет реализацию для операции — указывает алгоритм или процедуру, которая ассоциируется с операцией.

Ассоциация отражает важное соединение (связь) между понятиями или классами (объектами). Рассматриваемое понятие включает, по крайней мере, два ассоциативных конца. Свойство множественности для концов ассоциации показывает, какое число экземпляров объектов можно ассоциировать с отдельным экземпляром класса.

Процесс инкапсуляции состоит из отделения внешних аспектов объекта от последствий внутренней реализации этого объекта. Другим термином инкапсуляции является ***сокрытие информации***. Внешние аспекты объекта доступны другим объектам с помощью методов самого объекта, в то время как внутренние реализации этих методов скрыты от внешнего объекта, пересылающего сообщения. Инкапсуляция важна для получения потенциала или для поддержки объектно-ориентированных моделей. Поскольку подробности реализации скрыты от других объектов, они содержатся в пределах объекта. Влияние изменений, вносимых в реализацию метода, минимально для всей модели.

Потенциально все объекты являются повторно используемыми компонентами, так как они независимо инкапсулируют данные о состоянии и операции. Архитектуру ПС можно разрабатывать на базе объектов, уже созданных в предыдущих проектах. Такой подход снижает стоимость проектирования, программирования и тестирования ПС. Кроме того, появляется возможность использовать стандартные объекты, что уменьшает риск, связанный с разработкой программного средства. Однако иногда повторное использование эффективнее всего реализовать с помощью коллекций объектов (компонентов или объектных структур), а не через отдельные объекты.

Наследование определяет совместное использование атрибутов и поведение объектов в пределах иерархической структуры (суперклассов и/или подклассов). Каждый подкласс наследует все свойства — (атрибуты и операции) суперкласса (предка) и добавляет свои собственные уникальные свойства. Класс содержит описание всех атрибутов, ассоциаций и операций, входящих в объект, что является обобщением объекта. Каждый класс имеет набор наследуемых свойств — атрибутов, операций и ассоциаций. Эти структуры данных и алгоритмы немедленно становятся доступ-

ными для подклассов наследников. Класс может иметь потомков (подклассы), где потомок является специализацией предшественника. Потомок наследует (включает в себя) эту структуру и поведение общих предшественников, при этом он способен добавлять свои собственные атрибуты и операции. Такое отношение также известно как обобщение (генерализация). Наследование/специализация/обобщение является преимуществом метода ООП, поскольку поддерживает повторное использование классов. Благодаря применению этих понятий свойства суперклассов повторяются в каждом объекте подкласса, и таким образом поддерживается высокий фактор обновления. Изменения для атрибутов или операций немедленно наследуются всеми подклассами.

Модели систем, разрабатываемые при формировании требований, должны отображать реальные сущности, принадлежащие классам объектов. Все объекты связаны с ***различными уровнями в архитектуре системы***. Конкретные решения по архитектуре системы можно принимать в процессе ее реализации. Когда связи между разработчиками требований, проектировщиками и программистами не очень тесные (например, если система проектируется в одном подразделении предприятия, а реализуется в другом), требуется более детализированная модель. Следовательно, в процессе проектирования важно решить, какие требуются модели и какой должна быть ***степень их детализации***. Это решение зависит также от типа разрабатываемой системы.

Классы не должны содержать информацию об отдельных системных объектах. Можно разработать различные типы объектных моделей, показывающие, как классы связаны друг с другом, как объекты агрегируются из других объектов, как объекты взаимодействуют с другими объектами. Эти модели расширяют понимание разрабатываемой системы. Идентификация объектов и классов объектов считается наиболее сложной задачей в процессе объектно-ориентированной разработки систем. Определение объектов — это основа для анализа и проектирования системы.

Модель окружения системы и модель использования системы представляют собой две дополняющие друг друга модели взаимоотношений между данной системой и ее окружением. Модель окружения системы — это статическая модель, которая описывает другие системы из окружения разрабатываемого ПС. Модель использования системы — динамическая

модель, которая показывает взаимодействие данной системы со своим окружением. Модель окружения системы можно представить с помощью схемы связей, которая дает простую блок-схему общей архитектуры системы. С помощью ***пакетов*** языка **UML** ее можно представить в развернутом виде как совокупность подсистем. Такое представление показывает, что рабочее окружение системы находится внутри подсистемы, занимающейся сбором данных. При моделировании взаимодействия проектируемой системы с ее окружением при ООП применяется абстрактный подход, который не требует больших объемов данных для описания этих взаимодействий. Подход, применяемый в **UML**, состоит в том, чтобы разработать модель вариантов использования, в которой каждый вариант представляет собой определенное взаимодействие с системой.

Объектные модели, разработанные для формирования требований, могут использоваться как для представления данных, так и для процессов их обработки. В этом отношении они объединяют модели потоков данных и семантические модели данных. Они также полезны для классификации системных сущностей и могут представлять сущности, состоящие из других сущностей. Для некоторых классов систем объектные модели — естественный способ отображения реально существующих объектов, которые находятся под управлением системы. Например, для систем, обрабатывающих информацию относительно конкретных объектов (таких, как автомобили, самолеты, книги), которые имеют четко определенные атрибуты. Более абстрактные высокоуровневые сущности (например, библиотеки, медицинские регистрирующие системы или текстовые редакторы) труднее моделировать в виде классов объектов, поскольку они имеют достаточно сложный интерфейс, состоящий из независимых атрибутов и методов.

Объектные модели, разработанные во время анализа требований, упрощают переход к объектно-ориентированному проектированию и программированию. Однако конечные пользователи часто считают объектные модели неестественными и трудными для понимания. Часто они предпочитают функциональные представления процессов обработки данных. Поэтому полезно дополнить их моделями потоков данных, чтобы показать сквозную обработку данных в системе.

Модели данных — больших программных систем используют информационные базы данных. В одних случаях эта база данных существует

независимо от программной системы, в других — специально создается для разрабатываемой системы. Важной частью моделирования систем является определение логической формы данных, обрабатываемых системой. Наиболее широко используемой методологией моделирования данных является моделирование типа «сущность — связь — атрибут», которое показывает структуру данных, их атрибуты и отношения между ними.

Для описания структуры обрабатываемой информации модели данных часто используются совместно с моделями потоков данных. Проекты структуры ПС представляются ориентированными графами. Они состоят из набора узлов различных типов, соединенных дугами, отображающими связи между структурными узлами. В системе проектирования присутствуют средства вывода на дисплей этого графа (т.е. структурной диаграммы) и его преобразования к виду, удобному для хранения в базе данных проектов. Система редактирования выполняет преобразования структурной диаграммы из формата базы данных в формат, позволяющий отобразить ее на экране монитора в виде блок-схемы. Информация, предоставляемая редактором другим средствам анализа проекта, должна включить логическое представление графа проекта. Эти средства работают с объектами, их логическими атрибутами и связями между ними.

Сущность — реальный или абстрактный объект, имеющий определяющее значение для рассматриваемой системы (это может быть объект как самой системы, так и ее окружения). Каждая сущность должна иметь уникальное имя и обладать одним или несколькими атрибутами, которые либо принадлежат сущности, либо наследуются через связь. Сущность соответствует классу (или типу) объектов, а не конкретному экземпляру класса. Поименованная связь (ассоциация) между двумя сущностями осуществляется с помощью глаголов (например, имеет, определяет, принадлежит). Атрибут — любая характеристика сущности (предназначается для идентификации, классификации, численной параметризации или описания состояния сущности). Значения атрибутов однозначно идентифицируют экземпляр сущности.

Язык моделирования **UML** не имеет определенных обозначений для типа моделей данных, что желательно для объектно-ориентированного процесса разработки ПС, где для описания систем используются объекты и их отношения. Если сущностям поставить в соответствие простейшие классы объектов (без ассоциированных методов), тогда в качестве моде-

лей данных можно использовать модели классов **UML** совместно с именованными ассоциациями между классами.

Модели наследования. Важным этапом объектно-ориентированного моделирования является определение классов объектов, которые затем систематизируются. Это подразумевает создание схемы классификации, которая показывает, как классы объектов связаны друг с другом посредством общих атрибутов и сервисов. Схема классификации организована в виде иерархии наследования, на вершине которой представлены наиболее общие классы объектов. Более специализированные объекты наследуют их атрибуты и сервисы. Эти объекты могут иметь собственные атрибуты и сервисы. В нотации **UML** наследования показываются сверху-вниз, как принято в других объектно-ориентированных нотациях. Стрелка выходит из класса, который наследует атрибуты и операции, и направлена к родительскому классу. В **UML** вместо термина «наследование» чаще используется термин «обобщение». В моделях множественного наследования классы могут иметь нескольких родителей. Тогда наследуются атрибуты и сервисы от каждого родительского класса.

Упрощение моделей при ООП может вызывать некоторые затруднения специалистов в понимании следующего:

— одни и те же динамические и статические модели описывают в разработке только «способы» взаимодействия с объектами;

— инкапсуляция скрывает внутреннее содержание объекта, позволяя разработчику уделять больше внимания методам использования объектов — их существенных, неотъемлемых характеристик; позволяет разделять статус, функцию, поведение; ограничивает доступ к переменным, которые функционируют внутри алгоритма;

— агрегация позволяет создавать крупный объект из небольших, упрощать вид объектов, позволяя выполнять обработку сложных состояний.

8.3. Варианты представления моделей и средства объектно-ориентированного проектирования программных средств

Существует ***два типа*** объектно-ориентированных моделей системной архитектуры:

— статические модели, которые описывают структуру системы в терминах классов объектов и взаимоотношений между ними, которые документируются на данном этапе, являются отношениями обобщения, отношениями «используют — используются» и структурными отношениями;

— динамические модели, которые описывают структуру системы и показывают динамические взаимодействия между объектами системы (но не классами объектов), — документируемые взаимодействия содержат последовательность составленных объектами запросов к сервисам и описывают реакцию системы на взаимодействия между объектами.

В языке моделирования **UML** поддерживается ряд возможных статических и динамических моделей:

— модели подсистем, которые показывают логически сгруппированные объекты, они представлены с помощью диаграммы классов, в которой каждая подсистема обозначается как пакет, и является статическим;

— модели последовательностей, которые показывают взаимодействия между объектами, они представляются в **UML** с помощью диаграмм последовательности или кооперативных диаграмм — динамические модели;

— модели конечного автомата, которые показывают изменение состояния отдельных объектов в ответ на определенные события, в **UML** они представлены в виде диаграмм состояния — динамические модели.

Модель подсистемы является одной из наиболее важных и полезных статических моделей, поскольку показывает, как можно организовать систему в виде логически связанных групп объектов. В **UML** пакеты являются структурами инкапсуляции и не отображаются непосредственно в объектах разрабатываемой системы. Существует несколько способов создания, применения и введения новых определений для моделей ООП с использованием диаграмм вариантов использования сценариев, диаграмм действий, диаграмм класс/объект, а также диаграмм сотрудничества, взаимодействия, перехода состояния, контекста данных и словаря данных. Некоторые вводятся сверху-вниз, другие — снизу-вверх, и все они итеративно определяются заново с помощью сотрудничества.

Статическое представление объектно-ориентированного анализа — представляет собой диаграмму класса, а также диаграмму объекта для представления определенного экземпляра класса, куда входят компоненты атрибутов, служб и отношений наряду с понятиями, взятыми из

иерархии, абстракции, инкапсуляции и наследования. Идентифицируются ответственности классов, затем обработка идет наверх, а для достоверности применяются сценарии вариантов использования. В этом случае реализуется подход снизу-вверх. Подобный способ достаточно хорош, но существует еще путь сверху-вниз, когда при идентификации классов начинают с практических примеров, а затем продолжается обработка вниз, доходя до сценариев. Если начать с классов, можно сначала идентифицировать их через абстракции. Затем для каждого класса перечисляются ответственности, идентифицируются атрибуты и операции (поведение — снова через абстракцию) и, наконец, обращаются к сценариям вариантов использования для подтверждения диаграммы класса. Однако сначала рассматривается описание диаграмм классов.

Статическое представление объектно-ориентированной разработки проекта — диаграммы взаимодействия и сотрудничества отображают взаимодействия, которые пространственно ориентированы на окружение модели и характеризуют классы и ассоциации (экземпляры и связи). Целью сотрудничества является определение способов реализации вариантов использования. Диаграмма сотрудничества отображает отношения между экземплярами объектов. Поведение реализуется с помощью набора объектов, обменивающихся входными сигналами в рамках общего взаимодействия, что позволяет достичь поставленной цели. Для представления механизмов, применяемых при разработке, важно обращать внимание только на определенные объекты и изучать взаимодействие между ним. Рассматриваемые объекты всегда выделяются из большей системы, в которой данные объекты являются лишь компонентами.

Сотрудничество уточняет контекст, который позволяет выразить поведение реализуемого элемента в терминах, единых для всех участников сотрудничества. Таким образом, в то время как модель представляет систему в целом, сотрудничество является лишь частичным отображением данной модели. Именно сотрудничество определяет эффективность применения подмножества содержимого модели. Сотрудничество можно охарактеризовать на двух различных уровнях: на уровне спецификации или на уровне экземпляра. Диаграмма, представляющая сотрудничество на уровне спецификации, характеризует роль классов и ассоциаций. В то же время диаграмма на уровне экземпляра отображает экземпляры и связи,

которые согласуются с ролями в сотрудничестве. При отображении сотрудничества указывается, какие экземпляры свойств должны принимать участие в сотрудничестве, т.е. каждый участник указывает необходимые свойства, которыми должен обладать согласуемый экземпляр.

Динамическое представление объектно-ориентированного проектирования — диаграммы взаимодействия и последовательные диаграммы демонстрируют взаимодействие, отображенное в динамике. В ней отображаются экземпляры, участвующие во взаимодействии с помощью соответствующих линий жизни, а также входные сигналы, которыми они обмениваются. Эти сигналы собраны во временную последовательность. В отличие от диаграммы сотрудничества, последовательная диаграмма включает временные последовательности, но не содержит объектных отношений. Последовательная диаграмма может существовать в общей форме (описывать все возможные сценарии) и в форме экземпляра (описывать один действительный сценарий). Последовательные диаграммы и диаграммы сотрудничества выражают подобную информацию, однако используют различные способы. Последовательная диаграмма имеет две размерности: размерность по вертикали представляет время; размерность по горизонтали представляет различные объекты.

Динамическое представление объектно-ориентированной разработки проекта — диаграммы переходов между состояниями представляют основанное на состоянии поведение класса экземпляров объектов для всех сценариев. Обычно моделируются лишь классы с объектами, которые, как ожидается, проходят через наборы состояний. Состояние представляет условие или ситуацию во время жизни объекта, при которой удовлетворяется определенное условие, реализуется некоторая деятельность или же ожидается какое-либо событие. Диаграмма состояния отображает механизм состояния — поведение, определяющее последовательности состояния, которые проходит объект или взаимодействие во время своей жизни при отклике на события, вместе с соответствующими откликами и действиями.

Механизм состояния определяет набор понятий, которые могут использоваться для моделирования дискретного поведения с помощью конечных систем переходных состояний. Экземпляры событий генерируются в результате определенного действия в пределах системы или ее окру-

жения. Затем событие ориентируется на одну или несколько целей. Средства, с помощью которых экземпляры событий транспортируются к месту назначения, зависят от типа действия, цели, свойств среды коммуникации и многих других факторов. В некоторых случаях это происходит практически мгновенно и вполне надежно, в то время как в других случаях могут происходить периодические задержки передачи, потеря событий, переупорядочивание или дублирование. Поддерживается полная гибкость при моделировании различных типов коммуникационных возможностей. Событие ***получено***, если оно размещено в очереди событий по целевому назначению. Событие ***отправлено***, если оно извлечено из очереди событий и доставлено к механизму состояния для обработки. С этой точки зрения на него ссылаются как на ***текущее событие***. Наконец, событие ***поглощено***, если завершена его обработка. Поглощенное событие становится недоступным для обработки. Никакие требования не предъявляются к интервалам времени между получением события, отправлением и поглощением.

Состояние становится ***активным***, если оно вводится как результат некоторого перехода, а ***неактивным***, если завершается как результат перехода. Состояние может быть завершено и начато как результат одного и того же перехода. Диаграммы состояния представляют объекты, способные к динамическому поведению, путем указания соответствующего отклика на получение экземпляров события. Обычно они применяются при описании поведения классов. Состояние представляет собой условие во время жизни объекта или взаимодействие, во время которого оно удовлетворяет некоторое условие, выполняет определенное действие или ожидает некоторое событие. Концептуально объект остается в состоянии во время некоторого интервала времени.

Инструментальные средства анализа и проектирования ПС созданы для поддержки моделирования систем на этапах жизненного цикла программных средств. Они поддерживают создание, редактирование и анализ графических нотаций, используемых в структурных методах. Инструментальные средства анализа и проектирования часто поддерживают только определенные методы проектирования и анализа, например объектно-ориентированные. Другие инструментальные средства являются универсальными системами редактирования диаграмм многих типов, которые ис-

пользуются разными методами проектирования и анализа. Инструментальные средства, ориентированные на определенные методы, обычно автоматически поддерживают правила и базовые принципы этих методов, что позволяет выполнять автоматический контроль диаграмм.

Инструментальные средства обычно объединяются через общий ***репозиторий***, структура которого является собственностью разработчика пакета инструментальных средств. Центральный репозиторий позволяет проектировщику найти нужный проект, компонент и соответствующую проектную информацию. Обычно для создания общего репозитория инструментов используются системы баз данных типа Sybase или Oracle. Эти пакеты инструментальных средств содержат большое количество средств языков программирования четвертого поколения, предназначенных для генерирования программного кода на основе системной архитектуры, они также могут генерировать базы данных. Пакеты инструментальных средств обычно закрыты, т.е. не рассчитаны на добавление пользователями собственных инструментов или на изменение средств пакета, в который входят:

— редакторы диаграмм, предназначенные для создания диаграмм потоков данных, иерархий объектов, диаграмм «сущность-связь», эти редакторы не только имеют средства рисования, но и поддерживают различные типы объектов, используемых в диаграммах;

— средства проектирования, анализа и проверки выполняют проектирование ПС и создают отчеты об ошибках и дефектах в системной архитектуре, они могут работать совместно с системой редактирования, поэтому обнаруженные ошибки можно устранять на ранней стадии процесса проектирования;

— словарь данных хранит информацию об объектах, которые используются в структуре системы;

— средства генерирования отчетов на основе информации из центрального репозитория автоматически генерируют системную документацию;

— средства создания форм определяют форматы документов и экранных форм;

— средства импортирования и экспортирования позволяют обмениваться информацией из центрального репозитория различным инструментальным средствам;

— генераторы программного кода автоматически генерируют программы на основе проектов компонентов, хранящихся в центральном репозитории.

В некоторых случаях возможно генерировать программы или фрагменты программ на основе информации, представленной в системной модели. Поскольку в моделях не предусмотрена детализация низкого уровня, генератор программного кода не в состоянии сгенерировать законченный комплекс программ. Обычно необходимы программисты для завершения автоматически сгенерированных программ.

ЛЕКЦИЯ 9

УПРАВЛЕНИЕ РЕСУРСАМИ В ЖИЗНЕННОМ ЦИКЛЕ ПРОГРАММНЫХ СРЕДСТВ

9.1. Основные ресурсы для обеспечения жизненного цикла сложных программных средств

Общее понятие — ***доступные ресурсы обеспечения жизненного цикла ПС*** — включает реальные финансовые, временные, кадровые и аппаратурные ограничения затрат, в условиях которых происходит создание и совершенствование комплексов программ. В зависимости от характеристик объекта разработки на ее выполнение выделяются ресурсы различных видов и размеров. Эти факторы проявляются как дополнительные характеристики процессов ЖЦ и программных продуктов, а также их рентабельности, которые следует учитывать и оптимизировать. В результате доступные ресурсы становятся косвенными критериями или факторами, влияющими на выбор методов разработки, на достигаемые качество и эффективность применения программных продуктов. Многие проекты систем терпели и терпят неудачу из-за отсутствия у разработчиков и заказчиков при подготовке контракта четкого представления о реальных финансовых, трудовых, временных и иных ресурсах, необходимых для их реализации. Поэтому одной из основных задач при проектировании ПС является ***экономический анализ и определение необходимых ресурсов для создания и обеспечения всего ЖЦ ПС*** в соответствии с требованиями контракта и технического задания.

Наиболее общим видом ресурсов, используемых в жизненном цикле ПС, являются ***допустимые финансово-экономические затраты*** или эк-

вивалентные им величины трудоемкости соответствующих работ (см. лекцию 5). При разработке, тестировании и анализе качества этот показатель может применяться или как вид ресурсных ограничений, или как оптимизируемый критерий, определяющий целесообразную функциональную пригодность ПС. При этом необходимо также учитывать затраты на разработку, закупку и эксплуатацию системы качества, на технологию и комплекс автоматизации проектирования программ и баз данных, которые могут составлять существенную часть совокупной стоимости и трудоемкости разработки и всего ЖЦ ПС.

Затраты в жизненном цикле ПС определяются не только этапами разработки, но и этапами эксплуатации и сопровождения. Затраты на этих этапах могут значительно превышать затраты при разработке и характеризуются своими особыми закономерностями. Однако эффективность процесса разработки ПС невозможно определять без учета эффективности последующей эксплуатации, а для долго модифицируемых программ — без оценки эффективности их сопровождения. Ряд факторов влияет на затраты при разработке сложных ПС не только непосредственно, но и через возможное изменение затрат в дальнейшем при сопровождении или эксплуатации. Каждый из этапов: разработка, сопровождение и эксплуатация — может быть достаточно длительным. В пределах этапов различные группы затрат могут быть неодновременными и разделяться интервалами времени, исчисляемыми годами. Однако разновременность затрат трудно учитывать в общем виде и при существующих методиках имеется некоторая условность при оценке влияния времени на совокупные затраты проекта.

В соответствии с этапами жизненного цикла ПС основные затраты C_{Σ}, снижающие идеальную эффективность за цикл жизни $t_{ж}$, можно представить ***следующими составляющими*** (рис. 9.1):

C_P — совокупные затраты на разработку программ и обеспечение решения заданных функциональных задач, в том числе на технологическое обеспечение и аппаратуру ЭВМ при разработке ПС, в течение времени t_P;

C_C — затраты на сопровождение ПС за время t_c, включающие затраты на хранение и контроль их состояния, проведение модернизаций и исправление ошибок, тиражирование версий;

$C_Э$ — затраты на эксплуатацию программ и аппаратные средства ЭВМ, реализующих ПС, а также совокупные потери эффективности за время $t_Э$, вследствие ограниченных характеристик ЭВМ и неидеальности программ.

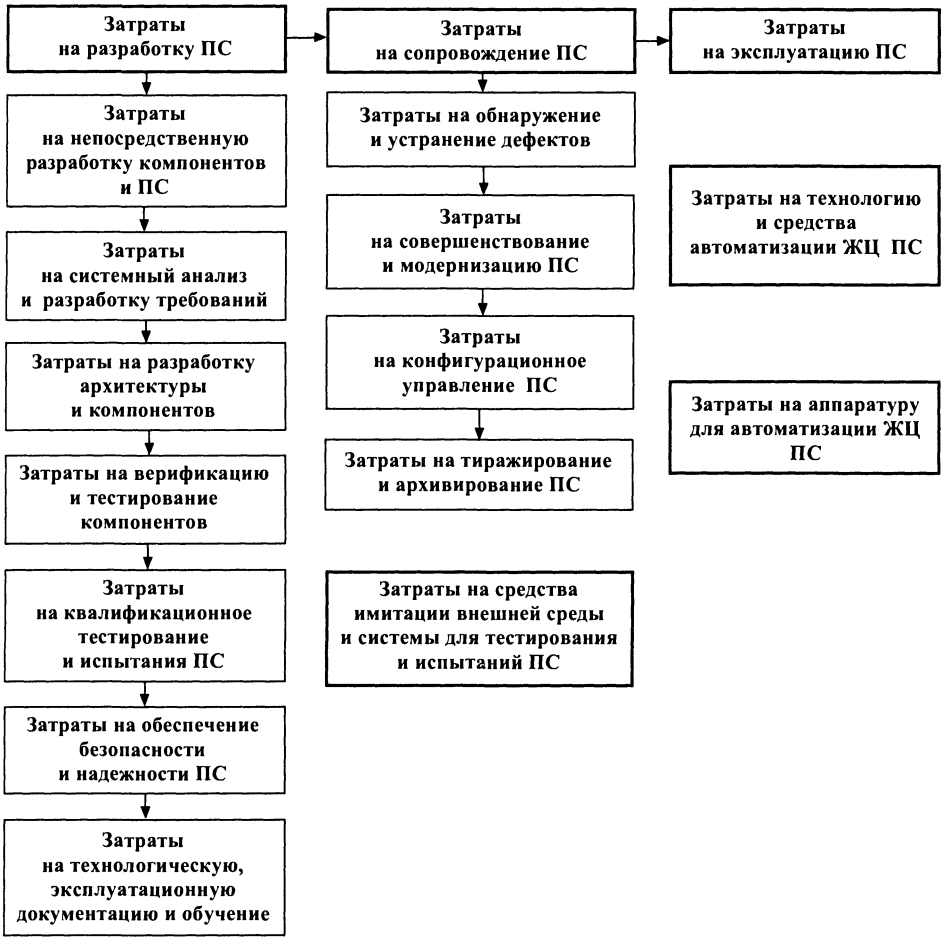

Рис. 9.1

В результате совокупные затраты ресурсов на программное средство за весь жизненный цикл длительностью $t_{ж}$ можно представить в виде

суммы: $C_Р + C_С + C_Э$. В зависимости от назначения и области использования программ экономическую эффективность целесообразно анализировать интегрально за весь период жизни, либо дифференциально за единицу времени (месяц, год). Для совместного анализа составляющих, определяющих эффективность, необходимо унифицировать методы временного анализа и единицы измерения составляющих затрат. При последующем изложении затраты рассматриваются на длительности цикла жизни или на соответствующих интервалах времени: разработки, эксплуатации и сопровождения.

Разработка сложного ПС требует во много раз больших затрат, чем производство каждого экземпляра при массовом тиражировании. Поэтому далее производство базового образца программ не выделяется в самостоятельный этап, а рассматривается совместно с процессом разработки ПС. Производство серийных образцов программного продукта включается в этап эксплуатации. Экономическая эффективность разработки и распределение ресурсов на ее выполнение могут значительно изменяться в зависимости от того, является ПС уникальным или будет изготовлен в сотнях экземпляров. Это обстоятельство привело к целесообразности оценки затрат на разработку ПС не только в абсолютных значениях для опытного или базового образца, но и в относительных величинах доли затрат на программы в каждом экземпляре реализующей (целевой) ЭВМ при серийном производстве систем.

При выделении ***составляющих затрат на разработку программ*** целесообразно учитывать их относительный вес в суммарных затратах и возможность локализации групп специалистов, влияющих на величину этих затрат. Разработка программ является областью с малой материало- и энергоемкостью, и основные затраты связаны с непосредственным или овеществленным трудом специалистов различных категорий. Поэтому для измерения затрат наиболее универсальной единицей стала трудоемкость в человеко-месяцах или человеко-годах. При этом учитываются все категории специалистов, участвующих непосредственно или косвенно в создании данного ПС.

Время или допустимая длительность разработки определенных компонентов и версий ПС является ***невосполнимым*** ограниченным ресурсом реальных проектов. Этот ресурс все больше определяет достижимое

качество сложных комплексов программ в процессе их разработки и сопровождения. Высокие требования заказчиков к сжатым срокам реализации проектов, естественно, ограничивают разработчиков и испытателей в продолжительности и объеме возможного системного анализа и проектирования, разработки и, особенно, тестирования программ. Увеличение числа привлекаемых для этого специалистов, при опытной эксплуатации или бета-тестировании, только в некоторых пределах позволяет ускорять разработку и увеличивать совокупное число тестов при проверках для повышения качества программ.

Доступные разработчикам ПС вычислительные ресурсы объектных и технологических ЭВМ являются одним из важнейших факторов, определяющим достижимое качество сложных ПС. В процессе проектирования целесообразно выделять определенные ресурсы ЭВМ на оперативное обеспечение качества, повышение защищенности, безопасности и надежности функционирования. Допустимая величина и рациональное распределение ресурсов ЭВМ на отдельные методы улучшения определенных конструктивных характеристик качества ПС оказывают существенное влияние на достигаемые ими значения.

Обобщенными ресурсами ЖЦ проекта ПС являются доступные ***стоимость или совокупные трудовые, временные и материальные затраты***, необходимые для приобретения, создания, модификации и эксплуатации компонентов и всего комплекса программ. Эти характеристики непосредственно влияют практически на все показатели качества и определяют рентабельность покупки или создания заново конкретного программного продукта. Это означает, что ***качество является относительным понятием***, которое зависит от ресурсов и субъектов, осуществляющих его оценку с позиции эффективности использования, а также от состояния рынка соответствующей продукции, ее производителей и технологий. Ориентация на потребителей подразумевает анализ их нужд и определение возможностей рынка удовлетворить эти потребности. При этом следует учитывать рыночную конкуренцию двух видов: между поставщиками готовых к применению программных средств с фиксированным качеством и между разработчиками, способными обеспечить жизненный цикл ПС или его существенную часть, с характеристиками качества, требующимися конкретному заказчику. В последнем случае на требуемое качество могут

оказывать влияние не только заказчик и непосредственные пользователи, но и различные посредники, организационные и торговые структуры, а также исполнители проекта.

В начале проектирования ПС всегда возникает задача оценивания ресурсов, которые необходимы и доступны для создания и обеспечения всего ЖЦ ПС, а также возможной экономической эффективности последующего применения комплекса программ по назначению при условии реализации требуемых характеристик качества. Экономическая эффективность и затраты имеют самостоятельное значение и методологию при анализе ЖЦ ПС. При планировании проектов программных средств часто инициатором разработки является разработчик-поставщик, который самостоятельно принимает все решения о проектировании за счет собственных ресурсов и предполагает возместить затраты при реализации программного продукта на рынке. В других случаях имеется определенный заказчик-потребитель, который способен задать основные цели, характеристики качества и обеспечить ресурсы для реализации проекта. Таким образом, ***при экономическом анализе ресурсов проектов ПС возможны два сценария*** (см. лекцию 5):

— создание и весь жизненный цикл комплекса программ и/или базы данных ориентируется разработчиком на массовое тиражирование и распространение на рынке, для заранее неизвестных покупателей-пользователей в различных сферах применения, при этом отсутствует приоритетный внешний потребитель-заказчик, который определяет и диктует основные требования, а также финансирует проект;

— разработка проекта ПС и/или БД предполагается поставщиком-разработчиком для конкретного потребителя-заказчика, задающего требования, который его финансирует, с определенным, необходимым ему тиражом и известной, ограниченной областью применения результатов разработки.

Разработка ПС характеризуется высокой долей творческого труда, особенно на начальных и завершающих этапах. Поэтому трудоемкость и длительность отдельных операций, частных работ и качество результатов при проектировании существенно зависят ***от индивидуальных особенностей*** относительно небольшого числа руководителей и исполнителей, а также от характеристик конкретного проекта. Отсюда принципиальной особенностью создания сложных комплексов программ является необхо-

димость активного участия руководителей и заказчиков проекта в системном анализе, итерационном экономическом оценивании затрат ресурсов и уточнении планов на базе прототипов завершенных разработок ПС и их личного опыта.

На рис. 9.1 отражены затраты на технологию, инструментальные средства автоматизации разработки и ЖЦ ПС, а также затраты на аппаратуру вычислительных систем, необходимую при обеспечении жизненного цикла комплексов программ. Эти затраты только обозначены в данной лекции, а их описания и оценки отнесены к лекции 16.

9.2. Ресурсы специалистов для обеспечения жизненного цикла сложных программных средств

Важнейшим ресурсом при создании программных средств являются люди — ***специалисты***, с их уровнем профессиональной квалификации, а также с многообразием знаний, опыта, стимулов и потребностей. Быстрый рост сложности и повышение ответственности за качество комплексов программ привели к появлению ***новых требований к специалистам программной инженерии,*** обеспечивающим все этапы жизненного цикла ПС. Эти требования отражены, в частности, в восьми современных принципах управления качеством продукции и технологии (см. лекцию 1). Теперь недостаточно навыков процедурного программирования небольших компонентов, а необходимы глубокие знания системотехники, технологии проектирования, методов обеспечения и контроля качества сложных комплексов программ в определенной области применения. Эти специалисты должны владеть новой интеллектуальной профессией, обеспечивающей высокое качество ЖЦ программных средств, а также контроль, испытания и удостоверение реального достигнутого качества на каждом этапе разработки и совершенствования программ. Крупномасштабное проектирование ПС различных классов, разделение труда специалистов по квалификации при разработке программ и данных, организация коллективов и экономика таких разработок стали важнейшей частью выбора, обучения и подготовки специалистов для обеспечения всего ЖЦ ПС.

При проектировании и создании высококачественных комплексов программ, прежде всего, необходимы организация и тесное ***взаимодействие***

представителей заказчика и разработчиков проекта. Взгляды и требования заказчика в основном отражаются в функциональных и потребительских характеристиках ПС. Устремления разработчиков направлены на способы их реализации с требуемым качеством. Эти различия исходных точек зрения на проект приводят к тому, что некоторые неформализованные представления тех и других имеют зоны неоднозначности и взаимного непонимания, что может приводить к конфликтам. Организация четкого взаимодействия и сокращение этих зон требует проведения определенных мероприятий и контактов по обмену знаниями, взаимному повышению квалификации и обучению.

Рис. 9.2

Представители заказчика, участвующие в проектировании, должны обучаться формализации автоматизируемых технологических процессов, для которых предназначены соответствующие ПС, и иметь представление об эффективных путях их реализации. С другой стороны, разработчики должны иметь в своем составе квалифицированных менеджеров, ***проблемно-ориентированных системных архитекторов*** (рис. 9.2), способных переводить функциональные требования заказчика в конкретные спецификации и технические требования к комплексу программ и его компонентам. Специалисты по проектированию сложных ПС (системные архитекторы) должны иметь, прежде всего, хорошую подготовку по системному анализу алгоритмов и пакетов прикладных программ, по методам оценки эффективности проектов, организации и планированию крупномасштабных разработок программ и баз данных. Им необходима высокая квалификация по архитектурному построению, комплексной отладке и испытаниям ПС определенных классов, умение организовать коллектив для решения общей целевой задачи системы. Это позволит на ранних этапах исключать или сокращать дефекты, обусловленные различием представления ими целей и задач проектов, а также их показателей качества.

Крупномасштабные ПС являются одними из наиболее сложных объектов, создаваемых человеком, и в процессе их разработки — ***творчество*** как поиск новых методов, альтернативных решений и способов осуществления заданных требований, а также формирование и декомпозиция этих требований составляют значительную часть всех трудозатрат. Индустриализация разработки ПС позволяет автоматизировать нетворческие, технические и рутинные операции и этапы, а также облегчает творческий процесс за счет селекции, обработки и подготовки информации, необходимой для принятия творческих решений. Следствием этого является значительное сокращение доли затрат на творческий труд в непосредственных затратах на разработку комплексов программ.

Однако ***в программной инженерии*** неуклонно повышается сложность создаваемых ПС, что вызывает возрастание затрат творческого труда на единицу размера программ. В перспективе, несмотря на автоматизацию и повышение инструментальной оснащенности технологии разработки программ, ***доля творческого труда при создании полностью новых крупномасштабных ПС возрастает***. Даже при сокращении суммарных

затрат на разработку программных компонентов за счет автоматизации нетворческого труда все более определяющей для технико-экономических показателей (ТЭП) создания ПС становится доля затрат на творческий труд и ***возрастают требования к творческим способностям специалистов***.

Необходимость всегда выполнять значительную долю творческих работ, которые даже для высококвалифицированных коллективов требуют определенных затрат, приводит к пониманию наличия потенциальных предельных значений ТЭП для новых разработок. По мере повышения квалификации коллектива и автоматизации творческой части труда следует ожидать асимптотического приближения к предельным значениям ТЭП. Эти предельные значения определяются ***интеллектуальными возможностями человека по интенсивности принятия творческих решений***. Они позволили выявить предельные значения производительности труда и длительности разработки сложных комплексов программ (см. лекцию 5). Реальным путем их оценки является изучение и экстраполяция прецедентов и экспериментальных данных реальных разработок ПС с наилучшими ТЭП, с учетом возрастания квалификации специалистов и уровня автоматизации разработок. Эти факторы способствуют эволюционному, относительно медленному приближению к предельным значениям ТЭП для новых ПС. Вряд ли можно ожидать в ближайшие годы повышения производительности труда на порядок ***при создании полностью новых, сложных программных продуктов***. Еще более консервативна длительность таких разработок.

Принципиальным путем улучшения ТЭП при разработке сложных ПС является исключение творчества на тех этапах, где возможны ***типовые, стандартные решения*** и апробированные заготовки, не требующие при их применении высококвалифицированного творческого труда. Основой такого подхода является применение унифицированной технологии, готовых испытанных компонентов и стандартизированной архитектуры определенных классов ПС. Использование готовых апробированных модулей почти исключает творческий труд по их программированию, автономной отладке и документированию. На этих этапах творческие усилия необходимы только для отбора готовых компонентов и разработки новых, отсутствующих среди апробированных. Однако практически полностью

сохраняется творческий труд при системном анализе, при комплексировании компонентов и их комплексной отладке, а также во время испытания ПС в целом (см. лекцию 14).

Для реализации сложных проектов ПС наиболее часто применяются ***две схемы организации коллективов специалистов***:

— формирование для выполнения каждого проекта жесткой организационной структуры целостного коллектива с полным составом необходимых специалистов под единым, централизованным руководством лидера проекта;

— выделение руководителя (главного конструктора) и небольшой группы интеграторов, по заданиям которых выполняются частные работы узкими специалистами по компонентам, не входящими организационно в единый коллектив для реализации каждого конкретного крупного проекта.

Первая схема предпочтительна, когда фирма реализует небольшое число особенно крупных проектов-заказов и имеет возможность для каждого из них скомплектовать полноценную, организационно замкнутую, «***команду***». Она полностью реализует проект и несет ответственность за его качество. Однако при этом возможны простои отдельных специалистов из-за несинхронного ожидания заданий или результатов последовательных этапов проектирования компонентов другими специалистами. ***Вторая схема*** для фирмы может иметь преимущества при большом числе относительно небольших проектов, близких по содержанию и функциональному назначению компонентов. В этом случае большинство специалистов одновременно участвуют в нескольких заказах по локальным заданиям лидеров и интеграторов различных проектов и могут использоваться более полно. Однако задачи интеграторов при этом усложняются и требуют более высокой квалификации. Хотя за качество проекта в целом также несут ответственность руководитель-лидер и группа интеграторов, усложняется взаимодействие с поставщиками компонентов и руководство их качеством.

Для реализации мероприятий по планированию и управлению жизненным циклом концептуально целостных, крупных ПС и обеспечения их качества необходимы организационные действия системных архитекторов, направленные на подбор и обучение коллектива специалистов разных категорий и специализаций (см. рис. 9.2). Концепция — ключевой, исход-

ный документ сложного жизненного цикла ПС. Он непосредственно ориентирован на решение проблемы комплексной реализации требований и является документом, к которому можно обратиться в любой момент, чтобы увидеть, что продукт или система должны делать.

Практически в каждом успешном проекте ***должен быть или был лидер***. Лидером продукта может быть: менеджер продукта, менеджер проектирования, руководитель проекта. Лидер должен:

— руководить процессом выявления и формирования требований заказчика;

— рассматривать конфликтующие пожелания, поступающие от различных участников проекта, и находить компромиссы, необходимые для определения набора функций, представляющих наибольшую ценность для максимального числа участников;

— вести переговоры с заказчиком, руководством, пользователями и разработчиками и поддерживать равновесие между тем, чего хочет заказчик, и тем, что может предоставить команда разработчиков за ресурсы и время, отведенные заказчиком для реализации проекта;

— осуществлять проверку спецификаций программного средства, чтобы удостовериться, что они соответствуют реальной концепции, представленной детальными функциями;

— осуществлять управление изменением приоритетов задач, а также добавлением и исключением функций.

История развития разработок программных продуктов — это история роста их масштабов. Крупные проекты, как правило, требуют координированной работы больших коллективов и многих команд. Возрастание сложности снижает способность человека решать задачи интуитивно по мере их возникновения. Чтобы добиться успеха в большом проекте, ***необходима четкая координация действий «команды»***, которая должна работать по общей методологии, чтобы решить проблему комплекса требований и качества. Успешное управление требованиями заказчика может осуществляться только эффективно организованной командой разработчиков.

Одним из наиболее важных факторов является то, что члены команды имеют различные, профессиональные навыки и квалификацию. Поэтому трудно ожидать, что некий универсальный способ организации команды будет во всех случаях предпочтительнее, чем альтернативные варианты.

Тем не менее определенные общие элементы присутствуют во многих успешных командах. Поэтому важно рассмотреть некую гипотетическую структуру команды.

Руководство крупным проектом ПС должны осуществлять один или два лидера — менеджера (см. рис. 9.2):

— ***менеджер проекта*** — это специалист, обеспечивающий коммуникацию между заказчиком и проектной командой, его задача — определить и обеспечить удовлетворение требований заказчика;

— ***менеджер-архитектор комплекса программ*** — управляет коммуникациями и взаимоотношениями в проектной команде, является координатором создания компонентов, разрабатывает базовые, функциональные спецификации и управляет ими, ведет график проекта и отчитывается за его состояние, инициирует принятие критичных для хода проекта решений.

В реализации крупного проекта можно выделить две категории специалистов: разрабатывающих компоненты и ПС в целом и обеспечивающих технологию и качество программного продукта. Организационное разделение специалистов, осуществляющих разработку ПС (первая категория), и специалистов, контролирующих и управляющих его качеством в процессе разработки и всего ЖЦ (вторая категория), должно обеспечивать независимый, достоверный контроль качества результатов разработки и эффективное достижение заданных характеристик.

Специалисты первой категории непосредственно создают компоненты и ПС в целом с заданными показателями качества. В процессе разработки их функции заключаются в тщательном соблюдении принятой в фирме технологии и в формировании всех предписанных руководствами исходных и отчетных документов. При этом предполагается, что выбранная технология способна обеспечить необходимые значения конструктивных показателей качества, а достижение заданных функциональных характеристик гарантируется тематической квалификацией соответствующих специалистов и регулярным контролем этих характеристик в процессе разработки. Система стандартизированного документирования частных работ должна обеспечить объективное отражение качества компонентов и процессов их создания на всех этапах ЖЦ ПС (см. лекцию 17).

Разделение труда специалистов этой категории в крупных проектных коллективах приводит к необходимости их дифференциации по квалификации и областям деятельности:

— ***спецификаторы*** подготавливают описания функций соответствующих компонентов с уровнем детализации, достаточным для корректной разработки текстов программ программистами и их интерфейсов;

— ***разработчики программных компонентов — программисты*** создают компоненты, удовлетворяющие спецификациям, реализуют возможности продукта, отслеживают и исправляют ошибки, при разработке сложных систем это требует детального знания высокоуровневых языков программирования, визуального программирования, сетевых технологий и проектирования баз данных;

— ***системные интеграторы*** сложных проблемно-ориентированных ПС работают над проектами в значительной степени отличными от программистов методами, на разных языках проектирования, используют различные средства автоматизации и имеют на выходе различные результаты крупных компонентов и комплексов программ;

— ***тестировщики*** обеспечивают проверку функциональных спецификации, систем обеспечения производительности, пользовательских интерфейсов, разрабатывают стратегию, планы и выполняют тестирование для каждой из фаз и компонентов проекта, должны быть административно независимыми от программистов и спецификаторов;

— ***управляющие сопровождением и конфигурацией, инструкторы интерфейсов*** отвечают за снижение затрат на модификацию и сопровождение продукта, обеспечение максимальной эффективности работы разработчиков по взаимодействию компонентов и реализации версий ПС, принимают участие в обсуждениях пользовательского интерфейса и архитектуры продукта;

— ***документаторы*** процессов и объектов ЖЦ ПС обеспечивают подготовку и издание сводных технологических и эксплуатационных документов в соответствии с требованиями стандартов.

Успех и качество при разработке сложных программных комплексов все больше зависит от слаженности работы и профессионализма коллектива этой категории специалистов на всех этапах и уровнях создания таких проектов. При выборе заказчиком надежного поставщика — разработчика проекта необходима ***оценка тематической и технологической квалификации*** возможного коллектива специалистов, а также его способности реализовать проект с заданными требованиями и качеством. Тематическую

квалификацию специалистов в области создания ПС определенного функционального назначения приближенно можно характеризовать средней продолжительностью работы в данной проблемной области основной части команды, непосредственно участвующей в разработке алгоритмов, спецификаций, программ и баз данных. Важнейшую роль при этом играет квалификация руководителей — лидеров разработки и системных аналитиков функциональных компонентов и в меньшей степени непосредственных разработчиков программ в конкретной прикладной области. Особенно важна не индивидуальная характеристика каждого специалиста, а прежде всего интегральный показатель ***квалификации «команды»***, реализующей некоторую, достаточно крупную функциональную задачу или весь проект. При низкой тематической квалификации допускаются наиболее грубые системные ошибки, требующие больших затрат при доработке программ или даже делающие проект практически не реализуемым.

Технологическая квалификация коллектива характеризуется опытом и длительностью работы с регламентированными технологиями, инструментальными комплексами автоматизации разработки, языками проектирования, программирования и тестирования ПС. Особое значение имеет коллективный опыт организации и выполнения сложных проектов на базе современных автоматизированных технологий и инструментальных средств. Опыт применения конкретного комплекса автоматизации и языков проектирования ПС может являться существенным фактором при выборе технологии для создания новых компонентов и обеспечении качества ПС.

В ***детальной модели COCOMO*** (см. лекцию 5) значительное внимание уделено ***влиянию организации и взаимодействия коллектива разработчиков*** на трудоемкость создания сложных программных средств — таблица 9.1. В составе организационных характеристик коллектива рекомендуется учитывать согласованность целей специалистов, участвующих в проекте, их психологическую совместимость и способность к дружной коллективной работе, наличие опыта работы в данном коллективе и другие объективные и субъективные свойства участников проекта. При этом большое значение могут иметь личная мотивация и психологические особенности поведения разных специалистов при комплексной работе над сложным проектом. Эти характеристики могут быть обобщены в качественный показатель влияния сложности взаимодействия специалистов в

коллективе, которому сопоставлены коэффициенты изменения трудоемкости разработки ПС (последняя строка в таблице 9.1). Наилучшим считается непрерывное корректное взаимодействие организованных специалистов с большим опытом работы в данном коллективе при полной согласованности их целей, планов и методов работы. В остальных случаях в той или иной степени (даже в 3—5 раз) может возрастать трудоемкость разработки ПС, что нельзя не учитывать при прогнозировании ТЭП и обосновании разработки крупных проектов.

Таблица 9.1

Характеристики и влияние коллективизма разработчиков программных средств на трудоемкость

Коллективизм	Значение характеристики				
	Очень низкий	Низкий	Номинальный	Высокий	Очень высокий
Согласованность целей коллектива	Минимальная	Незначительная	Относительная	Значительная	Полная
Способность членов коллектива адаптироваться к целям других	Малая	Незначительная	Относительная	Значительная	Полная
Опыт работы в составе данного коллектива	Нет	Малый	Незначительный	Значительный	Большой
Степень доверия и взаимодействия в коллективе	Нет	В малой степени	В некоторой степени	Значительная	Большая
Обобщенная коллективность работ	Некоторое взаимодействие в коллективе	Сложное взаимодействие	Зачастую коллективная работа	Высокая степень взаимодействия	Непрерывное взаимодействие
Обобщенный коэффициент влияния коллективности работ на трудоемкость	5,48	4,38	3,29	1,10	0,00

Специалисты второй категории — технологи, обслуживающие и сопровождающие технологический инструментарий, который применяется специалистами первой категории, обеспечивают применение системы качества проекта или предприятия, контролируют и инспектируют ее использование (см. рис. 9.2). Основные задачи второй категории специалис-

тов должны быть сосредоточены на контроле процессов и результатов выполнения работ и на принятии организационных и технологических мер для достижения их необходимого качества, обеспечивающего выполнение всех требований технического задания на ПС.

Технологи должны выбирать, приобретать и осваивать наиболее эффективный инструментарий для проектов, реализуемых конкретной фирмой с учетом особенностей создаваемых ПС требуемого качества и рентабельности технологических средств. Они должны разрабатывать регламентированный технологический процесс и систему качества, поддерживающие весь ЖЦ ПС и обучать разработчиков ПС квалифицированному применению соответствующих инструментальных средств и технологий.

Специалисты, управляющие обеспечением качества ПС, должны овладеть стандартами и методиками фирмы, поддерживающими регистрацию, контроль, документирование и воздействия на показатели качества на всех этапах ЖЦ программ. Они должны обеспечивать эксплуатацию системы качества проекта, выявление всех отклонений от заданных показателей качества объектов и процессов, а также от предписанной технологии на промежуточных и заключительных этапах разработки. Эти же специалисты должны анализировать возможные последствия выявленных отклонений от требований технического задания или спецификации на ПС. В результате должны приниматься меры либо по устранению отклонений, либо по корректировке требований, если устранение отклонений требует чрезвычайно больших ресурсов.

Инспекторы-испытатели по проверке систем качества предприятия и качества программных продуктов должны пройти обучение, дающее им знания и квалификацию, необходимые для проведения испытаний, оценки их результатов и эффективности применения систем качества. Для них (см. **ISO 10011-2)** необходимыми считаются знание и понимание стандартов и нормативных документов, в соответствии с которыми должны осуществляться оценки применения систем качества, а также обследование, анкетирование и составление отчетов по испытаниям. Инспектор-эксперт, в соответствии со стандартом, должен быть непредубежденным; обладать здравым смыслом; иметь аналитический склад ума и твердость воли; обладать способностью реально оценивать сложные действия с точки зрения их дальнейшей перспективы в рамках общей организационной структуры:

— получать и справедливо оценивать объективные данные испытаний характеристик продуктов и систем качества при незаинтересованном проведении проверок;

— относиться к персоналу системы качества, подвергающемуся проверке, таким образом, чтобы получить наилучшие результаты;

— приходить к приемлемым для разработчиков и заказчика выводам, основанным на реальных наблюдениях в процессе испытаний характеристик качества ПС.

Перечисленные выше специализации и квалификации персонала, участвующего в крупных проектах ПС, ***требуют соответствующей их подготовки, отбора и непрерывного обучения***, которые являются самостоятельной, важной проблемой проектирования и всего ЖЦ комплексов программ. Обучение представляет собой процесс и требует организации и сопровождения обучаемого персонала. Должны быть разработаны и документированы планы, требования и цели обучения, а также разработаны руководства, включая материалы, используемые для обучения (см. шестую часть стандарта **ISO 15504**). Персонал, ответственный за выполнение конкретных задач, если это необходимо, должен быть ***аттестован*** на основе соответствующего образования, подготовки и/или опыта работы. Может также стать необходимым включение в подготовку, ознакомление со специфической (проблемно-ориентированной) областью, в которой будет работать данное программное средство, и повышение квалификации в этой области. Требования к квалификации, обучению персонала и их реализации должны быть документально оформлены в системном проекте.

Совещания предоставляют заинтересованным специалистам из различных команд и организаций возможность работать вместе над достижением общей цели крупного проекта. Надлежащая подготовка является залогом успеха совещаний. Первым делом необходимо распространить идею внутри организации, разъясняя преимущества проведения совещаний членам команды. Подготовка включает в себя также выявление заинтересованных лиц, которые могут повлиять на процессы ЖЦ ПС и чьи потребности необходимо учесть, чтобы гарантировать успешный результат.

Необходимо заранее разослать подготовительные материалы, чтобы подготовить участников, а также повысить производительность проводимого совещания. Подготовительные материалы должны стимулировать как

конкретное, так и свободное мышление. Для гарантии успеха рекомендуется, чтобы совещание проводилось сторонним человеком, имеющим опыт в решении уникальных задач управления. Если совещание проводится членом команды, этот человек вначале не должен вносить свои идеи и участвовать в обсуждении. Иначе существует опасность, что совещание утратит необходимую для получения реальных фактов объективность и не будет способствовать созданию атмосферы, в которой можно достигнуть консенсуса. В любом случае специалист, ведущий встречи, играет ***ключевую роль в успехе совещания*** и должен:

— установить правила проведения встречи и добиваться их выполнения;

— управлять течением дискуссии и удерживать команду на главной цели совещания;

— способствовать процессу принятия решения и достижения консенсуса, но избегать участия в содержательной части дискуссии;

— удостовериться, что все заинтересованные лица участвуют и их пожелания учтены;

— контролировать поведение участников, которое может привести к расколу или мешает продуктивной работе.

Следует ***активно привлекать заказчиков*** к совещаниям, управлению их требованиями и масштабом проекта, чтобы обеспечить как качество, так и своевременность разработки ПС. Именно заказчики несут финансовую ответственность за выполнение внешних обязательств перед пользователями. Необходимы указания заказчиков при принятии основных решений, и только они могут реально определить, как, сократив функции ПС, получить полезный комплекс программ высокого качества, выполненный в срок и в пределах бюджета. Исключенный из процесса принятия решений заказчик будет недоволен и, естественно, будет стремиться обвинить разработчиков в недостаточной старательности. Привлечение заказчика помогает наименее болезненно решить проблемы управления масштабом и функциями проекта.

Для защиты как проекта, так и бизнес-целей заказчика может понадобиться вести ***переговоры об объеме работ для команды***. Во время переговоров с заказчиком о техническом задании и требованиях базового уровня следует руководствоваться принципом меньше обещать и больше делать.

Тогда неизбежные издержки разработки ПС (непредвиденные технологические риски, изменения требований, задержки при приобретении закупаемых компонентов, непредвиденный уход основных членов команды и т.п.) не приведут к нарушению графика вашего проекта. Однако заказчики, внешние или внутренние, естественно, желают получить как можно больше функциональных возможностей в каждой версии ПС. Именно эти функциональные возможности создают добавленную стоимость, которая важна им для достижения их бизнес-целей. Следует с пониманием относиться к требованиям клиентов, поскольку именно они, в конечном счете, добиваются успеха на рынке. Однако если пойти навстречу требованиям заказчика повышения функциональности, это может негативно сказаться на качестве и общей жизнеспособности проекта.

Обычно наиболее важным для реализации проекта ПС и зависящим от большинства его особенностей и факторов является ***трудоемкость, непосредственно определяющая стоимость*** создаваемого комплекса программ. Значения длительности разработки и числа специалистов взаимосвязаны и в некоторых пределах могут размениваться. Поэтому оценки этих показателей затрат можно варьировать, и при недостаточном числе специалистов, естественно, возрастает длительность разработки, хотя трудоемкость может остаться практически неизменной. Многократное применение одних и тех же апробированных компонентов и/или адаптация ПС к различным условиям применения является одним из ***перспективных методов повышения качества и снижения затрат труда специалистов*** в жизненном цикле сложных комплексов программ.

9.3. Ресурсы для обеспечения функциональной пригодности при разработке сложных программных средств

При проектировании ПС необходимо учитывать, что экономические, временные, вычислительные и другие ресурсы ***на разработку и весь ЖЦ программ*** всегда ограниченны и используемые затраты для улучшения каждой характеристики должны учитывать эти ограничения. Для рационального распределения этих ресурсов необходимо знать, ***как отражается изменение затрат на улучшении каждой характеристики качест-***

ва ПС. Эта взаимосвязь затрат ресурсов и значений каждой характеристики зависит от назначения, а также от ряда свойств и других особенностей комплекса программ, что усложняет учет влияния таких связей. Тем не менее выявлены основные тенденции такого взаимодействия, которые могут служить ***ориентирами*** при выборе и установлении требований к определенным характеристикам качества в конкретных проектах ПС.

Обеспечение функциональной пригодности является основной целью при использовании финансовых, трудовых, вычислительных и других ресурсов в жизненном цикле ПС. Однако это не значит, что затраты на решение этой основной задачи всегда являются доминирующими по величине. Необходимость выполнения ряда требований к остальным, конструктивным характеристикам качества часто приводит к тому, что использование ресурсов на их реализацию может превышать базовые затраты на обеспечение функциональной пригодности. В то же время затраты на выполнение этих требований всегда направлены на повышение и совершенствование функциональной пригодности. Поэтому обычно трудно четко выделить и количественно оценить все виды затрат, используемых только на функциональную пригодность.

В любом программном средстве можно выделить компоненты, в которых сосредоточены функциональные алгоритмы и программы, предназначенные для решения основных целевых задач ПС. В некоторые из них органически входят компоненты для повышения качества решения функциональных задач или они построены с учетом требований высокого качества результатов основных функций. Для анализа затрат ресурсов в жизненном цикле ПС при проектировании их целесообразно разделить на ***две части***:

— затраты на создание программных компонентов, обеспечивающих ***базовые свойства функциональной пригодности*** комплекса программ для его применения по прямому назначению пользователями, в соответствии с требованиями контракта и технического задания;

— основные составляющие ***дополнительных затрат***, обеспечивающие требуемые конструктивные характеристики качества для улучшения функциональной пригодности ПС в соответствии с целями и сферой его применения.

Кроме того, следует учитывать совокупность затрат ресурсов на технологию, инструментарий автоматизации разработки и систему качества,

обеспечивающие жизненный цикл ПС. Эта составляющая затрат зависит не только от характеристик проектируемого ПС, но и от интенсивности применения технологических средств (см. рис. 9.1).

Затраты на обеспечение функциональной пригодности зависят, в первом приближении, от сложности алгоритмов, объема комплекса программ и баз данных, которые определяют затраты труда и длительность полного цикла их разработки. Основные затраты идут на овеществленный, преимущественно интеллектуальный, труд специалистов различных категорий. Поэтому для их измерения наиболее универсальной единицей стали трудозатраты специалистов в человеко-днях или человеко-месяцах, которые обычно достаточно просто могут преобразовываться в стоимость процесса разработки.

Для учета классов крупномасштабных ПС с позиции затрат на их разработку проведено ранжирование экспериментальных данных и выделены классы (см. **ISO 12182**), наиболее сильно отличающиеся затратами на функциональную пригодность, ***при одном и том же общем размере полностью новых программ и информации баз данных***, которые характеризуются (см. лекцию 5):

— максимальной трудоемкостью — ПС систем управления динамическими объектами или процессами в реальном времени (СРВ);

— средней трудоемкостью — ПС административных, организационных и информационно-поисковых систем (ИПС);

— минимальной трудоемкостью создания — пакеты автономных расчетных прикладных программ (ППП).

Все остальные типы ПС могут быть упорядочены между выделенными классами и для них получены оценки изменения трудоемкости (стоимости) относительно максимальной для ПС реального времени. Малые проекты, создаваемые небольшими коллективами специалистов, характеризуются большим коэффициентом вариации — разбросом значений предполагаемой трудоемкости, вследствие высокой роли индивидуальных способностей отдельных специалистов. Трудоемкость разработки крупномасштабных ПС объемом свыше 100 тысяч строк отражается более определенными закономерностями и стабильными значениями, что обусловлено усреднением творческих возможностей специалистов и условий их труда в больших коллективах, а также возрастанием доли затрат и роли руководящего и вспомогательного персонала в ЖЦ ПС.

Основные составляющие ***дополнительных затрат***, обеспечивающие требуемые характеристики качества ПС, целесообразно структурировать в соответствии с их номенклатурой в стандарте **ISO 9126** (см. лекцию 11). Однако желательно эти затраты связать с процессами ЖЦ ПС. Важнейшее значение имеют установление и ***формализация исходных требований к характеристикам ПС***. Поэтому целесообразно выделять значительные ***ресурсы на системный анализ и проектирование требований*** ко всему комплексу характеристик и их атрибуты на начальных этапах проекта ПС (см. рис. 9.1). На этих этапах неопределенность оценки влияния факторов наибольшая, тем не менее даже приблизительный их учет позволяет избегать грубых ошибок (в несколько раз) при оценке затрат на разработку, которые делаются экспертами без детального анализа влияния различных факторов на требуемое качество ПС. Подобный анализ может быть базой для рационального, первичного распределения ограниченных ресурсов разработки и для управления их использованием по мере развития проекта. Учет возможного изменения затрат в зависимости от основных параметров проекта способствует упорядочению процесса разработки ПС и концентрации усилий на тех факторах и затратах, которые могут дать максимальный эффект в повышении качества, при конкретных условиях создания программ и реальных ограничениях ресурсов.

После проведения системного проектирования, а также предварительного распределения ресурсов возможна ошибка в оценке совокупных затрат на разработку ПС с требуемым качеством, приблизительно в полтора-два раза. Разработка архитектуры комплекса программ на основе прототипов и подготовка к программированию готовых апробированных алгоритмов позволяет ее уменьшить до 20—30%. Такую достоверность можно получить, конечно, только при подробном анализе и оценке влияния важнейших факторов и требований к качеству конкретного ПС (см. лекцию 5).

Крупные затраты, достигающие 30% от полных затрат на разработку сложных ПС, могут приходиться на ***верификацию и тестирование программных компонентов***, что должно обеспечивать корректность и надежность ПС в целом. Хотя эти характеристики и их атрибуты принципиально различаются, ресурсы на их реализацию полностью разделить невозможно. Поэтому оценивание и выделение ресурсов на решение этих задач целесообразно анализировать совместно.

Затраты на квалификационное тестирование и испытания ПС в целом обычно могут быть достаточно четко выделены из остальных ресурсов, так как в этих процессах непосредственно участвуют заказчик и пользователи. Величина этих затрат, без учета ресурсов, необходимых для имитации внешней среды, может составлять около 10% от общих затрат на разработку. При этом практически невозможно разделять затраты на оценивание отдельных стандартизированных характеристик и их атрибутов.

Затраты на обеспечение безопасности и надежности функционирования ПС определяются требуемым уровнем защищенности и сложностью (размером) программ для ее реализации (см. лекцию 11). При наличии особенно высоких требований к безопасности критических ПС эти затраты могут даже в 2—4 раза превышать затраты на решение базовых, функциональных задач. Для типовых административных систем трудоемкость создания программных средств защиты обычно составляет 20—40% затрат на решение основных, функциональных задач. В более простых случаях доля таких затрат может снижаться до 5—10%. Затраты на обеспечение высокой ***надежности*** в составе разработки ПС могут достигать 2—3-кратного увеличения общих затрат, при высоких требованиях наработки на отказ. Для минимального обеспечения автоматического рестарта в ординарных системах они составляют порядка 10—20%. Однако практически в любых системах должен присутствовать минимум программных компонентов, обеспечивающих надежность и защиту от преднамеренных и случайных угроз.

Затраты на создание достаточно ***полного комплекта документации*** практически пропорциональны размеру комплекса программ. Удельные затраты на документацию зависят от класса, назначения и широты применения программ, что трудно учесть в достаточно общем виде. Эти затраты сопутствуют в некоторой степени всем этапам разработки, однако оформление эксплуатационных документов обычно локализуется в специальном этапе работ. Затраты на разработку комплекта ***эксплуатационной документации*** для сложных программных продуктов, подлежащих длительному сопровождению, обычно определяются затратами ***ориентировочно*** 5—10 страниц на тысячу строк текста программы.

Достаточно определенно могут быть выделены и оценены ***затраты на обеспечение и реализацию требований к характеристикам***: защи-

щенности, сопровождаемости, мобильности и практичности (последние в части документирования) ПС. Эти затраты состоят из двух связанных частей: ***затрат на реализацию*** соответствующих характеристик качества в программных продуктах и ***затрат при использовании*** этих характеристик в процессе эксплуатации комплекса программ. Обычно совершенствование качества и повышение затрат на реализацию характеристик способствует снижению затрат при их эксплуатации. Последние трудно оценить априори, так как они зависят от внешней среды и активности применения конкретного ПС, а не от его свойств и требуемого качества. Поэтому далее основное внимание акцентировано на затратах, которые необходимы для достижения требуемых характеристик качества.

При анализе затрат на обеспечение требуемых функций крупномасштабных ПС ***доминирующим фактором***, влияющим на их величину, является ***сложность функциональной части*** комплекса программ и базы данных. Понятие сложности программ активно исследовалось последние десятилетия, и предложен ряд показателей и методов для ее измерения. Наибольшее внимание исследователей привлекала статистическая мера сложности и мера структурной сложности программ. Для относительно небольших программ эти методы позволили повысить достоверность определения сложности программ и установить адекватность этого показателя величине трудоемкости их создания. Однако для ПС средней и высокой сложности ряд дополнительных факторов затрудняет подобные оценки. В крупных ПС обычно реализуются задачи различной сложности. При наличии особо сложной функциональной задачи возрастание затрат на ее реализацию может нивелироваться рядом других типовых и более простых задач. Поэтому пока не проявились преимущества приведенных мер сложности и они не нашли широкого практического применения.

Наиболее активно в качестве простейшего показателя сложности используется ***размер — масштаб комплекса программ, выраженный числом*** операторов (команд) или строк текста на языке программирования (с учетом коэффициента, зависящего от класса ПС и специфики языка) (см. лекцию 5). Размер программ без комментариев является одной из наиболее достоверно измеряемых характеристик сложности ПС и достаточно адекватен экономическим затратам на его разработку. Реальное изменение создаваемых в настоящее время ***новых сложных ПС*** объемом от 10^3 до

10^6 строк определяет диапазон трудоемкости разработки таких программ от человеко-года до тысяч человеко-лет. С другой стороны, отсутствуют какие-либо данные о значительном преимуществе других достаточно простых мер сложности при прогнозировании ресурсов на разработку крупномасштабных ПС.

Ограниченные ресурсы времени реализации проектов ПС являются одним из самых ***сильных факторов***, влияющих на достижимое качество комплексов программ. Избыточный оптимизм многих разработчиков и давление заказчиков относительно сроков реализации контрактов на ПС негативно отражается, прежде всего, на характеристиках качества ПС. Поэтому при выборе и формализации в контракте значений требуемых характеристик качества ПС особое внимание следует обращать на возможность их достигнуть в согласованные сроки. Чем сложнее комплекс программ, а соответственно, чем выше в нем вероятность ошибок и дефектов, тем больше времени требуется для их устранения и на весь процесс разработки. При реальном допустимом времени реализации проекта оно ограничивает затраты на все промежуточные этапы работ, и тем самым на качество их выполнения.

При современных технологиях полностью новые, крупномасштабные комплексы программ, реализуемые в допустимое время 2—4 года, ограничены объемом 10^6— 10^7 строк текста. Выше отмечалось (см. лекцию 5), что диапазону размеров современных ПС в три-четыре порядка (до 10 млн строк) соответствуют приблизительно такие же диапазоны изменения трудоемкости и стоимости их разработок. Очевидна принципиальная нерентабельность разработки очень сложных ПС за 5—10 лет. С другой стороны, даже относительно небольшие программы высокого качества в несколько тысяч строк по полному технологическому циклу с сертификационными испытаниями продукции редко создаются за время, меньшее чем полгода-год. Таким образом, диапазон вариации длительностей разработок ПС много меньше, чем вариация их трудоемкости, а эти длительности ограничены сверху и снизу, и объем новых программ является одним из основных факторов, определяющим эти границы.

Размер базы данных ПС в некоторой степени коррелирован с размером текстов программ. Это привело к тому, что в ряде экономических исследований размер базы данных либо совсем не учитывался, либо вклю-

чался в объем ПС. В статистической теории сложности программ показано, что для программных модулей и относительно небольших групп программ имеется корреляция числа имен переменных и операторов в программе. Однако для крупных ПС корреляция может быть меньше. Это определило необходимость ***разделения ПС на два типа***: на осуществляющие преимущественно сложную логическую обработку относительно небольшого потока данных и на информационно-поисковые системы при наличии больших объемов информации баз данных и при относительно простой их обработке. Степень этого влияния трудно формализовать, так как большую роль играет структура базы данных и ее функциональное назначение. Поэтому обычно этот фактор отдельно не учитывается и только для очень больших и сложных структур баз данных рекомендуется увеличивать оцениваемую трудоемкость разработки.

9.4. Ресурсы на реализацию конструктивных характеристик качества программных средств

Имеющийся опыт показывает, что кроме функциональной пригодности и мобильности большинство факторов может изменять трудоемкость процессов разработки программ на десятки процентов и не более чем в 2—3 раза. Приводимые ниже экспертные оценки относятся к разработке полностью нового, крупного ПС, без использования готовых программных компонентов. Эти оценки ***могут служить ориентирами***, которые должны напоминать разработчикам, что каждое повышение требований к качеству ПС реализуемо за счет дополнительных ресурсов, которые могут быть соизмеримыми или даже превышать затраты на решение основных, функциональных задач.

Анализ затрат на ***обеспечение корректности*** зависит от полноты прослеживания реализации требований к ПС сверху вниз, в требованиях к компонентам вплоть до объектного кода программ и от степени их покрытия тестами. Эти затраты входят непосредственно в процесс разработки и зависят от объема и детальности процессов верификации и тестирования. Для сложных ПС при требовании их высокой корректности они могут составлять до 30% от затрат на обеспечение первичной, функциональной

пригодности. Для относительно простых комплексов программ эта величина в среднем не превышает 20%.

Затраты на взаимодействие программных средств и их компонентов с внутренней и внешней средой определяются сложностью (объемом) программ и затратами производительности ЭВМ, реализующими эти функции. Они включают затраты на визуализацию информации, обеспечивающей функциональную пригодность, телекоммуникацию с внешними абонентами системы и с операционной системой, и могут составлять 10—20% затрат на обеспечение основных функций ПС.

Особенности затрат на реализацию остальных требований к конструктивным характеристикам качества отмечаются при представлении соответствующих характеристик (см. лекцию 11) и сводятся к следующему:

— дополнительные затраты на обеспечение высокой ***надежности ПС*** могут достигать 2—3-кратного увеличения затрат относительно функциональной пригодности при требованиях наработки на отказ в десятки тысяч часов, а для минимального обеспечения автоматического рестарта в ординарных системах составляют порядка 10—20%;

— для повышения ***эффективности использования ресурсов*** ЭВМ затраты могут быть относительно невелики (несколько процентов) и их трудно выделить из затрат на решение основных, функциональных задач;

— затраты на обеспечение ***практичности*** зависят в основном от сложности применения ПС, от качества и количества эксплуатационной документации и электронных учебников и могут составлять до 20% затрат на решение основных, функциональных задач.

Стремление уменьшить затраты в период разработки без учета последующего использования ПС, его компонентов и всего жизненного цикла может оказаться мало полезным, а в некоторых случаях привести к значительному увеличению совокупных затрат в ЖЦ. При применении сложных ПС эти затраты исчисляются сотнями человеко-лет, что определяет особую актуальность снижения этих затрат. Поэтому необходим системный анализ распределения и использования ресурсов на разработку программ ***с учетом всего их жизненного цикла***, включая сопровождение и перенос на другие платформы. При использовании приведенных данных необходимо учитывать, что они могут служить ***только ориентирами*** при приближенной оценке затрат на непосредственную разработку и обеспечение качества конкретного ПС.

Сопровождаемость ПС можно оценивать ***потребностью трудовых и временных ресурсов*** для ее обеспечения и для реализации. Возможные затраты ресурсов на развитие и совершенствование качества комплекса программ зависят не только от внутренних свойств программ, но также от запросов и потребностей пользователей на новые функции и от готовности заказчика и разработчика удовлетворить эти потребности (см. лекцию 15).

В современных проектах ПС большую или меньшую долю составляют ***готовые апробированные компоненты*** из других подобных разработок — ***прототипы и/или покупные пакеты прикладных программ***. Это позволяет значительно ускорять работы и сокращать затраты на создание сложных комплексов программ. Перед разработчиками проекта ПС зачастую возникает дилемма: разрабатывать ли весь комплекс программ полностью из новых компонентов или использовать, адаптировать и приспосабливать готовые компоненты, какие и в каком количестве. В результате при первичном экономическом анализе ресурсов на создание ПС с требуемыми характеристиками качества целесообразно рассматривать ***два альтернативных варианта*** определения затрат на разработку:

— полностью нового ПС, для которого отсутствуют или недоступны подходящие готовые компоненты — прототипы и/или их заведомо нерентабельно использовать;

— программного продукта на базе комплексирования набора готовых программных компонентов и информации баз данных, для которого почти не требуется создания новых компонентов.

Мобильность и затраты конкретных ресурсов для переноса программ и данных на иные аппаратные и операционные платформы при проектировании могут быть учтены только очень приблизительно. Эти субхарактеристики включают адаптируемость, простоту установки и замещаемость программ, которые целесообразно ***оценивать количественно***: совокупными затратами, стоимостью, трудоемкостью и длительностью на реализацию процедур переноса программ и данных. Оценки мобильности зависят не только от внутренних субхарактеристик ПС, но также от организации, технологии и документирования реализации жизненного цикла и процессов переноса комплексов программ и их компонентов (см. лекцию 15).

При переносе программ и данных свойства систем практически всегда изменяются, что следует учитывать при системном анализе целесооб-

разности и эффективности переноса, а также могут быть необходимы тестирование, испытания и сертификация получаемых ПС в новой среде. Кроме того, ***любой перенос связан с затратами***. При создании мобильных ПС трудно предусмотреть все возможные особенности и различия платформ и внешней среды, для которых декларируется мобильность конкретных программных средств. Эти особенности и возможное расширение свойств окружающих прикладных программ и данных могут преподносить неприятные сюрпризы нестыковки, для ликвидации которых потребуются дополнительные ресурсы. Требования мобильности компонентов ПС приводят к необходимости их проектирования как автономных комплектующих изделий. Это реализуется за счет стандартизации их построения и организации интерфейсов, унификации структуры базы данных и гарантии качества функционирования. В результате подготавливается возможность создания новых ПС из набора готовых программных модулей или функциональных групп программ, ранее использованных и испытанных в другом сочетании при решении несколько иных функциональных задач. Однако в любом комплексе программ вряд ли возможно и нужно повторно использовать все 100% готовых программных модулей. При сборке нового ПС из комплектующих компонентов, может потребоваться доработка некоторых из них или создание специальных программ для их взаимодействия.

В ряде случаев целесообразна настройка — ***адаптация*** готовых компонентов на новые условия применения. При больших изменениях условий применения эффективной может оказаться сборка нового ПС с частичным использованием апробированных компонентов и с разработкой небольшой части программных модулей, обеспечивающих новые функции, интерфейсы и условия применения ПС. При этом относительное снижение затрат пропорционально доле использования готовых комплектующих компонентов и степени их пригодности для использования в новом ПС. В пределе при построении нового ПС на 80—90% из готовых компонентов суммарные затраты могут сокращаться в 3—5 раз. В этом случае, кроме 10—20% затрат на создание новых программных компонентов, необходимы ресурсы на комплексирование нового ПС, его квалификационное тестирование, испытания и документирование.

Практически всегда ***необходимы время и трудоемкость на системный анализ*** и оценивание целесообразности разработки комплекса про-

грамм из готовых компонентов с учетом стоимости приобретения и адаптации переносимых программ. Интегрирование компонентов в новой операционной среде, тестирование и испытания в комплексе с унаследованными программами также почти всегда требуют времени и других ресурсов. Некоторые, казалось бы, второстепенные, детали и факторы несовместимости готовых, переносимых программ и новой платформы могут заметно отражаться на трудоемкости проекта. Особенно тщательно их следует анализировать в унаследованных системах, в которых относительно мелкие детали взаимодействия новых и старых программ могут существенно влиять на экономические характеристики проекта.

Обычно наиболее важным для реализации проекта и зависящим от большинства его особенностей и факторов является трудоемкость, непосредственно определяющая стоимость создаваемого комплекса программ. Значения длительности разработки и числа специалистов взаимосвязаны и в некоторых пределах могут разменяваться. Поэтому оценки этих показателей можно варьировать, и при недостаточном числе специалистов, естественно, возрастает длительность разработки, хотя трудоемкость может остаться практически неизменной. Многократное применение одних и тех же апробированных компонентов и/или многократная адаптация ПС к различным условиям применения является одним из ***самых перспективных методов повышения качества и снижения затрат труда специалистов*** в жизненном цикле крупномасштабных комплексов программ.

9.5. Ресурсы на имитацию внешней среды для обеспечения тестирования и испытаний программных средств

При создании крупных ПС одной из больших составляющих могут быть необходимые ***ресурсы на генерацию тестов***. В ряде случаев они соизмеримы с затратами на создание основных функций комплексов программ, что определяется принципиальным соответствием ***сложности необходимых наборов тестов*** и тестового покрытия программ, и ***сложности функций***, реализуемых испытываемым ПС. Создание представительных совокупностей тестов возможно путем использования реальных объектов внешней среды или с помощью программных имитаторов, адек-

ватных этим объектам по результатам функционирования и генерируемой информации. При этом возникает проблема — какой метод и когда выгодней по затратам на генерацию тестов и по обеспечению необходимой степени покрытия тестами испытываемых ПС (см. лекцию 14).

Имитаторы тестов необходимы не только для оценивания достигнутых характеристик качества комплексов программ, но также для их комплексной отладки, квалификационного тестирования, испытаний и при создании версий. Поэтому затраты на программные имитаторы и их экономическую эффективность целесообразно рассматривать в проекте с учетом всего комплекса задач, которые они способны и должны решать в ЖЦ ПС. Анализ эффективности программной имитации внешней среды при разработке и определении качества ПС целесообразно разделить ***на две части***: оценка факторов, определяющих эффективность средств имитации тестов, и оценка экономического выигрыша при моделировании внешней среды на ЭВМ по сравнению с натурными экспериментами в реальных системах.

Факторы, определяющие эффективность программной имитации внешней среды на ЭВМ при разработке ПС, могут оцениваться в основном по их воздействию на качество создаваемых программ. Это влияние трудно непосредственно измерить, однако качественный анализ показывает, что автоматизированная имитация может значительно изменять не только достигаемые характеристики качества разрабатываемого ПС, но также трудоемкость и длительность его создания. Программная имитация внешней среды на ЭВМ может обеспечивать широкие наборы тестов и достаточно полные тестовые покрытия ПС и компонентов при испытаниях, в том числе за пределами характеристик реально существующих или доступных источников тестов, а также соответствующие критическим или опасным ситуациям функционирования объектов внешней среды. Для каждого параметра, отражающего внешнюю среду, отношение диапазона или числа тестов, возможных при программной имитации на ЭВМ по сравнению с натурными экспериментами, может служить оценкой величины, возрастания достоверности определения характеристик качества ПС.

При тестировании необходимо учитывать не только соотношение размеров областей изменения параметров тестов, но и распределение вероятностей значений каждого параметра в этих областях для реальных и перспективных объектов внешней среды. Некоторые значения тестов не только трудно создать при натурных экспериментах, но они являются маловеро-

ятными в реальных условиях. Однако такие, даже маловероятные ситуации и значения тестов могут быть ***критическими и/или особо важными*** для функционирования всей системы, для которой разрабатывается ПС. Выбор и имитация подобных ситуаций позволяют отрабатывать и оценивать качество ПС в критических маловероятных ситуациях, которые невозможно или опасно создавать на реальных объектах, но без их выполнения некоторые ПС недопустимо эксплуатировать в критических системах управления и обработки информации.

Экономическую эффективность программной имитации внешней среды на ЭВМ по сравнению с натурными экспериментами целесообразно оценивать при одинаковых объемах тестовых данных для испытаний и определения качества ПС. Показателем экономической эффективности имитации может служить соотношение затрат ресурсов на проведение натурных экспериментов и затрат на программную имитацию той же совокупности тестовых и эталонных данных.

Затраты ресурсов на натурные эксперименты для генерации тестов при проведении разработки, испытаний и определения качества пропорциональны реальному времени функционирования проверяемого ПС и затратам на применение привлекаемых средств реальной внешней среды. Они включают стоимость эксплуатации реального объекта, создающего тесты в единицу времени (например, затраты на функционирование административной системы, прокатного стана или системы управления воздушным движением и всех управляемых ею объектов). Таким образом, затраты на натурные эксперименты для оценивания характеристик ПС определяются использованием всей реальной внешней среды, в которой предстоит в дальнейшем функционировать программам, а также затратами на средства измерения характеристик этой среды и проверяемого ПС в процессе разработки, испытаний и определения качества.

Затраты на программную имитацию тестовых данных определяются ресурсами, необходимыми на проектирование и эксплуатацию сложных комплексов программ для этих целей, и следующими ***составляющими***:

— затратами на разработку комплекса программ для имитации информации внешних объектов и среды их функционирования;

— затратами на эксплуатацию программ имитации за время проведения тестирования, испытаний и/или определения характеристик качества тестируемого ПС;

— затратами на первичную установку и эксплуатацию моделирующей ЭВМ и вспомогательного оборудования, используемого в имитационном стенде.

Имитационные стенды практически всегда являются уникальными и достаточно полно используют ресурсы моделирующей ЭВМ. В ряде случаев эти комплексы программ могут иметь объем порядка 10^4 — 10^6 строк текста и должны создаваться с применением современных технологических систем. Затраты на эксплуатацию программ имитации в основном определяются длительностью проведения тестирования, испытаний и/или измерения характеристик качества ПС. Значения этого времени соответствуют реальному времени генерации тестовых данных и тестирования программ. Затраты на эксплуатацию ЭВМ, используемую в моделирующем имитационном стенде (МИС), включают: первичные затраты на закупку и установку оборудования, необходимого для имитации тестовых данных, стоимость имитирующей ЭВМ и устройств сопряжения имитационного стенда с ЭВМ, на которой функционируют тестируемые программы.

Обычно МИС используется для тестирования нескольких ПС разного, но близкого целевого назначения. Следовательно, затраты на имитационный стенд и на часть его программ распределяются на число проектов ПС, тестируемых с его использованием. В результате удельные затраты на создание и эксплуатацию стендов быстро убывают при унификации имитаторов и расширении области их применения для тестирования и оценивания качества большого числа ПС, имеющих близкое функциональное назначение. Даже приближенные оценки при системном анализе соотношения этих затрат в большинстве случаев показывают ***высокую рентабельность программных имитаторов внешней среды***, особенно для квалификационного тестирования и оценивания характеристик качества крупномасштабных ПС реального времени. Например, при тестировании ПС для управления воздушным движением применение имитационных стендов, по крайней мере, на порядок снижает затраты по сравнению с натурными экспериментами и использованием реальных объектов (самолетов), а для управления космическими аппаратами или атомными электростанциями это соотношение может быть значительно больше (≈10—100). При создании и определении качества административных систем с полной загрузкой имитация способна заменить сложную организацию функциони-

рования по определенной программе большого коллектива операторов банка, налоговой инспекции или таможенного органа.

При разработке и тестировании компонентов повторяемость некоторых тестов около 3—5, а при испытаниях и определении качества крупномасштабных ПС достигает 2—3. Поэтому целесообразно запоминать имитированные данные для их многократного использования и создавать ***фильмы сценариев поведения и характеристик тестов, отражающих внешнюю среду.*** В результате затраты на имитацию могут быть уменьшены в несколько раз за счет однократной имитации каждой совокупности тестов с полным использованием МИС. Предварительная подготовка фильмов позволяет удобно контролировать имитированные данные, более эффективно использовать МИС, а также обеспечивает абсолютную повторяемость экспериментов. Однако они не позволяют учитывать в имитированных данных реакцию функционирования испытываемых комплексов программ. Поэтому не полностью исключаются сложные эксперименты с одновременным использованием всей реальной аппаратуры МИС и оцениваемой системы, но обеспечивается сокращение в несколько раз количества таких экспериментов.

ЛЕКЦИЯ 10

ДЕФЕКТЫ, ОШИБКИ И РИСКИ В ЖИЗНЕННОМ ЦИКЛЕ ПРОГРАММНЫХ СРЕДСТВ

10.1. Общие особенности дефектов, ошибок и рисков в сложных программных средствах

Статистика ошибок и дефектов в комплексах программ и их характеристики в конкретных типах проектов ПС могут служить ***ориентирами*** для разработчиков при распределении ресурсов в жизненном цикле ПС и предохранять их от излишнего оптимизма при оценке достигнутого качества программных продуктов. Источниками ошибок в ПС являются специалисты — конкретные люди с их индивидуальными особенностями, квалификацией, талантом и опытом. При этом можно выделить ***предсказуемые модификации, расширения и совершенствования ПС*** и изменения, обусловленные выявлением случайных, ***непредсказуемых дефектов и ошибок.*** Вследствие этого плотность потоков и размеры необходимых корректировок в модулях и компонентах при разработке и сопровождении ПС могут различаться в десяток раз. Однако в крупных комплексах программ статистика и распределение типов выполняемых изменений для коллективов разных специалистов нивелируются и проявляются достаточно общие закономерности, которые могут использоваться как ориентиры при их выявлении и систематизации. Этому могут помогать оценки типовых дефектов, модификаций и корректировок путем их накопления и обобщения по опыту создания определенных классов ПС в конкретных предприятиях.

К ***понятию «риски»*** относятся негативные события и их величины, отражающие потери, убытки или ущерб от процессов или продуктов, ***выз-***

ванные дефектами при проектировании требований, недостатками обоснования проектов ПС, а также при последующих этапах разработки, реализации и всего жизненного цикла комплексов программ. В ЖЦ ПС не всегда удается достигнуть требуемого положительного эффекта и может проявляться некоторый ущерб — риск в создаваемых проектах, программных продуктах и их характеристиках. Риски проявляются, как ***негативные последствия дефектов функционирования и применения ПС***, которые способны нанести ущерб системе, внешней среде или пользователю в результате отклонения характеристик объектов или процессов от заданных требованиями заказчика, согласованными с разработчиками.

Оценки качества программных средств могут проводиться с двух позиций: с ***позиции положительной*** эффективности и непосредственной адекватности их характеристик назначению, целям создания и применения, а также с ***негативной позиции*** возможного при этом ущерба — риска от использования ПС или системы. Показатели качества преимущественно отражают положительный эффект от применения системы или ПС и основная задача разработчиков проекта состоит в обеспечении высоких значений качества. Риски характеризуют возможные ***негативные последствия дефектов*** или ущерб пользователей при применении и функционировании ПС и системы, и задача разработчиков сводится к сокращению дефектов и ликвидации рисков. Поэтому методы и системы управления качеством в жизненном цикле ПС близки к методам анализа и управления рисками проектов комплексов программ, они должны их дополнять и совместно способствовать совершенствованию программных продуктов и систем на их основе.

Характеристики дефектов и рисков непосредственно связаны с достигаемой корректностью, безопасностью и надежностью функционирования программ и ***помогают***:

— оценивать реальное состояние проекта и планировать необходимую трудоемкость и длительность для его положительного завершения;

— выбирать методы и средства автоматизации тестирования и отладки программ, адекватные текущему состоянию разработки и сопровождения ПС, наиболее эффективные для устранения определенных видов дефектов и рисков;

— рассчитывать необходимую эффективность контрмер и дополнительных средств оперативной защиты от потенциальных дефектов и невыявленных ошибок;

— оценивать требующиеся ресурсы ЭВМ по расширению памяти и производительности, с учетом затрат на реализацию контрмер при модификации и устранении ошибок и рисков.

Понятие ошибки в программе — в общем случае под ошибкой подразумевается неправильность, погрешность или неумышленное искажение объекта или процесса, что может быть ***причиной ущерба — риска*** при функционировании и применении программы. При этом предполагается, что ***известно правильное, эталонное состояние объекта или процесса***, по отношению к которому может быть определено наличие отклонения — ошибки или дефекта. Исходным эталоном для любого ПС являются спецификация требований заказчика или потенциального пользователя, предъявляемых к программам. Подобные документы устанавливают состав, содержание и значения результатов, которые должен получать пользователь при определенных условиях и исходных данных. Любое отклонение результатов функционирования программы от предъявляемых к ней требований и сформированных по ним эталонов-тестов, следует квалифицировать как ***ошибку — дефект в программе***, наносящий некоторый ущерб. Различия между ожидаемыми и полученными результатами функционирования программ могут быть следствием ошибок не только в созданных программах, но и ошибок в первичных требованиях спецификаций, явившихся базой при создании эталонов-тестов. Тем самым проявляется объективная реальность, заключающаяся в невозможности абсолютной корректности и полноты исходных спецификаций и эталонов для сложных проектов ПС.

На практике в процессе ЖЦ ПС исходные требования поэтапно уточняются, модифицируются, расширяются и детализируются по согласованию между заказчиком и разработчиком. Базой таких уточнений являются ***неформализованные представления и знания*** специалистов-заказчиков и разработчиков, а также результаты промежуточных этапов проектирования. Однако установить некорректность таких эталонов еще труднее, чем обнаружить дефекты в сопровождаемых программах, так как принципиально отсутствуют формализованные данные, которые можно использовать как исходные. В процессе декомпозиции и верификации исходной спецификации требований на ПС возможно появление ошибок в спецификациях на группы программ и на отдельные модули. Это способствует расширению спектра возможных дефектов и вызывает необходимость со-

здания гаммы методов и средств тестирования для выявления некорректностей в спецификациях на компоненты разных уровней.

Важной особенностью процесса выявления ошибок в программах является ***отсутствие полностью определенной программы-эталона***, которой должны соответствовать текст и результаты функционирования разрабатываемой программы. Поэтому установить наличие и локализовать дефект непосредственным сравнением с программой без ошибок в большинстве случаев невозможно. При отладке и тестировании обычно сначала обнаруживаются ***вторичные*** ошибки и ***риски***, т.е. последствия и результаты проявления некоторых внутренних дефектов или некорректностей программ (рис. 10.1). Эти внутренние ***дефекты*** следует квалифицировать как ***первичные*** ошибки или причины обнаруженных аномалий результатов. Последующая локализация и корректировка таких первичных ошибок должна приводить к устранению ошибок, первоначально обнаруживаемых в результатах функционирования программ.

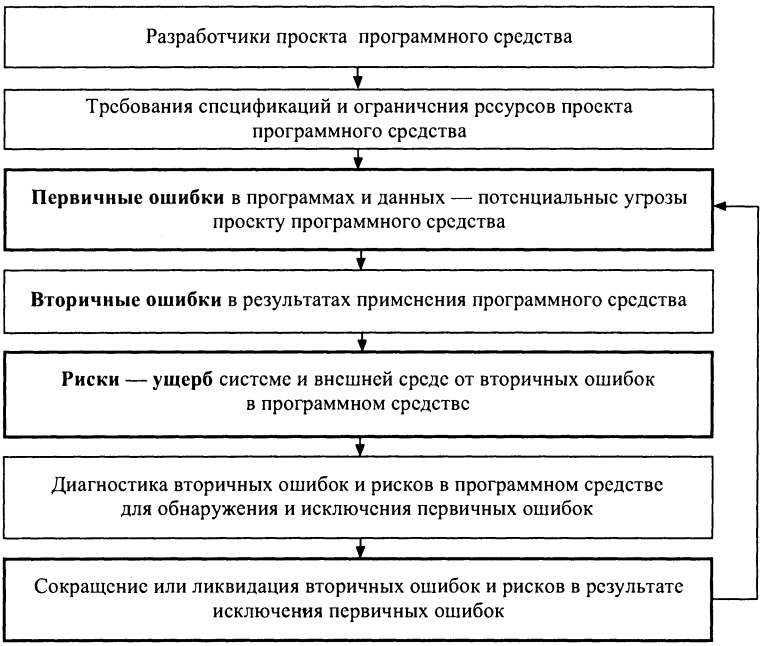

Рис. 10.1

Потери эффективности и риски программ за счет неполной корректности в первом приближении можно считать прямо пропорциональными (с коэффициентом) вторичным ошибкам в выходных результатах. Типичным является случай, когда одинаковые по величине и виду вторичные ошибки в различных результирующих данных существенно различаются по своему воздействию на общую эффективность и риски применения комплекса программ. Это влияние вторичных ошибок, в лучшем случае, можно оценить методами экспертного анализа при условии предварительной, четкой классификации видов возможных первичных ошибок в программах и выходных величин. Таким образом, оценка последствий, отражающихся на вторичных ошибках и функционировании программ, может, в принципе, производиться ***по значениям ущерба — риска вследствие неустраненных их причин — первичных ошибок в программе***. Вторичные ошибки являются определяющими для эффективности функционирования программ, однако не каждая первичная ошибка вносит заметный вклад в выходные результаты. Вследствие этого ряд первичных ошибок может оставаться необнаруженным и, по существу, не влияет на функциональные характеристики ПС.

Появление ошибок в программах, естественно, предшествует их обнаружению и устранению на основе вторичных проявлений. Наибольшее число первичных ошибок вносится на этапах системного анализа и разработки модификаций программ. При этом на долю системного анализа приходятся наиболее сложные для обнаружения и устранения дефекты. На последующих этапах разработки изменений ПС ошибки вносятся и устраняются в программах в процессе их корректировки по результатам тестирования. Общие тенденции состоят в быстром росте затрат на выполнение каждого изменения на последовательных этапах процессов модификации программ.

При системном анализе модификаций интенсивность обнаружения ошибок относительно невелика, и ее трудно выделить из процесса проектирования ПС. Интенсивность проявления и обнаружения вторичных ошибок наиболее велика на этапе активного тестирования и автономной отладки программных компонентов. Затем она снижается приблизительно экспоненциально. Различия интенсивностей ***устранения первичных ошибок, на основе их вторичных проявлений***, и внесения первичных ошибок

при корректировках программ определяют скорость достижения заданного качества версий ПС. Уровень серьезности последствий ошибок варьирует от классов проектов и от предприятия, но, в общем, можно разделить ошибки на три уровня.

Небольшими ошибками называют такие, на которые средний пользователь не обратит внимания при применении ПС вследствие отсутствия их проявления и последствия которых обычно так и не обнаруживаются. Небольшие ошибки могут включать орфографические ошибки на экране, пропущенные разделы в справочнике и другие мелкие проблемы. Такие ошибки никогда не помешают выпуску и применению версии системы и программного продукта. По десятибалльной шкале рисков небольшие ошибки находятся в пределах от 1 до 3-го приоритета (см. ниже).

Умеренными ошибками называют те, которые влияют на конечного пользователя, но имеются слабые последствия или обходные пути, позволяющие сохранить достаточную функциональность ПС. Это такие дефекты, как неверные ссылки на страницах, ошибочный текст на экране и даже сбои, если эти сбои трудно воспроизвести и они не оказывают влияния на существенное число пользователей. Некоторые умеренные ошибки, возможно, проникают в конечный программный продукт. Ошибки, которые можно исправить на этом уровне, следует исправлять, если на это есть время и возможность. По десятибалльной шкале умеренные ошибки находятся в диапазоне от 4 до 7-го приоритета.

Критические ошибки останавливают выпуск версии программного продукта. Это могут быть ***ошибки с высоким влиянием***, которые вызывают сбой в системе или потерю данных, отражаются на надежности и безопасности применения ПС, с которыми никогда не передается комплекс программ пользователю. По десятибалльной шкале — от 8 до 10-го приоритета.

Совокупность ошибок, дефектов и последствий модификаций проектов крупномасштабных комплексов программ можно упорядочить и условно представить в виде перевернутой пирамиды в зависимости от потенциальной опасности и возможной величины корректировок их последствий — рис. 10.2. В верхней части перечня расположены модификации, дефекты и ошибки, последствия которых обычно требуют наибольших затрат ресурсов для реализации изменений, и они постепенно сокращаются при снижении по перечню. Такое представление величины типов кор-

ректировок программ и данных полезно использовать ***как ориентир*** для учета необходимых ресурсов при разработке и сопровождении ПС, однако оно может содержать значительные отклонения при упорядочении статистических данных реальных проектов. Каждому типу корректировок соответствует более или менее определенная категория специалистов, являющихся источником изменений данного типа (таблица 10.1). Такую корреляцию целесообразно рассматривать и учитывать как общую качественную тенденцию при анализе и поиске их причин.

Рис. 10.2

Таблица 10.1

Специалисты — источники дефектов и ошибок	Типы первичных дефектов и ошибок программного средства и документации
Заказчики проекта	Дефекты организации проекта и исходных требований заказчика
Менеджер проекта	Дефекты, обусловленные реальной сложностью проекта
Менеджер-архитектор комплекса программ	Ошибки планирования и системного проектирования программного средства
Проблемно-ориентированные аналитики и системные архитекторы	Системные и алгоритмические дефекты и ошибки проекта
Спецификаторы компонентов проекта	Алгоритмические ошибки компонентов и документов программного средства
Разработчики программных компонентов — программисты	Программные дефекты и ошибки компонентов и документов программного средства
Системные интеграторы	Системные ошибки и дефекты реализации версий программного средства и документации
Тестировщики	Программные и алгоритмические ошибки программного средства и документации
Управляющие сопровождением и конфигурацией, инструкторы интерфейсов	Ошибки проектирования и реализации версий программного продукта
Документаторы	Дефекты и ошибки обобщающих документов

10.2. Причины и свойства дефектов, ошибок и модификаций в сложных программных средствах

Одной из основных причин изменений комплексов программ являются ***организационные дефекты при модификации и расширении функций ПС,*** которые отличаются от остальных типов и условно могут быть выделены как самостоятельные (см. рис. 10.2). Ошибки и дефекты данного типа появляются из-за недостаточного понимания коллективом специалистов технологии процесса ЖЦ ПС, а также вследствие отсутствия четкой его организации и поэтапного контроля качества продуктов и изменений. Это порождается пренебрежением руководителей к организации всего

технологического процесса ЖЦ сложных ПС и приводит к серьезной недооценке его дефектов, а также трудоемкости и сложности модификаций. При отсутствии планомерной и методичной разработки и тестирования изменений ПС остается невыявленным значительное количество ошибок, и, прежде всего, дефекты взаимодействия отдельных функциональных компонентов между собой и с внешней средой. Для сокращения этого типа массовых ошибок активную роль должны играть лидеры — менеджеры и системотехники, способные вести контроль и конфигурационное управление требованиями, изменениями и развитием версий и компонентов ПС.

Изменения характеристик системы и внешней среды, принятые в процессе разработки ПС за исходные, могут быть результатом аналитических расчетов, моделирования или исследования аналогичных систем. В ряде случаев может отсутствовать полная адекватность предполагаемых и реальных характеристик, что является ***причиной сложных и трудно обнаруживаемых системных ошибок и дефектов развития проекта***. Ситуация с такими ошибками дополнительно усложняется тем, что эксперименты по проверке взаимодействия ПС с реальной внешней средой во всей области изменения параметров зачастую сложны и дороги, а в отдельных случаях, при создании опасных ситуаций, недопустимы. В этих случаях приходится использовать моделирование и имитацию внешней среды с заведомым упрощением ее отдельных элементов и характеристик, хотя степень упрощения не всегда удается оценить с необходимой точностью. Однако полной адекватности моделей внешней среды и реальной системы добиться трудно, а во многих случаях и невозможно, что может являться причиной значительного числа крупных дефектов.

Первичные ошибки в программах проектов можно анализировать с разной степенью детализации и в зависимости от различных факторов. Практический опыт показал, что ***наиболее существенными факторами, влияющими на характеристики обнаруживаемых ошибок, являются***:

— методология, технология и уровень автоматизации системного и структурного проектирования ПС, а также непосредственного программирования компонентов;

— длительность с начала процесса тестирования и текущий этап разработки или сопровождения и модификации комплекса программ;

— класс ПС, масштаб (размер) и типы компонентов, в которых обнаруживаются ошибки;

— методы, виды и уровень автоматизации верификации и тестирования, их адекватность характеристикам компонентов и потенциально возможным в программах ошибкам;

— виды и достоверность эталонов-тестов, которые используются для обнаружения ошибок.

Первичные ошибки в ПС в порядке уменьшения их влияния на сложность обнаружения и масштабы корректировок можно разделить на следующие группы (см. рис. 10.2):

— ошибки, обусловленные сложностью компонентов и ПС в целом и наиболее сильно влияющие на размеры модификаций;

— ошибки вследствие большого масштаба — размера комплекса программ, а также высоких требований к его качеству;

— ошибки планирования и корректности требований модификаций часто могут быть наиболее критичным для общего успеха ЖЦ ПС и системы;

— ошибки проектирования, разработки структуры и функций ПС в более полные и точные технические описания сценариев того, как комплекс программ и система будут функционировать;

— системные ошибки, обусловленные отклонением функционирования ПС в реальной системе, и характеристик внешних объектов от предполагавшихся при проектировании;

— алгоритмические ошибки, связанные с неполным формированием необходимых условий решения и некорректной постановкой целей функциональных задач;

— ошибки реализации спецификаций изменений — программные дефекты, возможно, ошибки нарушения требований или структуры компонентов ПС;

— программные ошибки, вследствие неправильной записи текстов программ на языке программирования и ошибок трансляции текстов изменений программ в объектный код;

— ошибки в документации, которые наиболее легко обнаруживаются и в наименьшей степени влияют на функционирование и применение версий ПС;

— технологические ошибки подготовки физических носителей и документации, а также ввода программ в память ЭВМ и вывода результатов на средства отображения.

Сложность проявления, обнаружения и устранения ошибок значительно конкретизируется и становится измеримой, когда устанавливается связь этого понятия с конкретными ресурсами, необходимыми для решения соответствующей задачи, и возможными проявлениями дефектов. При разработке и сопровождении программ основным лимитирующим ресурсом обычно являются допустимые трудозатраты специалистов, а также ограничения на сроки разработки, параметры ЭВМ, технологию проектирования корректировок ПС. Показатели сложности при анализе можно разделить на ***две большие группы***:

— ***сложность ошибок при создании корректировок*** компонентов и комплекса программ — статическая сложность, когда реализуются его требуемые функции, вносятся основные дефекты и ошибки;

— ***сложность проявления ошибок функционирования*** программ и получения результатов — динамическая сложность, когда проявляются дефекты и ошибки, отражающиеся на функциональном назначении, рисках и качестве применения версии ПС.

К группе факторов, влияющих на ***сложность ошибок*** комплексов программ, относятся:

— величина — размер модифицируемой программы, выраженная числом строк текста, функциональных точек или количеством программных модулей в комплексе;

— количество обрабатываемых переменных или размер и структура памяти, используемой для размещения базы данных корректировок;

— трудоемкость разработки изменений комплекса программ;

— длительность разработки и реализации корректировок;

— число специалистов, участвующих в ЖЦ комплекса программ.

Некоторые из перечисленных параметров коррелированы между собой, причем определяющими часто являются размер изменений программы и объем базы данных. Остальные характеристики можно рассматривать как вторичные, однако они могут представлять самостоятельный интерес при анализе сложности и прогнозировании вероятного числа дефектов в измененной программе. ***Сложность ошибок*** комплексов программ целесообразно анализировать на базе трех наиболее специфических компонентов:

— ***сложность ошибок изменяемых программных компонентов и модулей*** определяется конструктивной сложностью модификации оформ-

ленного компонента программы и может быть оценена с позиции сложности внутренней структуры и преобразования данных в каждом модуле, а также интегрально по некоторым внешним статистическим характеристикам размеров модулей;

— ***сложность ошибок корректировок структуры комплекса*** или компонентов и связей между модулями по передачам управления и по обмену информацией определяется глубиной взаимодействия модулей и регулярностью структуры межмодульных связей;

— ***сложность ошибок изменения структуры данных*** определяется количеством и структурой глобальных и обменных переменных в базе данных, регулярностью их размещения в массивах, а также сложностью доступа к этим данным.

Масштаб — размер комплексов программ и их изменяемой части наиболее сильно влияет на количество ошибок, а также на требования к ***качеству*** ПС (см. лекцию 5). Качество откорректированного ПС характеризуется многими показателями, состав которых зависит от класса и конкретного назначения комплекса программ. Ниже предполагается, что всегда модификации ПС соответствуют заданному функциональному назначению и основным требованиям заказчика к их качеству. По мере увеличения размера и повышения требований к качеству ПС и его корректировкам затраты на обнаружение и устранение ошибок ПС увеличиваются все более высокими темпами. Одновременно расширяется диапазон неопределенности достигаемого качества. В зоне высокого качества программ возрастают трудности измерения этих характеристик, что может приводить к необходимости изменения затрат в несколько раз в зависимости от применяемых методов и результатов оценки качества ПС. Вследствие этого в ЖЦ сложных и сверхсложных ПС всегда велики проявления неустраненных ошибок и недостаточна достоверность оценок достигнутого качества.

Ошибки корректности формирования и планирования выполнения требований к ПС часто считаются наиболее критичными для общего успеха версий программного продукта и системы. Ошибки требований являются наиболее трудными для обнаружения и наиболее сложными для исправления. Вот почему исправление ошибок требований может быть в 15—70 раз дороже, чем ошибок их программирования. Требование к изме-

нению может быть пропущено в спецификации к системе и ПС. Это ведет к неудовлетворенности пользователя, и программа считается заказчиком и пользователем ошибочной. Пропуск некоторых требований — это наиболее обычная проблема среди ошибок требований. Ошибка требований может представлять собой конфликтующие требования в спецификации модификаций. Например, два требования, которым необходимо следовать, имеют противоположный смысл. Может проявляться неопределенность требований — такой способ формулирования требования, что даже если и не конфликтует с другим требованием, оно выражено недостаточно ясно, чтобы привести к единственному, конструктивному решению при разработке изменения. Конечный пользователь часто называет это ошибкой, хотя на самом деле это выбор конструктивного решения на основе неполного или неопределенного требования. Многочисленные исследования показали, что ошибки требований дороже всего исправить и труднее всего обнаружить.

Ошибки проектирования и разработки структуры ПС определяются процессами перевода неопределенных и общих положений, сделанных на стадии спецификаций требований, в более точные технические описания сценариев того, как измененные ПС и система должны работать. Ошибки структуры легче обнаружить, чем ошибки требований, но они в конечном итоге могут оказаться при корректировках такими же дорогостоящими. Главная причина того, что ошибки структуры дорого исправлять, состоит в том, что они могут влиять на систему в целом. Исправление изменений всей системы сложнее, и при этом возникает большая опасность занести новые ошибки, чем при исправлении нескольких нарушенных строк кода или при замене одного модуля.

Ошибки структуры можно разделить на три категории: пропуски, конфликты и ошибки перевода. Пропуски означают неспособность включить изменения одного или более требований в окончательную структуру ПС. Когда пропуск новой функции или компонента попадает в окончательную структуру, он станет ошибкой в конечном программном продукте. Конфликты возникают, когда модификация двух различных, конструктивных свойств имеют конфликтующую структуру. Это может происходить в случае явного конфликта, когда в структуре установлено, что файл может быть открыт двумя разными людьми в одно и то же время, тогда

как в базовом классе определяется только однопользовательский доступ. Ошибки, которые основаны на конфликтах на этом уровне, часто невозможно исправить без полного переписывания модулей версии ПС.

Ошибки перевода — наиболее коварные среди всех ошибок структурного уровня. Они проявляются, когда требования заказчика интерпретируются неправильно, по крайней мере, с точки зрения конечного пользователя. Если разработчик структуры либо неверно прочитает требования, либо не увидит содержание требования, так же как конечный пользователь, появится ошибка разработки структуры данного компонента или ПС.

Системные ошибки в ПС определяются, прежде всего, неполной информацией о реальных процессах, происходящих в источниках и потребителях информации. Кроме того, эти процессы зачастую зависят от самих алгоритмов и поэтому не могут быть достаточно определены и описаны заранее без исследования изменений функционирования ПС во взаимодействии с внешней средой. На начальных этапах не всегда удается точно и полно сформулировать целевую задачу всей системы, а также целевые задачи основных групп программ, и эти задачи уточняются в процессе проектирования. В соответствии с этим уточняются и конкретизируются спецификации на отдельные компоненты и выявляются отклонения от уточненного задания, которые могут квалифицироваться как системные ошибки.

Характеристики внешних объектов, принятые в качестве исходных данных в процессе разработки алгоритмов, могут являться результатом аналитических расчетов, моделирования или исследования аналогичных систем. Во всех случаях может отсутствовать полная адекватность условий получения предполагаемых и реальных характеристик внешней среды, что является причиной сложных и трудно обнаруживаемых ошибок. Это усугубляется тем, что очень часто невозможно заранее предусмотреть все разнообразие возможных внешних условий и реальных сценариев функционирования и применения версий программного продукта.

При автономной и в начале комплексной отладки версий ПС относительная доля системных ошибок может быть невелика (около 10%), но она существенно возрастает (до 35—40%) на завершающих этапах комплексной отладки новых базовых версий ПС. В процессе сопровождения системные ошибки являются преобладающими (около 60—80% от всех оши-

бок). Следует также отметить большое количество команд, корректируемых при исправлении каждой такой ошибки (около 20—50 команд на одну ошибку).

Алгоритмические ошибки программ трудно поддаются обнаружению методами статического автоматического контроля. Трудность их обнаружения и локализация определяется, прежде всего, отсутствием для многих логических программ строго формализованной постановки задачи, полной и точной спецификации, которую можно использовать в качестве эталона для сравнения результатов функционирования программ. К алгоритмическим ошибкам следует отнести, прежде всего, ошибки, обусловленные некорректной постановкой требований к функциональным задачам, когда в спецификациях не полностью оговорены все условия, необходимые для получения правильного результата. Эти условия формируются и уточняются в значительной части в процессе тестирования и выявления ошибок в результатах функционирования программ. Ошибки, обусловленные неполным учетом всех условий решения задач, являются наиболее частыми в этой группе и составляют до 50—70% всех алгоритмических ошибок.

К алгоритмическим ошибкам следует отнести также ошибки интерфейса модулей и функциональных групп программ, когда информация, необходимая для функционирования некоторой части программы, оказывается не полностью подготовленной программами, предшествующими по времени включения, или неправильно передаются информация и управление между взаимодействующими модулями. Этот вид ошибок составляет около 10% от общего количества, и их можно квалифицировать как ошибки некорректной постановки задач. Алгоритмические ошибки проявляются в неполном учете диапазонов изменения переменных, в неправильной оценке точности используемых и получаемых величин, в неправильном учете корреляции между различными переменными, в неадекватном представлении формализованных условий решения задачи в виде частных спецификаций или блок-схем, подлежащих программированию. Эти обстоятельства являются причиной того, что для исправления каждой алгоритмической ошибки приходится изменять в среднем около 20 команд (строк текста), т.е. существенно больше, чем при программных ошибках.

Особую, весьма существенную, часть алгоритмических ошибок в системах реального времени, при сопровождении составляют просчеты в

использовании доступных ресурсов вычислительной системы. Получающиеся при модификации программ попытки превышения использования выделенных ресурсов следует квалифицировать как ошибку, так как затем всегда следует корректировка с целью удовлетворения имеющимся ограничениям. Одновременная разработка множества модулей различными специалистами затрудняет оптимальное и сбалансированное распределение ограниченных ресурсов ЭВМ по всем задачам, так как отсутствуют достоверные данные потребных ресурсов для решения каждой из них. В результате возникает либо недостаточное использование, либо, в подавляющем большинстве случаев, нехватка каких-то ресурсов ЭВМ для решения задач в первоначальном варианте. Наиболее крупные просчеты обычно допускаются при оценке времени реализации различных групп программ реального времени и при распределении производительности ЭВМ. Алгоритмические ошибки этого типа обусловлены технической сложностью расчета времени реализации программ и сравнительно невысокой достоверностью определения вероятности различных маршрутов обработки информации.

Ошибки реализации спецификаций компонентов — это программные дефекты, возможно, ошибки требований, структуры или программные ошибки компонентов. Ошибки реализации наиболее обычны и, в общем, наиболее легки для исправления в системе, что не делает проблему легче для программистов (см. таблицу 10.1). В отличие от ошибок требований и структурных ошибок, которые обычно специфичны для приложения, программисты часто совершают при кодировании одни и те же виды ошибок.

Первую категорию составляют дефекты, которые приводят к отображению для пользователя сообщений об ошибках при точном следовании порядку выполнения требуемых функций. Хотя эти сообщения могут быть вполне законны, пользователи могут посчитать это ошибкой, поскольку они делали все правильно и, тем не менее, получили сообщение об ошибке. Часто ошибки этого типа вызваны либо проблемами с ресурсами, либо специфическими зависимостями от данных.

Вторая категория модификаций может содержать ошибки, связанные с дефектами в графическом интерфейсе пользователя. Такие ошибки могут являться либо нестандартными модификациями пользовательского интерфейса, которые приводят к тому, что пользователь совершает неверные

действия, либо они могут быть стандартными компонентами пользовательского интерфейса, используемыми иначе, чем ожидает конечный пользователь.

Третья категория может содержать пропущенные на стадии реализации функции, что всегда считается ошибкой, возможно, с большим риском. Многие тестировщики и пользователи бета-версий сообщают об ошибках, которые на самом деле являются желательными улучшениями. В данном случае можно не замечать обнаруженные таким образом отсутствия функций, которых не было в спецификациях.

Программные ошибки модифицированных компонентов по количеству и типам в первую очередь определяются степенью автоматизации программирования и глубиной статического контроля текстов программ. Количество программных ошибок зависит от квалификаций программистов, от общего размера комплекса программ, от глубины информационного взаимодействия модулей и от ряда других факторов. При разработке ПС программные ошибки можно классифицировать по видам используемых операций на следующие крупные группы: ошибки типов операций; ошибки переменных; ошибки управления и циклов. В логических компонентах ПС эти виды ошибок близки по удельному весу, однако для автоматизации их обнаружения применяются различные методы. На начальных этапах разработки и автономной отладки модулей программные ошибки составляют около одной трети всех ошибок. Каждая программная ошибка влечет за собой необходимость изменения около 10 команд, что существенно меньше, чем при алгоритмических и системных ошибках.

Ошибки в документации модификаций состоят в том, что система делает что-то одним образом, а документация отражает сценарий, что она должна работать иначе. Во многих случаях права должна быть документация, поскольку она написана на основе оригинальной спецификации требований системы. Иногда документация пишется и включает допущения и комментарии о том, как, по мнению авторов документации, система должна работать. В других случаях ошибку можно проследить не до кода, а до документации конечных пользователей, внутренних технологических документов, характеризующих систему, и даже до экранных подсказок и файлов помощи. Ошибки документации можно разделить на три категории — ***неясность, неполнота и неточность***. Неясность — это когда

пользователю не дается достаточно информации, чтобы определить, как сделать процедуру должным образом. Неполная документация оставляет пользователя без информации о том, как правильно реализовать и завершить задачу. Пользователь считает, что задача выполнена, хотя на самом деле это не так. Такие ошибки ведут к тому, что пользователь не удовлетворен версией ПС, даже если программа в действительности может сделать все, что хочет пользователь. Неточная документация — это худший вид ошибок документации. Такие ошибки часто возникают, когда при сопровождении в систему позже вносятся изменения и об этих изменениях не сообщают лицу, пишущему документацию.

Технологические ошибки документации и фиксирования программ в памяти ЭВМ составляют иногда до 10% от общего числа ошибок, обнаруживаемых при тестировании. Большинство технологических ошибок выявляется автоматически статическими методами. При ручной подготовке текстов машинных носителей при однократном фиксировании исходные данные имеют вероятность искажения около 10^{-3} — 10^{-4} на символ. Дублированной подготовкой и логическим контролем вероятность технологической ошибки может быть снижена до уровня 10^{-5} — 10^{-7} на символ. Непосредственное участие человека в подготовке данных для ввода в ЭВМ и при анализе результатов функционирования программ по данным на дисплеях определяет в значительной степени их уровень достоверности и не позволяет полностью пренебрегать этим типом ошибок в программах.

В ***примере анализа ошибок конкретного крупного проекта*** было принято, что завершилась инспекция начального запрограммированного кода крупного ПС на предмет его соответствия рабочей проектной спецификации, в ходе которой было обнаружено 3,48 ошибки на тысячу строк кода. Наибольшее совпадение аппроксимации рэлеевской кривой распределения ошибок с фактическими данными установлено для момента получения этих данных, ему соответствует значение, равное также 3,48. Значения числа ошибок на тысячу строк получены при пересчетах на более ранние этапы соответственно эскизного — (3,3) и рабочего — (7,8) проектирования программ. При прогнозировании в соответствии с рэлеевской кривой распределения вероятности проявления дефектов программ на следующем этапе квалификационного тестирования компонентов следовало ожидать обнаружения около 2,12 ошибки на тысячу строк исходного кода.

В случае сохранения той же закономерности в момент поставки клиенту на испытания программный продукт мог содержать менее 0,07 ошибки на тысячу строк кода. Отмечается также, что частость проявления 0,1—0,05 ошибки на тысячу строк кода можно считать допустимой для ответственных систем реального времени.

В исследованиях 20 крупных поставляемых программных продуктов, созданных в 13 различных организациях, коллективы специалистов добились среднего уровня 0,06 дефекта на тысячу строк нового и измененного программного кода. При использовании структурного метода в пяти проектах достигнуто 0,04—0,075 ошибки на тысячу строк. Таким образом, ***уровень ошибок около 0,05 на тысячу строк кода*** в разных публикациях считается близким к предельному для высококачественных программных продуктов.

Другим примером оценок уровня ошибок критического ПС особенно высокого качества может служить программный продукт бортовых систем «Шаттла», созданный NASA. По оценке авторов, в нем содержится менее одной ошибки на 10 000 строк кода. Однако стоимость программного продукта достигает 1000 $ за строку кода, что в среднем в сто раз больше, чем для административных систем, и в десять раз больше, чем для ряда ординарных критических управляющих систем реального времени.

Приведенные характеристики типов дефектов и количественные данные могут служить ***ориентирами при прогнозировании возможного наличия невыявленных ошибок*** в ЖЦ различных сложных ПС высокого качества. Следующим логическим шагом процесса их оценивания может быть усреднение для большого числа проектов фактических данных о количестве ошибок на конкретном предприятии, приходящихся на тысячу строк кода, которые обнаружены в различных ПС. Тогда в следующем проекте будет иметься возможность использования этих данных, в качестве меры количества ошибок, обнаружение которых следует ожидать при выполнении проекта с таким же уровнем качества ПС, или с целью повышения производительности при разработке для оценки момента прекращения дальнейшего тестирования. Подобные оценки гарантируют ***от избыточного оптимизма*** при определении сроков и при разработке графиков разработки, сопровождения и реализации модификаций программ с заданным качеством. Непредсказуемость конкретных ошибок в програм-

мах приводит к целесообразности последовательного, методичного фиксирования и анализа возможности проявления любого типа дефектов и необходимости их исключения на наиболее ранних этапах ЖЦ ПС при минимальных затратах.

10.3. Риски в жизненном цикле сложных программных средств

Причинами возникновения и проявления рисков могут быть: ***злоумышленные, активные воздействия заинтересованных лиц*** или ***случайные негативные проявления дефектов*** внешней среды, системы или пользователей. В первом случае риски могут быть обусловлены искажениями программ и информационных ресурсов и их уязвимостью от предумышленных, внешних воздействий (атак) с целью незаконного использования или искажения информации и программ, которые по своему содержанию предназначены для применения ограниченным кругом лиц. Для решения этой проблемы созданы и активно развиваются методы, средства и стандарты обеспечения защиты программ и данных от ***предумышленных негативных внешних воздействий***. Специфические факторы обеспечения информационной безопасности и риски, характерные для сложных информационных систем, — целостность, доступность и конфиденциальность информационных ресурсов, а также ряд типовых процедур систем защиты — криптографическая поддержка, идентификация и аутентификация, защита и сохранность данных пользователей ***при предумышленных атаках из внешней среды далее не рассматриваются.***

Риски при случайных, дестабилизирующих воздействиях дефектов программных средств и отсутствии предумышленного негативного влияния на системы, ПС или информацию баз данных существенно отличаются от предшествующих задач. Эти риски объектов и систем зависят от отказовых ситуаций, отрицательно отражающихся на работоспособности и реализации их основных функций, причинами которых могут быть дефекты и аномалии в аппаратуре, программах, данных или вычислительных процессах. При этом катастрофически, критически или существенно искажается процесс функционирования систем, что может наносить значительный ущерб при их применении. Основными источниками отказо-

вых ситуаций могут быть некорректные исходные требования, сбои и отказы в аппаратуре, дефекты или ошибки в программах и данных функциональных задач, проявляющиеся при их исполнении в соответствии с назначением. При таких воздействиях внешняя, функциональная работоспособность систем может разрушаться не полностью, однако невозможно полноценное выполнение заданных функций и требований к качеству информации для потребителей. Вредные и катастрофические последствия таких отказов в ряде областей применения систем могут превышать по результатам последствия злоумышленных воздействий, имеют свою природу, особенности и характеристики.

Рассматриваемые риски могут быть обусловлены нарушениями технологий или ограничениями при использовании ресурсов — бюджета, планов, коллектива специалистов, инструментальных средств, выделенных на разработку ПС. Результирующий ***ущерб*** в совокупности зависит от величины и вероятности проявления каждого негативного воздействия. Этот ущерб — риск характеризуется разнообразными метриками, зависящими от объектов анализа, и в некоторых случаях может измеряться прямыми материальными, информационными, функциональными потерями применяемых ПС или систем. Одним из косвенных методов определения величины риска может быть ***оценка совокупных затрат***, необходимых для ликвидации негативных последствий в ПС, системе или внешней среде, проявившихся в результате конкретного рискового события.

Процессы анализа и сокращения рисков должны сопутствовать основным этапам разработки и обеспечения ЖЦ сложных программных средств в соответствии с международными стандартами, а также методам систем обеспечения качества ПС. Эти процессы могут быть отражены ***пятью этапами работ и процедур***, которые рекомендуется выполнять при поддержке базовых работ жизненного цикла проектов сложных программных средств, и могут служить основой для разработки соответствующих планов работ при управлении и сокращении рисков — рис. 10.3:

— анализ рисков следует начинать с подготовки детальных исходных требований и характеристик проекта ПС, системы и внешней среды, для которых должны отсутствовать риски функционирования и применения;

— для управления рисками и их сокращения в рассматриваемых проектах сложных комплексов программ рекомендуется выделять три класса

рисков: функциональной пригодности ПС, конструктивных характеристик качества и нарушения ограничений ресурсов при реализации процессов ЖЦ ПС;

Подготовка исходных данных:
— определение целей, назначения и функций проекта программного средства и системы;
— разработка предварительных требований к функциональной пригодности и конструктивным характеристикам ПС;
— формирование группы экспертов для анализа угроз и управления рисками

Выделение, идентификация, анализ угроз и рисков программного средства:
— выделение источников и угроз нарушения требований и ограничений ресурсов, определение критериев работоспособности проекта ПС;
— анализ причин, выделение категорий угроз и возможных последствий рисков функциональной пригодности ПС;
— анализ причин, угроз и возможных рисков конструктивных характеристик и ограничения ресурсов проекта ПС

Оценивание опасности угроз и рисков и выбор контрмер для их сокращения:
— оценивание возможных последствий, уровней потенциальных угроз и приоритетов категории рисков проекта ПС;
— планирование методов и ресурсов реализации контрмер для сокращения опасных, приоритетных рисков проекта ПС;
— распределение ресурсов на контрмеры для сбалансированного сокращения интегрального риска проекта ПС

Сокращение или ликвидация опасных рисков ПС:
— реализация контрмер для сокращения интегрального риска и составление отчетов о состоянии проекта;
— корректировка требований к функциональной пригодности, конструктивным характеристикам и ресурсов проекта ПС;
— регистрация результатов сокращения интегрального риска на очередном этапе проекта ПС

Контроль, регистрация, мониторинг и утверждение допустимого интегрального риска ПС:
— контроль, отслеживание и мониторинг реализации сокращения интегрального риска по этапам проекта ПС;
— документирование и утверждение допустимого интегрального риска по этапам жизненного цикла проекта ПС;
— оформление итогового отчета по результатам сокращения рисков ПС

Рис. 10.3

— в каждом классе предлагается анализировать несколько категорий наиболее важных рисков, которые упорядочивать по степени опасности, угроз для проекта, обусловленных ограничениями ресурсов, дефектами и/или недостаточным качеством разработки и жизненного цикла ПС;

— контрмеры для сокращения рисков рекомендуется анализировать и применять последовательно, начиная с ликвидации наиболее опасных исходных причин — угроз, затем проводить анализ и уменьшение уязвимости компонентов и ПС в целом, а при недостаточности этих контрмер воздействовать непосредственно на уменьшение итогового ущерба — риска в жизненном цикле ПС и системы;

— процессы устранения рисков должны завершаться процедурами мониторинга, сопровождения и конфигурационного управления изменениями версий комплексов программ высокого качества с минимальными допустимыми рисками.

Таким образом, в жизненном цикле программных средств ущерб — риски могут проявляться — рис.10.4:

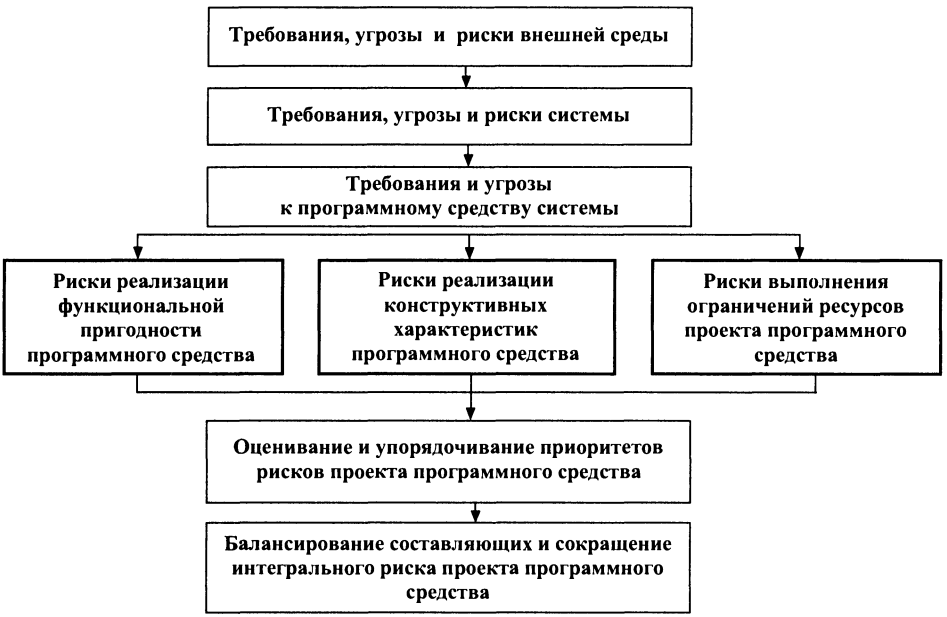

Рис. 10.4

— в искажениях или неполной реализации требуемого назначения, функций или взаимодействия ПС с компонентами системы или внешней среды — недостатками и дефектами ***функциональной пригодности комплексов программ***;

— в недостаточных и не соответствующих требованиям ***конструктивных характеристиках*** качества ПС при его функционировании и применении по прямому назначению;

— в нарушениях ограничений на использование ***экономических, временных или технических ресурсов*** при создании и применении ПС.

Анализ и оценка рисков должны начинаться с исследования понятий, требований и функций, ***способствующих одобрению и применению*** заказчиком и пользователями конкретного программного продукта. При этом должны быть определены требования к характеристикам ПС и оценки возможного ущерба при их нарушении. Исследования процессов разработки проектов ПС показали, что во многих случаях стоимость и длительность их реализации значительно превышали предполагаемые, а характеристики качества не соответствовали требуемым, что наносило ущерб заказчикам, пользователям и разработчикам. Эти потери — ущерб проектов, могли бы быть значительно сокращены своевременным анализом, прогнозированием и корректированием рисков возможного нарушения требований контрактов, технических заданий и спецификаций на характеристики, выделяемые ресурсы и технологию создания комплексов программ.

Управления рисками предполагает ясное понимание внутренних и внешних причин и реальных источников угроз, влияющих на качество проекта ПС, которые могут привести к его провалу или большому ущербу. В результате анализа следует создавать план ***отслеживания изменения рисков в жизненном цикле ПС***, который должен регулярно рассматриваться и корректироваться. Главной целью управления рисками является обнаружение, идентификация и контроль за редко встречающимися ситуациями и факторами, которые приводят к негативным — рисковым результатам проекта. Это должно отражаться на применении регламентированных процессов, в которых факторы и угрозы рисков систематически идентифицируются, оцениваются и корректируются.

Одной из самых распространенных причин и опасных источников рисков являются ***ошибки при оценке масштаба — размера*** проекта или

программного продукта (см. лекцию 5). Эти ошибки чаще всего бывают случайными — непредумышленными, вследствие недостаточной компетентности заказчика или разработчика-поставщика. Однако в некоторых случаях в превышении значения согласованного масштаба заказанного продукта могут быть заинтересованы разработчики для получения больших ресурсов от заказчика, а уменьшенную оценку масштаба могут стремиться представить заказчики для сокращения выделяемых затрат на разработку проекта или программного продукта. Величина оцененного и согласованного между заказчиком и разработчиком масштаба ПС непосредственно отражается на:

— ***бюджете и трудоемкости*** разработки и обеспечения всего жизненного цикла программного продукта;

— ***затратах времени и сроках*** создания и всего жизненного цикла реализации и применения ПС;

— потребности в ***численности и квалификации*** специалистов-разработчиков для реализации проекта и создания программного продукта в соответствии с требованиями заказчика.

Эти три ключевые характеристики обычно тесно связаны и могут изменяться в процессах разработки проекта заказчиком или разработчиком в сторону увеличения или уменьшения в соответствии с их интересами, целями и возможностями. При стремлении заказчика сократить сроки реализации проекта может требоваться увеличить бюджет (трудоемкость) и численность специалистов. При объективно недостаточном числе и квалификации специалистов, естественно, возрастают сроки создания программного продукта и, возможно, бюджет проекта. Практически всегда основным фактором, определяющим значения и взаимосвязь этих важнейших характеристик и их рисков при создании сложных ПС является оценка и отслеживание масштаба проекта, а также его согласование между заказчиком и разработчиком.

Для снижения возможного ущерба — рисков применяются ***оценки, контроль и мониторинг рисков, а также различные контрмеры***. Уменьшение рисков должно производиться путем максимально возможного приближения проекта к требованиям заказчика и к выделенным ресурсам или путем снижения этих требований и увеличения заказчиком ресурсов на проект ПС. В крупных проектах систем, использующих комплексы про-

грамм, риски могут быть обусловлены дефектами функциональных характеристик самих ПС и их жизненного цикла, а также недостатками систем и внешней среды, в которой они используются. Основной ущерб от рисков ПС проявляется в последствиях их применения — ***в дефектах и недостатках функционирования систем и внешней среды***. Поэтому анализ и оценка рисков ПС должны быть тесно связаны с исследованием их проявления в системах, где они используются.

Риски ПС могут проявляться в процессах проектирования, разработки и сопровождения при изменении и развитии комплексов программ и при применении готового программного продукта по прямому назначению. Это приводит к необходимости анализа рисков функционирования ПС в условиях, различающихся:

— источниками и причинами ***угроз*** и опасного проявления рисков;

— классами, категориями, ***уязвимостью*** ПС и системы, вероятностью проявления и величиной последствий рисков;

— возможными и реализуемыми ***контрмерами*** для сокращения рисков и их эффективностью.

Оценка и измерение рисков во многих случаях характеризуется значительной неопределенностью и применением качественных метрик. Факторы и угрозы рисков могут быть распределены, и отличаться уязвимостью по этапам жизненного цикла ПС и системы. При анализе и управлении рисками целесообразно выделять, прежде всего, наиболее характерные этапы ЖЦ ПС: технико-экономическое обоснование проекта ПС; разработку требований спецификаций; проектирование; кодирование; тестирование и документирование.

Неопределенность оценок масштаба комплексов и компонентов программ является одним из важнейших факторов, ***влияющим на риски всего жизненного цикла проекта ПС*** (см. лекцию 5). Точность оценки размера нового ПС значительно повышается после формулирования начальных требований и спецификаций заказчиков, проведения анализа требований после завершения разработки предварительного проекта. На этапе концепции проекта ошибки оценки размера ПС уменьшаются приблизительно до 40%. Это вполне объяснимо, поскольку еще не уточнены структура и многие детали проекта. Эти вопросы могут быть разрешены во время разработки структуры и спецификаций требований к ПС, и тогда можно оценить размер ПС с точностью до 15—20%.

Как только структура ПС будет разбита с учетом выделения самых нижних, доступных уровней компонентов, может быть создан более точный показатель размера комплекса программ. При этом используются процессы измерения и суммирования. После завершения разработки и подтверждения проектных спецификаций при детальном проектировании комплекса программ может быть определена структура внутренних данных и функции программных компонентов. На этом этапе ***оценка размера проекта ПС*** может составить около 10%. Уточнения размеров ПС и компонентов могут быть решены к концу детального проектирования, однако при этом сохраняется неопределенность оценки размера комплекса программ около 5—10%, связанная с тем, насколько хорошо программисты понимают спецификации, в соответствии с которыми они должны кодировать программу.

В рассматриваемом ниже ***примере проекта ПС*** средние ошибки оценивания масштаба могут быть больше или меньше реальных значений, что по-разному отражается на ***характеристиках договора с заказчиком*** на проект, а также на факторах и типах рисков — рис. 10.5. Если масштаб проекта ПС в договоре ***завышен***, то следствием могут быть, прежде всего, ***риски — ущерб экономических категорий*** — превышения планируемых ресурсов: стоимости, трудоемкости, длительности разработки и числа необходимых специалистов. При этом заказчик терпит убытки, разработчик получает неоправданную прибыль. При реализации функциональной пригодности и конструктивных характеристик ПС разработчик может свободно использовать ресурсы и не предпринимать жестких мер для их экономии, что может отразиться на эффективности последующего применения комплекса программ.

Если при оценке масштаба проекта ПС его величина в договоре определена ***недостаточной — заниженной***, то основной ущерб — риски достаются разработчикам. Они вынуждены принимать жесткие меры для выполнения плановых сроков, выделенного бюджета работ при относительно малом числе специалистов, что угрожает рисками срыва сроков и нарушения стоимости проекта. Недооценка размера проекта комплекса программ и ресурсов для его реализации может резко увеличивать число дефектов в программах. Кроме того, ***ограниченные ресурсы*** для реализации в полном объеме функциональной пригодности могут отражаться на ее качестве и удобстве применения, а также на конструктивных характе-

ристиках: на безопасности, надежности, качестве взаимодействия с внешней средой и с пользователями, качестве документации и других эксплуатационных факторах.

Рис. 10.5

Оценки масштаба проекта ПС и рисков должны быть проанализированы и скорректированы для установления в договоре между заказчиками и разработчиками исходного компромиссного масштаба, допустимого для разработки первичных спецификаций требований с требуемым качеством.

Некоторые требования могут потребовать изменения (обычно увеличения) масштаба и соответственно ресурсов на этапах предварительного и детального проектирования для обеспечения возможности реализации требуемой функциональной пригодности с допустимыми рисками.

10.4. Риски при формировании требований к характеристикам сложных программных средств

Для продолжения разработки проекта после оценки рисков масштаба комплекса программ необходимо выполнить цикл поэтапного определения и формирования ***совокупности спецификаций требований к проекту ПС***. Первым этапом является создание концепции проекта ПС и ***комплекса первичных требований*** к иерархическому набору функций, на которые могут быть разбиты предполагаемые фактические компоненты ПС. В дальнейшем разбиение может детализироваться, формируя упрощенный или более точный уровень абстракции и взаимодействия компонентов. Поэтапная разработка спецификаций требований проекта ПС выполняется итерационно после первичной оценки масштаба проекта ПС и обусловленных такими оценками рисков, вследствие ошибок размера в большую или меньшую сторону (см. рис. 10.5). Ограничения при прогнозировании требований определяются, прежде всего, имеющимися данными, которые могут быть использованы в качестве исходных аналогов или обобщенных характеристик.

Для устранения или снижения рисков до допустимых пределов может быть необходимо изменение требований к функциональной пригодности, к конструктивным характеристикам и доступным ресурсам. Поэтому на этапах проектирования ***последовательно должны определяться***:

— при проектировании концепции и первичной оценке масштаба проекта — предварительные требования к назначению, функциональной пригодности и к номенклатуре необходимых конструктивных характеристик качества ПС;

— при предварительном проектировании — уточненная оценка масштаба проекта, требования к функциональным и конструктивным характеристикам качества с учетом общих ограничений ресурсов, перечень источников угроз и величины возможных рисков;

— при детальном проектировании — подробные спецификации требований к функциональным и конструктивным характеристикам качества с детальным учетом и распределением реальных ограниченных ресурсов, а также интегральные риски при их оптимизация по критерию качество/затраты.

На ***этапе создания концепции и системного анализа*** формируются цели разработки проекта ПС, выбираются методы и алгоритмы решения основных, функциональных задач, а также формулируются предварительные критерии качества создаваемых программ. При этом, естественно, встает вопрос о ресурсах, которые потребуются для достижения целей, и о возможности их реализации. Целенаправленная и методичная экспертная оценка возможного масштаба и ресурсов проекта ***уменьшает величину риска***, однако обычно она остается все-таки довольно большой.

До завершения разработки первичного технического задания на ПС могут быть сформулированы только ***приближенные исходные требования,*** отражающие объект разработки и условия его создания. После выбора технологии, средств автоматизации разработки и технологических ЭВМ появляется возможность достоверно учесть эти исходные данные для подготовки уточненного сценария разработки. Регистрация этих данных может служить дополнительным ориентиром для оценки полных ***ресурсов на разработку и возможных рисков***.

Разработку и утверждение спецификаций требований к функциональным характеристикам и качеству ПС с учетом анализа рисков целесообразно проводить ***итерационно на этапах предварительного и детального проектирования***. Чем крупнее и сложнее проект ПС и соответственно выше его стоимость, тем тщательнее следует разрабатывать требования к его характеристикам и распределять ресурсы на их реализацию. Поэтому при средней и относительно невысокой сложности ПС во многих случаях можно удовлетвориться подготовкой требований к комплексу программ с подробностью анализа, соответствующего предварительному проектированию. Для крупных и особо сложных проектов необходим более ***детальный анализ факторов и рисков*** при разработке требований и их оптимизация по критерию качество/затраты.

При первоначальном определении требований к функциональной пригодности и к конструктивным характеристикам заданные заказчиком огра-

ничения ресурсов не всегда могут учитывать ряд особенностей и характеристик проекта, что обусловит недопустимое снижение (или завышение) требований к некоторым характеристикам или рискам ПС. Кроме того, возможно, что некоторые характеристики противоречивы или принципиально нереализуемы в данном проекте. В результате ***несбалансированные требования*** и доступные ресурсы ***проявятся как риски*** — ущерб в виде потерь в качестве или в потребности дополнительных ресурсов (см. рис. 10.5).

В зависимости от сложности проекта окончательным результатом работ при предварительном или детальном проектировании должны быть детализированные и утвержденные требования к номенклатуре, свойствам и значениям качества и ***допустимым рискам проекта ПС***, которые достаточны для его полноценного рабочего проектирования и последующей эффективной эксплуатации. Однако на последующих этапах жизненного цикла и при конфигурационном управлении требования могут изменяться по согласованию между заказчиком и разработчиком, которые чаще всего приурочиваются к подготовке новой версии ПС. Для этого ***необходим мониторинг*** масштаба проекта, требований и реализаций характеристик и рисков в течение всего ЖЦ ПС. После определения назначения и функций ПС подготовка исходных данных и концепции проекта должны завершаться ***выделением номенклатуры приоритетных конструктивных характеристик***, имеющих достаточно сильное влияние на функциональную пригодность и сокращающих риски ПС.

Принципиальные и технические возможности, точность реализации свойств и измерения значений характеристик ПС, а также общие ресурсы конкретного проекта всегда ограничены в соответствии с их содержанием и возможностями заказчика и разработчиков. Это определяет ***рациональные диапазоны значений каждого риска***, которые могут быть выбраны для проекта ПС на основе требований заказчика, здравого смысла, а также путем анализа пилотных проектов и прецедентов в спецификациях требований реализованных проектов. При ограниченности ресурсов проекта ПС распределение приоритетов должно становиться более строгим и могут снижаться приоритеты характеристик, для реализации которых ресурсов недостаточно. В результате формируется полный ***набор требуемых функциональных и конструктивных характеристик, свойств и допустимых рисков в ЖЦ ПС*** (см. рис. 10.5).

Для разработчиков особенно важно формализовать требования к качеству и согласовать их риски с заказчиком при утверждении контракта и технического задания на проект ПС. Требования к функциональным характеристикам, качеству и допустимым рискам, утвержденные после предварительного проектирования, могут быть закреплены в техническом задании ***как обязательные для детального и рабочего проектирования***. Эти данные могут использоваться при последующем оценивании качества и реальных рисков, и при их сопоставлении с требованиями в процессе квалификационных испытаний или сертификации ПС. Однако для крупномасштабных и дорогих проектов может потребоваться уточнение требований к качеству при детальном проектировании с позиции улучшения соотношения значений качества и затрат ресурсов, которые необходимы или допустимы для их реализации в ЖЦ ПС.

Для заказчика и пользователей может иметь значение не только определение функциональной пригодности, но и оценка потенциального спроса на рынке конкретного программного продукта, а также его ***конкурентоспособности*** с другими аналогичными по функциям ПС с учетом его качества и стоимости. Это обстоятельство может определять необходимость уточнения требований к отдельным характеристикам не только для их реализации разработчиками в ЖЦ ПС, но также для оценивания интегрального качества готового программного продукта, поставляемого на рынок. Для заказчиков и пользователей интегральное качество и риски сосредоточиваются на функциональной пригодности и метриках в использовании. Для разработчиков важны, прежде всего, внутренние метрики и затраты ресурсов, при которых они достигаются. Поэтому при оценивании интегральных характеристик особо сложных ПС необходимо детально учитывать это различие целей и интересов их потребителей.

Обычно заказчики и разработчики первоначально устанавливают требования к каждой характеристике ПС без учета относительных затрат на их достижение, а также без детального анализа их совместного влияния на полную функциональную пригодность и риски у потребителей. Это может приводить к значительным перекосам и ***несбалансированным значениям требований к отдельным, взаимосвязанным характеристикам***, на которые нерационально используются ограниченные ресурсы ЖЦ ПС, или к неадекватно низким их значениям. В проектах крупномасштабных ПС это

может угрожать значительным повышением стоимости и рисков и/или снижением конкурентоспособности создаваемого программного продукта из-за недостаточного уровня отдельных показателей качества.

Атрибуты качества ПС имеют различные меры и шкалы, вследствие чего они в большинстве своем непосредственно ***несопоставимы между собой***. Они предварительно выбираются и согласовываются с заказчиком при последовательном, почти независимом анализе каждого атрибута качества, для использования в контракте и техническом задании. Для обобщенного оценивания качества необходим учет относительного влияния и риска каждой конструктивной характеристики ПС, на функциональную пригодность. При этом не всегда учитываются ресурсы для их реализации в конкретном ПС. Это часто приводит к выдвижению ряда нерациональных требований, которые значительно отличаются: либо по степени влияния на функциональную пригодность, либо по величине ресурсов, необходимых для их реализации. Для целенаправленного эффективного управления рисками сложного ПС при проектировании целесообразно иметь механизм объединения разнородных характеристик в некоторый интегральный показатель, отражающий их совокупное влияние на его функциональную пригодность. Таким образом, при разработке требований к характеристикам проекта выявилась проблема анализа системной эффективности ПС и обобщения его характеристик, а также оценивания совместного влияния конструктивных характеристик на функциональную пригодность ПС с учетом затрат на их реализацию.

Для управления рисками и детального сопоставительного оценивания выбранных характеристик качества целесообразно каждому из них присваивать коэффициент или приоритет влияния на функциональную пригодность. Группа квалифицированных экспертов из состава заказчиков, потенциальных пользователей и/или разработчиков должны оценивать и устанавливать значения таких ***коэффициентов — рисков (приоритетов) в пределах унифицированной шкалы***, например, от единицы до десяти, для конкретного проекта ПС. Точность определения коэффициентов вряд ли может превышать 10%, поэтому количество градаций шкалы целесообразно не больше десяти. Аналогично, по такой же шкале экспертами целесообразно оценивать относительные затраты ресурсов, которые следует выделять на реализацию сокращения рисков. Для каждого вида

рисков отношение коэффициента влияния на функциональную пригодность к относительным затратам на его достижение можно рассматривать как ***обобщенный уровень приоритета требований к сокращению риска***.

Значения приоритетов — рисков предназначены для указания относительной степени важности соответствующих свойств или атрибутов характеристик качества для функциональной пригодности проекта или допустимых ресурсов при их реализации. Для конкретного проекта ПС состав и значения приоритетов следует ***поэтапно адаптировать и уточнять*** с учетом их назначения и функций. Наивысший приоритет (10) следует интерпретировать как обязательное выполнение разработчиком соответствующего требования к указанному свойству или атрибуту качества с отсутствием риска. Низшее значение приоритета (1) означает, что данный малый риск может не учитываться в данном проекте. Промежуточные значения приоритетов должны отражать относительное влияние соответствующих атрибутов на функциональную пригодность и ее свойства с учетом доступных ресурсов на их реализацию. При этом возможно, что некоторые требования к атрибутам качества при их низких приоритетах могут не полностью реализоваться в реальном комплексе программ. Для проведения последовательного ***оценивания обобщенных приоритетов*** при управлении рисками и корректировке атрибутов качества ***конкретного проекта ПС*** целесообразно проводить:

— экспертную оценку коэффициента влияния, требуемой конструктивной характеристики качества на функциональную пригодность (в диапазоне 1—10);

— экспертную оценку относительных затрат ресурсов на реализацию требуемых значений качества без рисков (в диапазоне 1—10);

— оценку относительного коэффициента влияния каждой конструктивной характеристики на функциональную пригодность с учетом затрат на ее реализацию — обобщенный уровень приоритета, требуемого значения атрибута качества с учетом рисков (0,1—10).

Для крупномасштабных проектов комплексов программ при уточнении требований к качеству при детальном проектировании целесообразно использовать ***обобщенный уровень приоритета — риска***. При этом набор значений обобщенных уровней приоритетов для выбранных атрибутов качества конкретного проекта ПС можно разделить на ***три группы***:

— доминирующие характеристики и риски, оказывающие наибольшее влияние на функциональную пригодность при допустимых затратах (обобщенный приоритет > 8);

— показатели конструктивных характеристик и рисков, имеющие достаточное влияние на функциональную пригодность и значительные затраты на реализацию (обобщенный приоритет < 7, но > 4);

— характеристики качества, значения требований к которым не соответствуют их влиянию на функциональную пригодность и/или затратам на реализацию и могут не учитываться (обобщенный приоритет < 3).

Эти данные могут использоваться, прежде всего, ***как ориентиры*** для селекции и исключения из требований, конструктивных характеристик качества с особенно низкими обобщенными приоритетами рисков, в наименьшей степени влияющих на функциональную пригодность ПС и не оправдывающих больших затрат на реализацию. Анализ оставшихся характеристик качества и рисков может проводиться для выделения завышенных требований, а также, возможно, для снижения их значений и приближения их влияния к средним значениям. Кроме того, сравнительный анализ обобщенных приоритетов на основе отношения влияния на качество и затраты позволяет выделять атрибуты качества, отличающиеся большими затратами — рисками, не оправданными их степенью воздействия на функциональную пригодность. Подобные процедуры могут завершать разработку требований к свойствам, характеристикам качества и рискам при детальном проектировании крупномасштабных ПС.

Следует подчеркнуть, что выше анализировалось и оценивалось преимущественно ***изменение функциональной пригодности и снижение риска*** при совершенствовании конструктивных характеристик ПС. Однако для заказчика и пользователей доминирующее значение могут иметь номенклатура и особенности реализации некоторых основных функций комплекса программ, которые, как правило, требуют наибольших затрат и определяют основной эффект и риск от применения ПС, а также потенциальный уровень спроса на рынке. Если затраты на разработку ПС можно оценивать и прогнозировать с некоторой достоверностью, то эффективность применения и особенно будущий спрос на конкретный комплекс программ со стороны пользователей априори оценить трудно. Такие оценки могут проводиться на основе специальных маркетинговых исследова-

ний и опыта эксплуатации аналогичных комплексов программ или достаточно близких их прототипов. Это подтверждает целесообразность выделения для автономного анализа интегральных, конструктивных характеристик программного продукта и их влияния на функциональную пригодность.

На основе анализа и оценивания характеристик масштаба, набора требований, возможных затрат ресурсов и значений рисков в ЖЦ ПС ***следует определять***:

— целесообразно ли продолжать работы над конкретным проектом ПС или следует его прекратить вследствие больших рисков, недостаточных ресурсов специалистов, времени или трудоемкости (бюджета) разработки;

— при наличии достаточных ресурсов, следует ли провести маркетинговые исследования для определения рентабельности полного выполнения проекта ПС и создания программного продукта для поставки на рынок;

— достаточно ли полно и корректно формализованы концепция и требования к проекту ПС, на основе которых проводились оценки затрат и рисков, или их следует откорректировать и выполнить повторный анализ с уточненными исходными данными;

— есть ли возможность применить готовые повторно используемые компоненты ПС, в каком относительном объеме комплекса программ и рентабельно ли их применять в конкретном проекте ПС или весь проект целесообразно разрабатывать как полностью новый.

ЛЕКЦИЯ 11

ХАРАКТЕРИСТИКИ КАЧЕСТВА ПРОГРАММНЫХ СРЕДСТВ

11.1. Основные факторы, определяющие качество сложных программных средств

Общее представление о качестве ПС международным стандартом **ISO 9126:1-4:2002** рекомендуется описывать тремя взаимодействующими и взаимозависимыми ***метриками характеристик качества***, отражающими:

— внутреннее качество, проявляющееся в процессе разработки и других промежуточных этапов жизненного цикла ПС;

— внешнее качество, заданное требованиями заказчика в спецификациях и отражающееся характеристиками конечного продукта;

— качество при использовании в процессе нормальной эксплуатации и результативностью достижения потребностей пользователей с учетом затрат ресурсов.

Внутренние метрики в соответствии со стандартами могут применяться в ходе проектирования и программирования к компонентам ПС, таким, как спецификация или исходный программный текст. При разработке ПС промежуточные компоненты следует оценивать с использованием внутренних метрик, которые отражают функциональные и конструктивные свойства программ. Основная цель применения внутренних метрик — обеспечивать, чтобы разработчиками было получено требуемое внешнее качество. Рекомендуется использовать внутренние метрики, которые имеют наиболее сильные связи с приоритетными внешними метриками, чтобы они могли помогать при прогнозировании их достижимых значений. Внутренние метрики дают возможность разработчикам, испытателям и заказчикам, начиная с системного проектирования, прогнозиро-

вать качество жизненного цикла программ и заниматься вопросами технологического обеспечения качества до того, как ПС становится готовым к использованию продуктом. Измерения внутренних метрик используют свойства, категории, числа или характеристики элементов ПС, которые, например, имеются в процедурах исходного программного текста, в графе потока управления, в потоке данных и в описаниях изменения состояний памяти.

Внешние метрики используют меры ПС, отражающие поведение системы, частью которой они являются, путем испытаний, эксплуатации и наблюдения исполняемых программ или функционирования системы. Перед приобретением или использованием ПС его следует оценить с использованием метрик, основанных на реализации деловых и профессиональных целей, связанных с применением программного продукта в определенной организационной и технической среде. Внешние метрики обеспечивают заказчикам, пользователям и разработчикам возможность прослеживать и анализировать качество ПС в ходе испытаний или опытной эксплуатации. Подходящие внешние метрики специфицируются для получения числовых значений или категорий и свойств внутренних характеристик качества, чтобы их можно было использовать для проверки того, что промежуточные продукты в процессе разработки удовлетворяют внутренним спецификациям качества.

Метрики качества в использовании отражают, в какой степени продукт удовлетворяет потребности конкретных пользователей в достижении заданных целей. Эта метрика не отражена в числе шести базовых характеристик ПС, регламентируемых стандартом **ISO 9126-1** вследствие ее общности, однако рекомендуется для ***интегральной оценки*** результатов функционирования и применения комплексов программ в стандарте **ISO 9126-4**. Связь качества в использовании с другими характеристиками ПС зависит от ***задач и функций их потребителей*** (см. лекцию 6).

Стандарт **ISO 9126:1-4** — целесообразно использовать как основу для формального ***регламентирования характеристик качества в жизненном цикле проектов программных средств***. Модель характеристик качества ПС и компонентов состоит из шести групп базовых показателей, каждая из которых детализирована несколькими нормативными субхарактеристиками.

Функциональные возможности детализируются:

— пригодностью для применения по назначению;

— корректностью (правильностью, точностью) реализации требований;

— способностью к взаимодействию с компонентами и средой;

— защищенностью — безопасностью функционирования.

Надежность характеризуется:

— уровнем завершенности — отсутствием дефектов и ошибок;

— устойчивостью при наличии дефектов и ошибок;

— восстанавливаемостью после проявления дефектов;

— доступностью — готовностью реализации требуемых функций.

Эффективность рекомендуется отражать:

— временной эффективностью реализации комплекса программ;

— используемостью вычислительных ресурсов.

Применимость (практичность) предлагается описывать:

— понятностью функций и документации;

— простотой использования комплекса программ;

— изучаемостью процессов функционирования и применения.

Сопровождаемость представляется:

— анализируемостью — удобством для анализа предложений модификаций;

— изменяемостью компонентов и комплекса программ;

— тестируемостью изменений при сопровождении.

Мобильность (переносимость) предлагается отражать:

— адаптируемостью к изменениям среды;

— простотой установки — инсталляции после переноса;

— замещаемостью компонентов при корректировках комплекса программ.

Характеристики и субхарактеристики в стандарте определены кратко, без комментариев и подробных рекомендаций по их применению к конкретным системам и проектам ПС. Изложение имеет концептуальный характер и не содержит рекомендаций по выбору и упорядочению приоритетов, а также необходимого минимума критериев в зависимости от особенностей объекта, среды разработки, сопровождения и применения.

Для выбора характеристик качества ПС и достоверного сравнения их с требованиями, а также для сопоставления их значений между различны-

ми программными продуктами необходимы оценки, измерения и использование определенных ***мер и шкал***. Стандартами рекомендуется, чтобы было предусмотрено измерение каждой характеристики качества ПС (субхарактеристики или ее атрибута) с точностью и определенностью, достаточной для сравнений с требованиями технических заданий и спецификаций, и чтобы измерения были объективны и воспроизводимы. Следует предусматривать нормы допустимых ошибок измерения, вызванных инструментами и/или ошибками человека-эксперта. Чтобы измерения были объективными, должна быть документирована и согласована процедура для присвоения числового значения, свойства или категории каждому атрибуту программного продукта. Характеристики, субхарактеристики и атрибуты качества ПС с позиции возможности и точности их измерения можно разделить на ***три уровня детализации показателей***, особенности которых следует уточнять при их выборе:

— категорийные-описательные, отражающие набор свойств и общие характеристики объекта — его функции, категории ответственности, защищенности и важности, которые могут быть представлены номинальной шкалой категорий-свойств;

— количественные — представляемые множеством упорядоченных, числовых точек, отражающих непрерывные или дискретные закономерности и описываемые интервальной или относительной шкалой, которые можно объективно измерить и численно сопоставить с требованиями;

— качественные — содержащие несколько упорядоченных или отдельных свойств — категорий, которые характеризуются порядковой или точечной шкалой набора категорий (есть — нет, хорошо — плохо), устанавливаются, выбираются и оцениваются в значительной степени субъективно и экспертно.

К ***первому уровню*** относятся показатели качества, которые характеризуются наибольшим разнообразием значений — свойств программ и наборов данных и охватывают весь спектр классов, назначений и функций современных ПС. Эти свойства можно сравнивать только в пределах однотипных ПС и трудно упорядочивать по принципу предпочтительности. Среди стандартизированных показателей качества к этой группе, прежде всего, относится функциональная пригодность, являющаяся доминирующей характеристикой любых ПС. Номенклатура и значения всех остальных

ных показателей качества непосредственно определяются требуемыми функциями программного средства и, в той или иной степени, влияют на выполнение этих функций.

Функциональная пригодность — наиболее ответственная, объективно трудно формализуемая и оцениваемая в проекте характеристика комплексов программ. Данная характеристика связана с тем, ***какие*** основные и дополнительные функции и задачи должен решать программный продукт для удовлетворения потребностей пользователей, в то время как другие, ***конструктивные характеристики*** главным образом связаны с тем, ***как и при каких условиях*** заданные функции могут выполняться с требуемым качеством. Субхарактеристики и атрибуты функциональной пригодности можно характеризовать в основном свойствами, категориями и качественным описанием функций, для которых зачастую трудно определить численные меры и шкалы.

Ко ***второму уровню*** показателей качества относятся достаточно достоверно и объективно измеряемые численные характеристики ПС. Значения этих ***конструктивных характеристик*** обычно в наибольшей степени влияют на функциональную пригодность в использовании ПС. Поэтому выбор и обоснование их требуемых значений должно проводиться наиболее аккуратно и достоверно уже при проектировании ПС. Их субхарактеристики могут быть описаны упорядоченными шкалами объективно измеряемых значений, требуемые численные величины которых могут быть установлены и выбраны заказчиками или пользователями ПС. Такими характеристиками являются надежность и эффективность комплексов программ. Эти величины могут выбираться и фиксироваться в техническом задании или спецификации требований и сопровождаться методикой объективных, численных измерений при квалификационных испытаниях для сопоставления с требованиями. Длительность решения основных задач, пропускная способность по числу их решений за некоторый интервал времени, длительность ожидания результатов (отклика) и некоторые другие характеристики динамики функционирования ПС могут быть выбраны и установлены количественно в спецификациях требований заказчиком.

Третий уровень стандартизированных показателей качества ПС трудно полностью описать измеряемыми количественными значениями и их некоторые субхарактеристики и атрибуты имеют описательный, качествен-

ный вид. В зависимости от функционального назначения ПС по согласованию с заказчиком можно определять экспертно степень необходимости (приоритет) этих свойств и балльные значения уровня реализации их атрибутов в жизненном цикле конкретного ПС.

Проблема состоит в выявлении факторов, от которых они зависят, в создании методов и средств уменьшения их влияния на функциональную пригодность ПС, а также в эффективном распределении ограниченных ресурсов для обеспечения необходимого качества функционирования комплекса программ, равнопрочного при всех реальных негативных воздействиях. Комплексное, скоординированное применение этих методов и средств в процессе создания, развития и применения ПС позволяет исключать проявления ряда негативных факторов или значительно ослаблять их влияние. Тем самым уровень достигаемого ***качества функционирования ПС может быть предсказуемым и управляемым***, непосредственно зависящим от ресурсов, выделяемых на его достижение, а главное, от системы качества и эффективности технологии, используемых на всех этапах жизненного цикла ПС.

11.2. Свойства и атрибуты качества функциональных возможностей сложных программных средств

Системная эффективность целевого применения программных средств определяется степенью удовлетворения потребностей определенных лиц — заказчиков и/или пользователей, которую во многих случаях желательно измерять экономическими категориями: прибылью, стоимостью, трудоемкостью, предотвращенным ущербом-риском, длительностью применения и т.п. Решение этих задач должно быть направлено на обеспечение высокой функциональной пригодности ПС, ***путем сбалансированного улучшения, конструктивных характеристик качества в условиях ограниченных ресурсов на ЖЦ***. Для этого в процессе системного анализа при подготовке технического задания и требований спецификаций значения атрибутов и субхарактеристик качества должны выбираться с учетом их влияния на функциональную пригодность. Ориентирами могут служить диапазоны изменения атрибутов конструктивных характеристик ка-

чества ПС, границы количественных или качественных шкал которых сверху и снизу могут быть выбраны ***на основе следующих принципов***:

— предельные значения характеристик качества должны быть ограничены сверху допустимыми или рациональными затратами ресурсов на их достижение при разработке и совершенствовании ПС;

— наибольшие допустимые затраты ресурсов, например труда и времени для реализации конструктивных характеристик, должны обеспечивать функциональную пригодность жизненного цикла ПС на достаточно высоком уровне;

— допустимые наихудшие значения отдельных конструктивных характеристик качества могут соответствовать значениям, при которых заметно начинает снижаться функциональная пригодность при применении ПС;

— ограниченные значения отдельных конструктивных характеристик качества не должны негативно отражаться на возможных высоких значениях других приоритетных характеристик.

Способность ПС обеспечивать решение конкретных задач, удовлетворяющих установленные потребности заказчиков и пользователей при применении комплекса программ в заданных условиях, отражена в стандарте **ISO 9126:1** характеристикой — ***функциональные возможности***. В ней на первом месте стоит самая важная субхарактеристика ЖЦ ПС — ***функциональность*** или ***функциональная пригодность***. Кроме нее в состав функциональных возможностей включены, по существу, конструктивные субхарактеристики: корректность и способность к взаимодействию. Более сложно классифицировать защищенность-безопасность, функции которой непосредственно и органически связаны с конкретными особенностями функциональной пригодности (см. п. 11.5).

Функциональная пригодность — это набор и описания атрибутов, определяющих ***назначение, основные, необходимые и достаточные функции ПС***, заданные техническим заданием и спецификациями требований заказчика или потенциального пользователя. В процессе проектирования комплекса программ атрибуты функциональной пригодности должны конкретизироваться в спецификациях на ПС в целом и на компоненты. Атрибутами этой характеристики качества могут быть функциональная полнота решения заданного комплекса задач, степень покрытия функциональных требований спецификациями и их стабильность при совершенствовании

ПС, число реализуемых требований заказчиков. Кроме них функциональную пригодность отражают множество различных специализированных критериев, которые тесно связаны с конкретными решаемыми задачами и сферой применения комплекса программ. Их можно рассматривать как частные критерии или как факторы, влияющие на основной показатель качества ПС.

Эта характеристика может значительно модифицироваться в жизненном цикле ПС, и соответственно изменяться конкретное содержание базовых функций, которые подлежат применению и оцениванию. На последовательных этапах ЖЦ функции промежуточных продуктов (спецификаций компонентов, модулей, текстов программ и т.п.) должны оцениваться на соответствие описаниям в отдельных, частных документах. Это позволяет поэтапно формализовать применяемые субхарактеристики и атрибуты функциональной пригодности. Такими атрибутами могут быть: функциональная адекватность программ документам и декларированным требованиям, утвержденным заказчиком; степень покрытия тестами исходных требований; полнота и законченность реализации этих требований; точность выполнения требований детальных спецификаций на функциональные компоненты ПС.

Среди всего многообразия функциональных характеристик программных средств можно выделить ***две группы***, одна из которых отражает разнообразные специфические особенности, связанные непосредственно с назначением, функциями и сферой применения ПС, а вторая — доступна для частичной унификации состава и структуры и для оценивания стандартизированными методами. Эта вторая группа характеризует ***ряд базовых, инвариантных свойств качества***, которые позволяют определять некоторые субхарактеристики функциональной пригодности ПС, независимо от конкретных целей и сфер применения. С этой позиции функциональная пригодность определяется качеством взаимосвязи и согласованности последовательных формулировок содержания и реализации основных фрагментов в цепочке стандартизированных требований технического задания на ПС: ***целей — назначения — функций — исходной информации — результатов для пользователей***, определяющих, что:

— описание целей программного средства корректно переработано в подробное описание его назначения и внешней среды применения;

— назначение ПС полностью и корректно детализировано в требованиях к функциям комплекса программ и его компонентов;

— реализация требований к функциям ПС обеспечена достоверным и адекватным составом и содержанием исходной информации от объектов внешней среды;

— реализация функций ПС способна подготавливать всю требуемую и достаточно корректную информацию для пользователей и объектов внешней среды.

Прослеживание качества результатов при углублении и детализации этих описаний и обеспечение их взаимной адекватности является основой для определения группы показателей функциональной пригодности. Они должны максимально полно и точно представляться в контракте, техническом задании и спецификациях требований к ПС и к его функциональным компонентам, так как устранение их дефектов требует особенно крупных ресурсов.

Любые ПС, прежде всего, должны иметь экономическую, техническую, научную или социальную эффективность применения, которая в проектах должна отражать основную ***цель их жизненного цикла*** в системе. Эта ***системная эффективность*** может быть описана количественно или качественно, в виде набора полезных свойств программного продукта, их отличий от имеющихся у других комплексов программ, а также источников возможной эффективности. В результате должны быть формализованы цель использования и набор главных требований заказчика и пользователей при приобретении программного продукта, а также предполагаемая его сфера применения и назначение. Полнота и точность представления этой характеристики ПС может оцениваться в основном экспертно и является исходной для прослеживания всех последующих, производных свойств и атрибутов функциональной пригодности.

Цель и назначение ПС детализируются и формализуются в ***требованиях к функциям*** компонентов и всего комплекса программ, способного реализовать декларированные цели:

— соответствие комплекса программ функциям системы;

— соответствие автоматизируемых функций и комплексов задач назначению ЖЦ ПС;

— общие технические требования к реализации функций, компонентов и задач.

Адекватность и полнота отражения требуемыми функциями сформулированного назначения ПС является характеристикой, определяющей потенциальную ***возможность реализации его функциональной пригодности*** в целом. Прослеживание детализации и покрытия целей, требованиями к функциям сверху вниз (начиная от целей ПС), а также конкретизация и корректировка целей снизу вверх от потенциально реализуемых функций компонентов должны обеспечивать адекватность и качество этой части декларируемой основы функциональной пригодности.

Функции ПС реализуются в определенной аппаратной, операционной и пользовательской внешней среде системы, характеристики которых существенно влияют на функциональную пригодность. Для выполнения требуемых функций комплекса программ необходима ***адекватная исходная информация от объектов внешней среды***, содержание которой должно полностью обеспечивать реализацию декларированных функций. Полнота формализации номенклатуры, структуры и качества входной информации для выполнения требуемых функций является одной из важных составляющих при определении функциональной пригодности ПС в соответствующей внешней среде.

Цель и функции ПС реализуются тогда, когда ***выходная информация*** достигает потребителей — объектов или операторов-пользователей, с требуемым содержанием и качеством, достаточным для обеспечения ее эффективного применения. Содержательная часть этой информации определяется конкретными задачами системы, основными технико-экономическими и/или социальными показателями функционирования системы и отражается метриками в использовании. ***Степень покрытия*** всей выходной информацией: целей, назначения и функций ПС для пользователей — следует рассматривать как ***основную меру качества функциональной пригодности***. Прослеживание и оценивание адекватности и полноты состава выходной информации снизу вверх к назначению ПС должны завершать выбор базовых субхарактеристик качества функциональной пригодности, независимо от сферы применения системы.

В процессе проектирования в составе функциональной пригодности могут быть выделены две группы базовых субхарактеристик, определяющие ***функциональные и структурные требования и особенности ПС***. При формализации и выборе ***функциональных требований*** следует, возможно, четко формулировать в документах контракта:

— экономические, организационные, технические и/или социальные стратегические цели всего жизненного цикла ПС и его компонентов;

— назначение, внешнюю среду и условия эффективного применения ПС;

— системную эффективность и в том числе требуемые технико-экономические показатели применения ПС в составе системы;

— функциональные задачи основных компонентов и ПС в целом, а также системную эффективность каждого;

— необходимое и достаточное качество и временной регламент решения каждой функциональной задачи;

— соответствие ПС и его компонентов стандартам и нормативным документам на проектирование и применение;

— ограничения параметров внешней среды и условий для применения ПС, гарантирующие требуемые характеристики функциональной пригодности.

Функциональная пригодность в течение жизненного цикла ПС зависит от ***структурных (архитектурных) характеристик***, которые должны отражаться в требованиях технического задания и/или спецификаций на компоненты и ПС в целом:

— соответствие функций и структуры программного средства аппаратной и операционной среде и их ограниченным ресурсам;

— соответствие правил структурного построения комплекса, функциональных компонентов и модулей типовым требованиям к архитектуре ПС и его компонентов, а также к уровню покрытия ими заданных функций комплекса программ;

— состав, структура и способы организации данных, а также требования к обмену данными между компонентами ПС должны быть адекватны организации информационного обеспечения и функциям системы;

— должны быть заданы временной регламент и характеристики процесса динамической реализации автоматизированных функций;

— должно быть определено допустимое время задержки выдачи результатов решения задач;

— должны быть предусмотрены и реализованы требования к контролю, хранению, обновлению и восстановлению программ и данных.

В ряде систем особое значение для функциональной пригодности имеет системное проектирование ***организации информационного обеспечения и базы данных*** (см. п. 11.4). При этом должны быть определены:

— назначение базы данных и состав информации в ней;

— совместимость ПС с другими системами по источникам и потребителям информации базы данных;

— организация сбора, передачи, контроля и корректировки информации базы данных;

— интенсивность и объемы потоков информации базы данных.

После формулирования требований к функциональной пригодности нового проекта ПС обычно целесообразно попытаться найти готовые ПС с требуемыми функциями. Однако множество мелких конструктивных, казалось бы, второстепенных требований и нестандартизированных интерфейсов зачастую затрудняют непосредственное применение в новом проекте готовых компонентов с подходящими функциями. Это определяет необходимость тщательного оценивания для новых проектов всего набора приоритетных конструктивных характеристик качества в ПС с подобными характеристиками функциональной пригодности.

В наибольшей степени функциональная пригодность во многих случаях зависит от ***корректности и надежности ПС***. Значительные трудности при создании ориентиров для выбора мер и шкал характеристик качества проявляются при анализе корректности, способности к взаимодействию и защищенности — безопасности ПС.

Правильность — корректность: это способность ПС обеспечивать правильные (или приемлемые) результаты для пользователей. Эталонами для выбора требований к корректности при проектировании могут быть: верифицированные и взаимоувязанные требования к функциям комплекса, компонентов и модулей программ, а также правила их структурного построения, организация взаимодействия и интерфейсов (таблица 11.1). Эти требования при разработке должны быть прослежены сверху вниз до модулей и использоваться как эталоны при установлении необходимой корректности соответствующих компонентов. В процессе проектирования и разработки модулей и групп программ применяются частные структурные критерии корректности, которые включают корректность структуры программ, обработки данных и межмодульных интерфейсов. Каждый из частных критериев может характеризоваться несколькими методами измерения качества и достигаемой степенью корректности программ: детерминировано, стохастически или в реальном времени.

Таблица 11.1

Субхарактеристики, атрибуты качества и свойства для выбора функциональных возможностей программных средств

Субхарактеристики	Атрибуты качества и свойства
Корректность	— соответствие требований к функциям ПС требованиям к информационной системе; — соответствие требований к функциональным компонентам требованиям к функциям ПС; — соответствие текстов программ требованиям к функциональным компонентам ПС; — соответствие объектного кода исходному тексту программ функциональных компонентов ПС; — степень покрытия тестами функций и возможных маршрутов исполнения программ
Способность к взаимодействию	— с операционной системой и аппаратной средой; — с внешней средой системы и с пользователями; — между программными компонентами; — между компонентами распределенных информационных систем
Защищенность	— соответствие критериям и требованиям защиты от предумышленных угроз безопасности ПС; — соответствие методам и средствам защиты от проявления случайных дефектов программ и данных; — обеспечение эффективности оперативных методов защиты и восстановления при проявлениях и реализации угроз безопасности; — соответствие стандартам и нормативным документам на защиту от различных типов угроз безопасности; — обеспечение равнопрочной защиты в соответствии с опасностью угроз и доступностью ресурсов для защиты

Требования к субхарактеристике ***корректность*** могут представляться в виде описания двух основных свойств, которым должны соответствовать все программные компоненты и ПС в целом. ***Первое требование*** состоит в выполнении определенной степени (%) прослеживаемости сверху вниз реализации требований технического задания и спецификации на ПС при последовательной детализации описаний программных компонентов вплоть до текстов и объектного кода программ.

Второе требование заключается в выборе степени и стратегии покрытия тестами структуры и функций программных компонентов, сово-

купности маршрутов исполнения модулей и всего комплекса программ для последующего процесса верификации и тестирования, достаточного для функционирования ПС с необходимым качеством и точностью результатов, при реальных ограничениях ресурсов на тестирование. Мерой выбранной корректности может быть относительное число протестированных функций и маршрутов, которое может измеряться в процентах от общего числа исполняемых. Опыт показывает, что зачастую в готовом, сложном ПС оказываются протестированными только около 50—70% функций и маршрутов, и практически очень трудно эту величину довести до 90—95%. Косвенно эту величину при определенной автоматизации процессов и квалификации специалистов отражает трудоемкость и длительность тестирования, что непосредственно влияет на функциональную пригодность ПС.

Способность к взаимодействию — состоит в свойстве ПС и его компонентов взаимодействовать с одним или большим числом определенных компонентов внутренней и внешней среды (см. табл. 11.1). При выборе и установлении при проектировании способности программных и информационных компонентов к взаимодействию ее можно оценивать объемом технологических изменений в ПС, которые необходимо выполнять при дополнении или исключении некоторой функции или компонента, когда отсутствуют изменения операционной, аппаратной или пользовательской среды. С этим показателем связана корректность и унифицированность межмодульных интерфейсов, которые определяются двумя видами связей: по управлению и по информации.

Требования к характеристике способность к взаимодействию могут быть достаточно полно формализованы как набор свойств и утверждены в процессе системного проектирования, с некоторыми уточнениями на последующих этапах. Их основой являются ссылки на нормативные документы, на интерфейсы открытых систем или на выбранные для конкретного проекта стандарты де-факто. При выборе свойств программных компонентов, обеспечивающих способность к взаимодействию в конкретном проекте ПС, следует оценивать величину вычислительных ресурсов, необходимых для их реализации. При этом важно учитывать возможность повторного использования апробированных компонентов и переноса на различные платформы.

Унификация свойств интерфейсов на взаимодействие с внутренней, внешней средой и с пользователями должна отражаться в специальных разделах технологической документации и иметь возможность проверки заказчиком и/или экспертами по документам и текстам программ. Эта характеристика состоит в описании свойств и практически не влияет на качество функционирования текущей версии ПС. Степень унификации интерфейсов может измеряться их относительным числом или объемом текста (например, в процентах от объема программ), которые подвергаются изменениям при любых корректировках взаимодействия программ. Ряд общих понятий, методов и функций, которые могут рассматриваться как достаточно полная база и набор свойств компонентов, обеспечивающих высокую способность к взаимодействию, обобщены в ***концепции, методах и стандартах открытых систем***.

Защищенность и безопасность функционирования — одна из наиболее трудно формализуемых характеристик качества сложных ПС, которая занимает исключительное по важности положение среди всех конструктивных характеристик комплексов программ. Цели, назначение и функции защиты тесно связаны с особенностями функциональной пригодности каждого ПС. Разработка и формирование требований к свойствам защищенности должны осуществляться на основе потребностей эффективной реализации назначения и функций ПС при различных, реальных угрозах. В процессе системного анализа и проектирования должны быть выявлены потенциальные предумышленные и случайные угрозы функционированию ПС и установлен необходимый уровень защиты от них данного комплекса программ. В соответствие с этим уровнем заказчиком выбирается и устанавливается стандартизированная ***категория защищенности и безопасности ПС*** и необходимый набор методов, свойств и средств защиты с учетом ограниченных ресурсов на их реализацию. В результате сформированные требования должны обеспечивать равнопрочную защиту от реальных угроз и реализацию необходимых мер контроля и подтверждения целостности и характеристик качества функциональной пригодности комплекса программ в условиях проявления различных угроз безопасности функционирования ПС (см. п. 11.5).

11.3. Конструктивные характеристики качества сложных программных средств

Конструктивные характеристики разделены на две группы: количественные и качественные, которые различаются возможностями конкретизации мер и шкал. ***Две группы стандартизированных характеристик качества ПС*** — Надежность и Эффективность в наибольшей степени доступны ***количественным измерениям***. Для них в таблице 11.2 представлены ***примеры возможных мер и шкал измерения*** основных количественных атрибутов субхарактеристик качества. Они могут служить ориентирами при выборе и установлении требуемых значений этих показателей качества в спецификациях ПС.

Таблица 11.2

Основные количественные характеристики программных средств и их атрибуты

Характеристики качества	Мера	Шкала
Надежность		
Завершенность:		
наработка на отказ при отсутствии рестарта;	Часы	10—1000
степень покрытия тестами функций и структуры программ	%	50—100
Устойчивость:		
наработка на отказ при наличии автоматического рестарта;	Часы	10—1000
относительные ресурсы на обеспечение надежности и рестарта	%	10—90
Восстанавливаемость:		
длительность восстановления	Минуты	10^{-2} — 10
Доступность-готовность:		
относительное время работоспособного функционирования	Вероятность	0,9—0,999
Эффективность		
Временная эффективность:		
время отклика — получения результатов на типовое задание;	Секунды	0,1—100
пропускная способность — число типовых заданий, исполняемых в единицу времени	Число в минуту	1—1000
Используемость ресурсов:		
относительная величина использования ресурсов ЭВМ при нормальном функционировании программного средства	Вероятность	0,7—0,95

Надежность: свойства комплекса программ обеспечивать достаточно низкую ***вероятность потери работоспособности — отказа*** в процессе

функционирования ПС в реальном времени. Основные атрибуты надежности могут быть объективно измерены и сопоставлены с требованиями. Требования к значениям атрибутов субхарактеристики завершенность — допустимой наработки на отказ — устанавливаются при отсутствии автоматического рестарта и при наличии администратора, контролирующего работоспособность ПС. Применением программно-аппаратных механизмов автоматического рестарта эта наработка при проявлении отказов может быть повышена, т.е. при некоторых отказах возможно их автоматическое обнаружение и оперативное восстановление работоспособности, вследствие чего значения устойчивости и наработки на отказ возрастают. Это должно учитываться при определении требований к коэффициенту готовности — вероятности застать ПС в работоспособном состоянии. Так же как при формировании требований к корректности (см. табл. 11.1), для надежности большое значение имеет покрытие тестами в процессе отладки структуры и функций программных компонентов и ПС в целом.

Надежность функционирования программ является ***понятием динамическим***, проявляющимся во времени, и существенно отличается от понятия статической корректности программ. Надежность ПС наиболее полно характеризуется устойчивостью или способностью к безотказному функционированию и восстанавливаемостью работоспособного состояния после произошедших сбоев или отказов. В свою очередь, устойчивость зависит от степени покрытия тестами функций и структуры программ, от уровня неустраненных дефектов и ошибок (завершенность) и от способности ПС реагировать на их проявления так, чтобы это не отражалось на показателях надежности. Последние определяются эффективностью контроля данных, поступающих из внешней среды и от средств обнаружения аномалий функционирования ПС. В реальных условиях по различным причинам исходные данные могут попадать в области значений, не проверенные при разработке и испытаниях, а также не заданные требованиями спецификации и технического задания, вызывающие сбои и отказы. При этом некорректная программа может функционировать совершенно надежно.

Завершенность: свойство ПС не попадать в состояния отказов вследствие ошибок и дефектов в программах и данных. Количество или плотность проявления скрытых дефектов и ошибок непосредственно отражает-

ся на длительности нормального функционирования комплекса программ между отказами. Завершенность можно характеризовать наработкой (длительностью) на отказ (при отсутствии автоматического восстановления — рестарта), измеряемой обычно часами. На эту субхарактеристику влияют только отказы, вследствие проявившихся дефектов. Они могут быть обусловлены неполным тестовым покрытием при испытаниях компонентов и ПС в целом, а также недостаточной завершенностью тестирования их функций.

Устойчивость к дефектам и ошибкам: свойство ПС автоматически поддерживать заданный уровень качества функционирования при проявлениях дефектов и ошибок или нарушениях установленного интерфейса. Для этого в ПС должна вводиться временная, программная и информационная избыточность, реализующая оперативное обнаружение дефектов и ошибок функционирования, их идентификацию и автоматическое восстановление (рестарт) нормального функционирования ПС. Эффективное, оперативное устранение проявления дефектов, ошибок и некорректного взаимодействия с операционной и внешней средой определяют субхарактеристику — устойчивость комплексов программ.

Восстанавливаемость: свойство ПС в случае отказа возобновлять требуемый уровень качества функционирования, а также исправлять поврежденные программы и данные. После отказа ПС иногда бывает неработоспособно в течение некоторого времени, продолжительность которого определяется его восстанавливаемостью. Для этого необходимы вычислительные ресурсы и время на выявление неработоспособного состояния, диагностику причин отказа, а также на реализацию процессов восстановления. Основными показателями процесса восстановления являются его длительность и вероятностные характеристики. Восстанавливаемость характеризуется также полнотой восстановления нормального функционирования программ в процессе ручного или автоматического их перезапуска — рестарта. Перезапуск должен обеспечивать возобновление нормального функционирования ПС, на что требуются ресурсы ЭВМ и время, которые можно характеризовать относительной величиной (% от общих ресурсов).

Доступность или готовность: свойство ПС быть в состоянии выполнять требуемую функцию в данный момент времени при заданных

условиях использования. Внешне доступность может оцениваться относительным временем, в течение которого ПС находится в работоспособном состоянии, в пропорции к общему времени применения. Следовательно, доступность — комбинация завершенности (от которой зависит частота отказов), устойчивости к ошибкам и восстанавливаемости, которые в совокупности обусловливают длительность простоя для восстановления после каждого отказа, а также длительность наработки на отказ. Обобщение характеристик отказов и восстановления производится в критерии ***коэффициент готовности***. Этот показатель отражает вероятность иметь восстанавливаемые программы и данные в работоспособном состоянии в произвольный момент времени.

Нижняя граница шкалы атрибутов ***надежности*** в таблице 11.2 отражена значениями, при которых резко уменьшается функциональная пригодность и использование данного типа ПС становится неудобным, опасным или нерентабельным. Примером таких наихудших, предельных величин для многих классов ПС могут быть наработка на отказ менее десяти часов, коэффициент готовности ниже 0,9 и время восстановления более десяти минут. С другой стороны, наилучшие значения этих атрибутов практически ограничены теми ресурсами, которые могут быть выделены для их достижения при разработке и эксплуатации. Вычислительные и программные ресурсы объектной ЭВМ на непосредственное обеспечение надежности функционирования ПС обычно находятся в диапазоне от 10% до 90%, причем последние значения соответствуют критическим, особо высоконадежным системам. Даже для таких критических программных средств редко наработка на отказ превышает несколько тысяч часов, коэффициент готовности не выше 0,999, а время восстановления при отказах не меньше нескольких секунд.

Эффективность: в стандарте **ISO 9126** отражены две субхарактеристики качества — временная эффективность и используемость ресурсов ЭВМ, которые рекомендуется описывать в основном количественными атрибутами, характеризующими динамику функционирования компонентов ПС. В этой стандартизированной характеристике отражается только ***частная конструктивная эффективность*** использования ресурсов ЭВМ, которую не следует смешивать с ***системной эффективностью*** функциональной пригодности ПС при применении в конкретной системе.

Основные требования к атрибутам характеристики ***временная эффективность использования вычислительных ресурсов системы*** сосредоточены на наиболее критичных показателях производительности и длительности решения функциональных задач. В отличие от объемов памяти, временные характеристики труднее устанавливать и измерять, и их ограниченность сильнее влияет на функциональную пригодность ПС. Обычно для оперативной работы пользователей важно иметь малое время отклика из ЭВМ после получения типового задания и начала решения требуемой функциональной задачи. Требуемая пропускная способность решения функциональных задач зависит от их содержания и числа действующих пользователей. Используемость ресурсов памяти и производительности вычислительных средств могут устанавливаться исходя, с одной стороны, из экономической целесообразности применения наиболее дешевой, с минимальными ресурсами ЭВМ, загрузка которой будет в среднем не ниже 0,5. С другой стороны, высокая загрузка (выше 0,9) может приводить к нежелательной задержке или даже потере заданий при случайном, кратковременном повышении их интенсивностей, что может негативно отражаться на функциональной пригодности.

Временная эффективность: свойства ПС, характеризующие требуемые времена отклика и обработки заданий, а также производительность решения задач с учетом количества используемых вычислительных ресурсов в установленных условиях. Эти ресурсы могут включать другие программные продукты, аппаратные средства, средства телекоммуникации и т.п. Временная эффективность ПС определяется длительностью выполнения заданных функций и ожидания результатов в средних и/или наихудших случаях, с учетом приоритетов задач. Она зависит от скорости обработки данных, влияющей непосредственно на интервал времени завершения конкретного вычислительного процесса, и от пропускной способности — производительности, т.е. от числа заданий, которое можно реализовать на данной ЭВМ в заданном интервале времени (см. табл. 11.2). Эти показатели качества тесно связаны с дисциплиной диспетчеризации и временем реакции (отклика) ПС на задания при решении различных функциональных задач. Величина этого времени зависит от длительности решения совокупности задач центральным процессором ЭВМ, от затрат времени на обмен с внешней памятью, на ввод и вывод данных и от

длительности ожидания в очереди до начала решения задачи. Эта субхарактеристика тесно связана с длительностью обработки типового задания, а также с интервалом времени решения типовых или наиболее часто вызываемых функциональных задач данным ПС. Пропускная способность комплекса программ на конкретной ЭВМ отражается числом сообщений или заданий на решение определенных задач, обрабатываемых в единицу времени, зависящую от характеристик внешней среды.

Используемость ресурсов: степень загрузки доступных вычислительных ресурсов в течение заданного времени при выполнении функций ПС в установленных условиях. Ресурсная экономичность отражается занятостью ресурсов центрального процессора, оперативной, внешней и виртуальной памяти, каналов ввода-вывода, терминалов и каналов сетей связи. Эта величина определяется структурой и функциями ПС, а также архитектурными особенностями и доступными ресурсами ЭВМ. В зависимости от конкретных особенностей ПС и ЭВМ при выборе атрибутов может доминировать либо величина абсолютной занятости ресурсов различных видов, либо относительная величина использования ресурсов каждого вида при нормальном функционировании ПС. Ресурсная экономия влияет не только на стоимость решения функциональных задач, но зачастую, особенно для встраиваемых ЭВМ, определяет принципиальную возможность полноценного функционирования конкретного ПС в условиях реально ограниченных вычислительных ресурсов. Несмотря на быстрый рост доступных ресурсов памяти и производительности ЭВМ, часто потребности в них для решения конкретных задач ПС обгоняют их техническое увеличение, и задача оценки и эффективного использования вычислительных ресурсов остается актуальной.

Качественным анализом с учетом влияния на функциональную пригодность можно определить предельные значения для основных атрибутов конструктивной характеристики — ***эффективность***. Используемость вычислительных ресурсов памяти и производительности ЭВМ для каждой из функциональных задач или прикладных программ может составлять несколько процентов. Однако для всего комплекса программ стабильное использование ресурсов ЭВМ ниже 50—70% нерентабельно и позволяет, в принципе, перейти на более дешевую ЭВМ с меньшими ресурсами. В то же время использование ресурсов более чем на 95% может приводить к

значительным задержкам или отказам при решении низкоприоритетных задач. При нестационарных потоках заданий на решение основных, функциональных задач ПС необходимы некоторые резервы памяти и производительности ЭВМ, что определяет рациональные значения используемости ресурсов до 80—90% от максимальных значений. Атрибут временной эффективности — время отклика на задание пользователя непосредственно зависит от решаемых функциональных задач и в общем случае может устанавливаться в диапазоне от 0,1 секунды до нескольких десятков секунд. Эти значения зависят от динамических характеристик объектов внешней среды, для которых решаются функциональные задачи ПС. В административных системах может быть допустимо среднее время отклика в несколько секунд, а для оперативного управления динамическими объектами (самолетами, ракетами) оно сокращается до десятых и сотых долей секунды.

Три группы конструктивных характеристик качества ПС — практичность, сопровождаемость и мобильность трудно измерять количественно, и они доступны в основном ***качественным оценкам их свойств***. В некоторых проектах для субхарактеристик сопровождаемости и мобильности при системном проектировании могут доминировать технико-экономические меры трудоемкости (человеко-часы) и длительности (часы) для процедур, обеспечивающих реализацию атрибутов этих субхарактеристик. Однако для ряда атрибутов в этой группе характеристик приходится применять порядковые меры экспертных балльных шкал с небольшим числом (2—4) градаций. В таблице 11.3 представлены ***примеры возможных мер и шкал измерения*** основных субхарактеристик и их атрибутов качества. Они могут служить ориентирами при выборе и установлении требуемых значений этих показателей качества в спецификациях ПС.

Таблица 11.3

Основные качественные характеристики программных средств и их атрибуты

Характеристики качества	Мера	Шкала
Практичность ***Понятность:*** четкость концепции ПС; демонстрационные возможности; наглядность и полнота документации	Порядковая	Отличая; хорошая; удовлет.; неудовлет.

Окончание табл. 11.3

Характеристики качества	Мера	Шкала
Простота использования:		
простота управления функциями;	Порядковая	Отличая;
комфортность эксплуатации;		хорошая;
		удовлет.;
		неудовлет.
среднее время ввода заданий;	Секунды	1—1000
среднее время отклика на задание.	Секунды	1—1000
Изучаемость		
трудоемкость изучения применения ПС;	Чел.-часы	1—100
продолжительность изучения;	Часы	1—1000
объем эксплуатационной документации;	Страницы	10—1000
объем электронных учебников	Кбайты	100—10000
Сопровождаемость		
Анализируемость:		Отличая;
стройность архитектуры программ;		хорошая;
унифицированность интерфейсов;	Порядковая	удовлет.;
полнота и корректность документации		неудовлет.
Изменяемость:		
трудоемкость подготовки изменений;	Чел.-часы	1—1000
длительность подготовки изменений.	Часы	1—1000
Тестируемость:		
трудоемкость тестирования изменений;	Чел.-часы	1—1000
длительность тестирования изменений	Часы	1—100
Мобильность		
Адаптируемость:		
трудоемкость адаптации;	Чел.-часы	1 – 100
длительность адаптации.	Часы	1 – 100
Простота установки:		
трудоемкость инсталяции;	Чел.-часы	1 – 100
длительность инсталяции.	Часы	1 – 100
Замещаемость:		
трудоемкость замены компонентов;	Чел.-часы	1 – 100
длительность замены компонентов	Часы	1 – 100

Практичность — применимость: свойства ПС, отражающие сложность его понимания, изучения и использования, а также привлекательность для квалифицированных пользователей при применении в указанных условиях. Требования к практичности и ее субхарактеристикам — понятности и простоте использования зависят от назначения и функций ПС и могут формализоваться заказчиками набором свойств, необходимых

для обеспечения удобной и комфортной эксплуатации программ. Количественно простоту использования можно характеризовать требованиями допустимой средней длительности ввода типовых заданий и времени отклика на них. Требования к продолжительности изучения ПС, достаточной для эффективной эксплуатации системы квалифицированными специалистами, могут составлять часы или недели. Для обеспечения полноценного изучения процессов применения ПС этими специалистами необходима эксплуатационная документация, объем которой существенно зависит от назначения и функций ПС и может быть задан на основе анализа прецедентов подобных успешных проектов. Для некоторых проектов ПС, подлежащих широкому тиражированию, могут быть желательны адекватные по содержанию электронные учебники, требования к объему и функциям которых также целесообразно оценивать по прецедентам. Следует учитывать, что малый объем эксплуатационной документации может снизить качество и полноту использования функций сложного ПС, а очень большой объем — также может ухудшить эксплуатацию из-за трудности выделения из множества второстепенных деталей и освоения наиболее существенных свойств и особенностей применения ПС.

В число пользователей могут быть включены администраторы, операторы, конечные и косвенные пользователи, которые находятся под влиянием или зависят от качества функционирования ПС. В практичности следует учитывать все разнообразие характеристик внешней среды пользователей, на которые может влиять ПС, включая требующуюся подготовку к использованию и оценке результатов функционирования программ. Применимость (практичность) использования — понятие достаточно субъективное и трудно формализуемое, однако в итоге зачастую значительно определяющее функциональную пригодность и полезность применения ПС.

Понятность: свойства ПС, обеспечивающие пользователю понимание, является ли программа пригодной для его целей и как ее можно использовать для конкретных задач и условий применения. Понятность зависит от качества документации и субъективных впечатлений от функций и характеристик ПС. Ее можно описать качественно четкостью функциональной концепции, широтой демонстрационных возможностей, полнотой, комплектностью и наглядностью представления в эксплуатационной документации возможных функций и особенностей их реализации.

Она должна обеспечиваться корректностью и полнотой описания исходной и результирующей информации, а также всех деталей функций ПС для пользователей.

Простота использования: возможность пользователю удобно и комфортно эксплуатировать и управлять ПС. Аспекты изменяемости, адаптируемости и легкости инсталляции могут быть предпосылками для простоты использования и выбора конкретного ПС. Она соответствует управляемости, устойчивости к ошибкам и согласованности с ожиданиями и навыками пользователей. Эта субхарактеристика должна учитывать физические и психологические особенности пользователей и отражать уровень контролируемости и комфортности условий эксплуатации ПС, возможность предотвращения ошибок пользователей. Должны обеспечиваться простота управления функциями ПС и достаточный объем параметров управления, реализуемых по умолчанию, информативность сообщений пользователю, наглядность и унифицированность управления экраном, а также доступность изменения функций в соответствии с квалификацией пользователя и минимум операций, необходимых для запуска определенного задания и анализа результатов. Кроме того, удобство использования характеризуется рядом динамических параметров: временем ввода и отклика на задание, длительностью решения типовых задач, временем на регистрацию результатов, которые перекрываются с атрибутами субхарактеристики временная эффективность.

Простоту использования комплексов программ административных информационных систем в значительной степени характеризует корректность и адекватность описаний интерактивных директив управления, объем и время ввода заданий и время ожидания пользователями результатов при их исполнении. Простота использования может обобщенно оцениваться качественно шкалами с двумя-четырьмя категориями. Такой же метод наиболее адекватен для оценивания комфортности эксплуатации и простоты управления функциями ПС. Однако некоторые атрибуты этой субхарактеристики доступны для более полной количественной оценки путем измерения трудоемкости и длительности соответствующих процессов подготовки и обучения квалифицированных пользователей к полноценной и эффективной эксплуатации ПС.

Изучаемость: свойства ПС, обеспечивающие удобное освоение его применения достаточно квалифицированными пользователями. Она мо-

жет определяться трудоемкостью и длительностью подготовки пользователя к полноценной эксплуатации ПС. Атрибуты изучаемости зависят от возможности предварительного обучения и от совершенствования знаний в процессе эксплуатации, от возможностей оперативной помощи и подсказки при использовании ПС, а также от полноты, доступности и удобства использования руководств и инструкций по эксплуатации. Качество изучаемости ПС зависит от внутренних свойств и сложности комплекса программ, а также от субъективных характеристик квалификации конкретных пользователей.

На значения изучаемости существенно влияют демонстрационные возможности справочных средств обучения, качество и объем эксплуатационной документации, а также электронных учебников, которые можно оценивать соответственно по числу сопровождающих страниц документов или занятых учебниками килобайтов памяти на ЭВМ. Изучаемость можно отражать трудоемкостью и продолжительностью изучения пользователями соответствующей квалификации, методов и инструкций применения ПС для полноценной эксплуатации. Эти атрибуты может характеризовать трудоемкость от единиц до сотен человеко-часов и продолжительность от единиц до тысяч часов, необходимых для освоения квалифицированного применения особенно сложных комплексов программ.

Оценки ***практичности*** зависят не только от собственных характеристик ПС, но также от организации и адекватности документации процессов их эксплуатации. При этом предполагается, что в контракте, техническом задании или спецификации зафиксированы и утверждены требования к основным параметрам и качеству организационных методов и средств поддержки использования ПС. Эти требования могут влиять на функциональную пригодность и успех внедрения комплекса программ у пользователей, а также значительно различаться в зависимости от функционального назначения и сферы применения ПС. По порядковой шкале — «отлично, хорошо, удовлетворительно или неудовлетворительно» можно оценивать понятность: четкость концепции ПС, его демонстрационные возможности, наглядность и полноту документации, а также частично простоту использования: комфортность эксплуатации и простоту управления функциями.

Сопровождаемость: приспособленность ПС к модификации и изменению конфигурации. Модификации могут включать исправления, усовер-

шенствования или адаптацию ПС к изменениям во внешней среде применения, а также в требованиях и функциональных спецификациях заказчика (см. лекцию 15). Простота и трудоемкость модификаций определяется внутренними метриками качества комплекса программ, которые отражаются на внешнем качестве и качестве в использовании, а также на сложности управления конфигурациями версий ПС (см. лекцию 16).

Требования к сопровождаемости количественно можно установить для субхарактеристик изменяемости и тестируемости экономическими категориями допустимой трудоемкости и длительности реализации этих задач при некоторых условиях. Обобщенно это отражается на длительности и трудоемкости подготовки и реализации типовых изменений, обусловленных необходимостью устранения дефектов и усовершенствованиями функций ПС. Для подготовки и выполнения каждого изменения (без учета затрат времени на обнаружение и локализацию дефекта) нужно устанавливать допустимую среднюю продолжительность и суммарную трудоемкость работ специалистов при их реализации.

Анализируемость: подготовленность ПС к диагностике его дефектов или причин отказов, а также к идентификации и выделению его компонентов для модификации и исправления. Эта субхарактеристика зависит от стройности архитектуры, унифицированности интерфейсов, полноты и корректности технологической и эксплуатационной документации на ПС (см. табл. 11.3). На анализируемость влияет качество средств контроля и мониторинга изменений функциональных характеристик, а также дефектов и корректировок программ и данных.

Изменяемость: приспособленность ПС к простой реализации специфицированных изменений и к управлению конфигурацией. Реализация модификаций включает проектирование, кодирование и документирование изменений. Для этого требуются определенная трудоемкость и время, связанные с исправлением дефектов и/или модернизацией функций, а также с изменением процессов эксплуатации. При выборе атрибутов этой субхарактеристики следует учитывать влияние структуры, интерфейсов и технических особенностей ПС. Изменяемость зависит не только от внутренних свойств ПС, но также от организации и инструментальной оснащенности процессов сопровождения и конфигурационного управления, на которые ориентированы архитектура, внешние и внутренние интерфейсы программ.

Тестируемость: свойство ПС, обеспечивающее простоту проверки качества изменений и приемки модифицированных компонентов программ. Эта субхарактеристика зависит от величины области влияния изменений, которые необходимо тестировать при модификациях программ и данных, от сложности тестов для проверки их характеристик. Ее атрибуты зависят от четкости правил структурного построения компонентов и всего комплекса программ, от унификации межмодульных и внешних интерфейсов, от полноты и корректности технологической документации. В этой субхарактеристике учитываются в основном техническая и организационная составляющие процесса тестирования модификаций и не входит функциональная часть их подготовки. Обобщенно ее можно оценивать затратами труда и времени на тестирование некоторых средних по объему и сложности модификаций программ.

Субхарактеристики анализируемость и стабильность в составе ***сопровождаемости*** качественно характеризуются атрибутами, близкими к атрибутам практичности: стройностью архитектуры комплекса программ, унифицированностью интерфейсов, полнотой и корректностью документации. Для этих субхарактеристик может применяться простейшая порядковая шкала. Субхарактеристики изменяемость и тестируемость доступны количественным оценкам по величине трудоемкости и длительности реализации этих функций при типовых операциях с применением различных методов и средств автоматизации. Подготовка и каждое тестирование программы в зависимости от сложности изменения с учетом его проверки и корректировки документации может требовать трудоемкости от одного до нескольких сотен человеко-часов и времени до тысячи часов при выпуске новой версии сложного комплекса программ.

Мобильность: подготовленность ПС к переносу из одной аппаратно-операционной среды в другую (см. лекцию 15). Переносимость программ и данных на различные аппаратные и операционные платформы является важным показателем функциональной пригодности для многих современных ПС. Установление требований к мобильности ПС может быть сведено к формализации трудоемкости и длительности процессов: адаптации к новым характеристикам пользователей и внешней среды, инсталляции версий ПС в среде пользователей и замены крупных компонентов версий ПС по требованиям заказчиков или конкретных пользователей.

Наиболее простым и легко формализуемым из перечисленных процессов является инсталляция готовой версии ПС с комплектом документации на платформе пользователя без дополнительных изменений, которая может требовать нескольких часов работы специалистов. Более сложный процесс включает адаптацию ПС по формализованным инструкциям к новой специфической аппаратной, операционной или внешней среде конкретного пользователя, которая может потребовать большего времени и числа специалистов. Еще более сложный и трудоемкий процесс замены крупных компонентов ПС и перенос их на иную аппаратурную и операционную платформу.

Это свойство может оцениваться объемом, трудоемкостью и длительностью необходимых доработок компонентов и операций по адаптации, которые следует выполнить для обеспечения полноценного функционирования ПС после переноса на иную платформу (см. табл. 11.3). Мобильность может осуществляться на уровне исходных текстов программ на языке программирования или на уровне объектного кода, исполняемого ЭВМ. Она зависит от структурированности и расширяемости комплексов программ и данных, а также от наличия дополнительных ресурсов, необходимых для реализации переносимости и модификации компонентов при их переносе.

Адаптируемость: приспособленность программ и информации баз данных к модификации для эксплуатации в различных аппаратных и операционных средах без применения других действий или средств, чем те, что предназначены для этой цели при первичной разработке в исходной версии ПС. Она зависит от свойств и структуры аппаратной и операционной среды, от методов и средств, заложенных в ПС для подготовки к переносу на новые платформы. Адаптируемость включает масштабируемость внутренних возможностей (например, экранных полей, размеров таблиц, объемов транзакций, форматов отчетов). Если ПС должно адаптироваться конечным пользователем, адаптируемость соответствует пригодности для индивидуализации комплекса программ при изменениях внешней среды и может быть компонентом простоты использования.

Простота установки — инсталляции: способность ПС к простому внедрению (инсталляции) в новой аппаратной и операционной среде заказчика или пользователя. Если ПС должно устанавливаться конечным пользователем, легкость установки будет предпосылкой для удобства ис-

пользования. Так же, как и адаптируемость, она может измеряться трудоемкостью и длительностью процедур установки, а также степенью удовлетворения требований заказчика и пользователей к характеристикам и сложности инсталляции.

Замещаемость: приспособленность каждого компонента ПС к относительно простому использованию вместо другого выделенного и указанного заменяемого компонента. Она может включать атрибуты как простоты установки, так и адаптируемости. Большую роль для этого свойства играют четкая структурированность архитектуры и стандартизация внутренних и внешних интерфейсов ПС. Это свойство отражается на трудоемкости и длительности замены в основном крупных компонентов ПС.

Меры и шкалы ***мобильности*** в некоторой степени подобны качественным и количественным мерам и шкалам сопровождаемости. Компоненты мобильности: адаптируемость, простота установки и замещаемость доступны количественным технико-экономическим оценкам. При выборе характеристик ПС наиболее жесткие требования обычно предъявляются к трудоемкости и длительности инсталляции версий ПС на новой платформе, которые могут занимать от нескольких минут до нескольких десятков часов и требовать соответствующей трудоемкости до десятков человекочасов. Большей потребностью времени и трудоемкости обычно характеризуются адаптация версий ПС к условиям новой внешней среды и к требованиям пользователей, а также замена и ввод крупных компонентов в новую программно-аппаратную среду. Интегрально мобильность оказывает влияние на функциональную пригодность при переносе программ и данных на иные операционные и аппаратные платформы, при расширении и изменении их функций. Для этого реализация основных функций комплекса программ должна быть подготовлена к мобильности, для чего требуются дополнительные трудовые, временные и вычислительные ресурсы. Отсутствие такой подготовки при проектировании ПС отражается на возрастании затрат на процедуры, входящие в мобильность и для некоторых типов ПС могут ограничивать их функциональную пригодность.

11.4. Характеристики качества баз данных

Современные базы данных являются одними из массовых специфических объектов в сфере информатизации, для которых в ряде областей

необходимо особенно высокое качество и его квалифицированное системное проектирование. Базу данных можно рассматривать как ***два компонента***:

— ***программные средства системы управления базой данных*** (СУБД), независимые от сферы их применения, структуры и смыслового содержания накапливаемых и обрабатываемых данных;

— ***информацию базы данных*** (ИБД), доступную для накопления, упорядочивания, обработки и использования в конкретной проблемно-ориентированной сфере применения.

При этом одна и та же система управления базой данных (СУБД) может обрабатывать различные по структуре, составу и содержанию данные, а одни и те же данные могут управляться программными средствами различных СУБД. Хотя эти компоненты тесно взаимодействуют при реализации конкретной прикладной БД, первоначально при проектировании они создаются или выбираются практически независимо и могут рассматриваться в их ЖЦ как ***два объекта***, которые различаются:

— номенклатурой и содержанием показателей качества, определяющих их назначение, функции и потребительские свойства;

— технологией и средствами автоматизации разработки и обеспечения всего ЖЦ каждого объекта;

— категориями специалистов, обеспечивающих: создание, эксплуатацию или применение компонентов БД;

— комплектами эксплуатационной и технологической документации, поддерживающими жизненный цикл объектов.

Первым компонентом для системного анализа и требований к качеству является комплекс программ СУБД. Практически весь набор характеристик и атрибутов качества ПС, изложенный в стандарте **ISO 9126**, в той или иной степени может использоваться при формировании требований к качеству СУБД. Во всех случаях важнейшими характеристиками качества СУБД являются требования к функциональной пригодности для процессов формирования и изменения информационного наполнения БД администраторами, а также доступа к данным и представления результатов пользователям БД. Ниже за основу принята номенклатура и содержание стандартизированных характеристик сложных комплексов программ, которые адаптируются применительно к понятиям и особенностям компо-

нентов баз данных. В зависимости от конкретной проблемно-ориентированной области применения СУБД приоритет при системном анализе требований к качеству может отдаваться различным конструктивным характеристикам: либо надежности и защищенности применения (финансовая сфера), либо удобству использования малоквалифицированными пользователями (социальная сфера), либо эффективности использования ресурсов (сфера материально-технического снабжения). Однако практически во всех случаях сохраняется некоторая роль ряда других конструктивных показателей качества.

Вторым компонентом БД является собственно накапливаемая и обрабатываемая информация. В системах баз данных доминирующее значение приобретают сами данные, их хранение и обработка. Ниже сделан акцент на системный анализ требований и составляющих характеристик качества этого объекта — на ***информацию баз данных (ИБД)*** с предположением, что средства СУБД способны их обеспечить. Для оценивания качества информации БД может сохраняться общий, методический подход к выделению адекватной номенклатуры стандартизированных в **ISO 9126** базовых характеристик и субхарактеристик качества ПС. Выделяемые показатели качества должны иметь практический интерес для пользователей БД и быть упорядочены в соответствии с приоритетами практического применения. Кроме того, каждый выделяемый показатель качества ИБД должен быть пригоден для достаточно достоверного оценивания или измерения, а также для сравнения с требуемым значением при испытаниях заказчиком.

При проектировании каждой БД в контракте, техническом задании и в спецификации должны селектироваться и формализоваться представительный набор функциональных требований к качеству ИБД, адекватный ее назначению и области применения, а также требованиям заказчика и потенциальных пользователей. Так же как для ПС, характеристики качества ИБД можно разделить на ***функциональные и конструктивные***. Их номенклатура, содержание и субхарактеристики ниже базируются на описаниях, рекомендуемых стандартом **ISO 9126**. Они представляются достаточно универсальными и применимыми для ***систематизации характеристик качества информации баз данных***. Однако номенклатура показателей качества не всегда может ограничиваться только характеристиками

информации в БД, а должна включать ряд уточнений, отражающих комплексную эффективность и функциональную пригодность совместного применения СУБД и ИБД пользователями в реальных условиях.

Функциональная пригодность ИБД может представлять сложную проблему для определения соответствия требований реальным значениям необходимых атрибутов качества, особенно для больших распределенных БД при использовании разнообразной и сложной информации об анализируемых объектах. Мерой качества функциональной пригодности может быть ***степень покрытия целей, назначения и функций БД***, доступной пользователям информацией. Так же как для ПС, для баз данных в составе функциональной пригодности целесообразно использовать группу субхарактеристик, определяющих функциональные и структурные требования к базам данных. Дополнительно функциональная пригодность многих ИБД может отражаться:

— полнотой накопленных описаний объектов — относительным числом объектов или документов, имеющихся в БД, к общему числу объектов по данной тематике или по отношению к числу объектов в аналогичных БД того же назначения;

— идентичностью данных — относительным числом описаний объектов, не содержащих дефекты и ошибки, к общему числу документов об объектах в ИБД;

— актуальностью данных — относительным числом устаревших данных об объектах в ИБД к общему числу накопленных и обрабатываемых данных.

К **конструктивным характеристикам качества информации БД** в целом можно отнести, с некоторым уточнением понятий, субхарактеристик и атрибутов, практически все стандартизированные показатели качества ПС, которые представлены в **ISO 9126**. Требования к информации баз данных также должны содержать обеспечение ее надежности, эффективности использования ресурсов ЭВМ, практичности — применимости, сопровождаемости и мобильности. Содержание и атрибуты этих конструктивных характеристик в данном случае несколько отличаются от применяемых для программ, однако их сущность, как базовых понятий и характеристик качества объектов, целесообразно использовать при проектировании для систематизации и регламентированного формирования требований к этим

компонентам систем. Меры и шкалы для оценивания конструктивных характеристик, в значительной степени могут применяться те же, что при анализе качества программных средств.

Корректность или достоверность данных — это степень соответствия информации об объектах в БД, реальным объектам вне ЭВМ в данный момент времени, определяющаяся изменениями самих объектов, некорректностями записей о их состоянии или некорректностями расчетов их характеристик. При системном проектировании выбор и установление требований к корректности данных в БД можно оценивать по ***степени покрытия накопленными, актуальными и достоверными данными*** состояния и изменения внешних объектов, которые они отражают (см. табл. 11.1). Кроме того, к корректности БД можно отнести некоторые объемно-временные характеристики сохраняемых и обрабатываемых данных:

— объем базы данных — относительное число записей описаний объектов или документов в базе данных, доступных для хранения и обработки, по сравнению с полным числом реальных объектов во внешней среде;

— оперативность — степень соответствия динамики изменения описаний данных в процессе сбора и обработки, состояниям реальных объектов, или величина допустимого запаздывания между появлением или изменением характеристик реального объекта, относительно его отражения в базе данных;

— глубина ретроспективы — максимальный интервал времени от даты выпуска и/или записи в базу данных самого раннего документа до настоящего времени;

— динамичность — относительное число изменяемых описаний объектов к общему числу записей в БД за некоторый интервал времени, определяемый периодичностью издания версий БД.

Защищенность информации БД реализуется в основном программными средствами СУБД, однако в сочетании с поддерживающими их средствами организации и защиты данных. Цели, назначение и функции защиты тесно связаны с особенностями функциональной пригодности каждой ИБД. При проектировании свойства защищать информацию баз данных от негативных воздействий описываются обычно составом и номенклатурой методов и средств, используемых для защиты от внешних и внутренних угроз.

Надежность информации баз данных может основываться на применении понятий и методов теории надежности, которая позволяет получить ряд четких, измеряемых интегральных показателей их качества. Надежная ИБД, прежде всего, должна обеспечивать достаточно низкую вероятность потери работоспособности — отказа, в процессе ее функционирования в реальном времени. Быстрое реагирование на потерю или искажение данных и восстановление их достоверности и работоспособности за время меньшее, чем порог между сбоем и отказом, обеспечивают высокую надежность БД. Если в этих ситуациях происходит достаточно быстрое восстановление, такое, что не фиксируется отказ, то такие события не влияют на основные показатели надежности — наработку на отказ и коэффициент готовности ИБД. Непредсказуемость вида, места и времени проявления дефектов ИБД в процессе эксплуатации приводит к необходимости создания специальных, дополнительных систем оперативной защиты от непредумышленных, случайных искажений данных. Надежность должна повышаться за счет средств обеспечения помехоустойчивости, оперативного контроля и восстановления ИБД.

Стандартом **ISO 9126** рекомендуется анализировать и учитывать надежность комплексов программ четырьмя субхарактеристиками, которые могут быть применены также для формирования требований к характеристикам качества информации БД. ***Завершенность*** — свойство ИБД, состоящее в способности не попадать в состояния отказов вследствие потерь, искажений, ошибок и дефектов в данных. ***Устойчивость*** к дефектам и ошибкам — свойство ИБД автоматически поддерживать заданный уровень качества данных в случаях проявления дефектов и ошибок или нарушения установленного интерфейса по данным с внешней средой. Для этого в ИБД рекомендуется вводить оперативное обнаружение дефектов и ошибок информации, их идентификацию и автоматическое восстановление (рестарт) нормального функционирования ИБД.

Восстанавливаемость — свойство ИБД в случае отказа возобновлять требуемый уровень качества информации, а также корректировать поврежденные данные. Для этого необходимы вычислительные ресурсы и время на выявление неработоспособного состояния, диагностику причин отказа, а также на реализацию процессов восстановления. ***Доступность или готовность*** — свойство ИБД быть в состоянии полностью выпол-

нять требуемую функцию в данный момент времени при заданных условиях использования информации базы данных. Обобщение характеристик отказов и восстановления производится в критерии коэффициент готовности ИБД. Этот показатель отражает вероятность иметь восстанавливаемые данные в работоспособном состоянии в произвольный момент времени.

Эффективность использования ресурсов ЭВМ при анализе реального функционирования БД отражается временными характеристиками взаимодействия конечных пользователей и администраторов ИБД в процессе эксплуатации базы данных по прямому назначению. ***Временная эффективность*** БД определяется длительностью выполнения заданных функций и ожидания результатов от ИБД в средних и/или наихудших случаях, с учетом приоритетов задач. Она зависит от объема, структуры и скорости обработки данных, влияющих непосредственно на интервал времени завершения конкретного вычислительного процесса, и от пропускной способности — производительности, т.е. от числа заданий, которое можно реализовать на данной ЭВМ в заданном интервале времени (см. табл. 11.2).

Используемость ресурсов или ресурсная экономичность в стандартах отражается занятостью ресурсов центрального процессора, оперативной, внешней и виртуальной памяти, каналов ввода-вывода, терминалов и каналов сетей связи. Эта величина определяется структурой, функциями и объемом ИБД, а также архитектурными особенностями и доступными ресурсами ЭВМ. В зависимости от конкретных задач и особенностей ИБД и ЭВМ при проектировании и выборе атрибутов качества ИБД может доминировать либо абсолютная величина занятости ресурсов различных видов, либо относительная величина использования ресурсов каждого вида при нормальном функционировании ИБД.

Практичность — применимость — зачастую значительно определяет функциональную пригодность и полезность применения ИБД для квалифицированных пользователей. В число пользователей могут быть включены администраторы, конечные и косвенные пользователи, которые находятся под влиянием или зависят от качества информации БД. В эту группу показателей качества входят субхарактеристики и атрибуты, с различных сторон отражающие функциональную понятность, удобство освоения, системную эффективность и простоту использования данных. Некоторые субхарактеристики можно оценивать экономическими показателя-

ми — затратами труда и времени специалистов на реализацию некоторых функций взаимодействия с данными (см. табл. 11.3).

Понятность зависит от качества документации и субъективных впечатлений потенциальных пользователей от функций и характеристик ИБД. В проекте ее можно представить качественно четкостью функциональной концепции, широтой демонстрационных возможностей, полнотой, комплектностью и наглядностью представления в эксплуатационной документации возможных функций и особенностей реализации данных в БД. Она должна обеспечиваться корректностью и полнотой описания исходной и результирующей информации, а также всех деталей применения ИБД для пользователей.

Простота использования ИБД — возможность удобно и комфортно ее эксплуатировать и управлять данными. Для этого должны быть обеспечены: достаточный объем параметров управления, реализуемых по умолчанию, информативность сообщений пользователям, наглядность и унифицированность управления экраном, а также доступность изменения функций ИБД в соответствии с квалификацией пользователей и минимум операций, необходимых для реализации определенного задания и анализа результатов. Некоторые атрибуты этой субхарактеристики доступны при установлении количественных требований путем указания трудоемкости и длительности соответствующих процессов подготовки и обучения квалифицированных пользователей к эффективной эксплуатации ИБД.

Изучаемость может определяться требованиями затрат трудоемкости и длительности подготовки пользователя к полноценной эксплуатации информации БД. Изучаемость ИБД зависит от внутренних свойств и сложности структуры информации БД, а также от субъективных характеристик квалификации конкретных пользователей. Она может также характеризоваться объемом эксплуатационной документации и/или объемом и качеством электронных учебников.

Сопровождаемость информации БД в проекте может отражаться удобством и эффективностью исправления, усовершенствования или адаптации структуры и содержания описаний данных в зависимости от изменений во внешней среде применения, а также в требованиях и функциональных спецификациях заказчика. Обобщенно качество сопровождаемости ИБД можно представить потребностью трудовых и временных ресурсов

для ее обеспечения и для реализации. Возможные затраты экономических, трудовых и временных ресурсов на развитие и совершенствование качества ИБД зависят не только от внутренних свойств данных, но также от запросов и потребностей пользователей на новые функции и от готовности заказчика и разработчика удовлетворить эти потребности. По объему предполагаемых изменений, а также вновь вводимых в очередную версию данных с учетом сложности и новизны их разработки могут быть сформулированы требования на их реализацию.

Совокупность субхарактеристик сопровождаемости ПС, представленная в стандарте **ISO 9126**, вполне применима для описания требований к этому показателю качества информации БД, в основном теми же организационно-технологическими субхарактеристиками. ***Анализируемость*** ИБД зависит от стройности архитектуры, унифицированности интерфейсов, полноты и корректности технологической и эксплуатационной документации на БД. ***Изменяемость*** состоит в приспособленности структуры и содержания данных к реализации специфицированных изменений и расширений и к управлению конфигурацией данных. Изменяемость зависит не только от внутренних свойств ИБД, но также от организации и инструментальной оснащенности процессов сопровождения и конфигурационного управления, на которые ориентированы в проекте архитектура, внешние и внутренние интерфейсы данных.

Тестируемость зависит от величины области влияния изменений, которые необходимо тестировать при модификациях структуры и содержания данных в ИБД, от сложности тестов для проверки их характеристик. Ее атрибуты зависят от четкости формализации в системном проекте правил структурного построения компонентов и всего комплекса ИБД, от унификации межмодульных и внешних интерфейсов, от полноты и корректности технологической документации. Субхарактеристики изменяемость и тестируемость данных доступны количественному определению по величине трудоемкости и длительности реализации этих функций при типовых операциях с данными при применении различных методов и средств автоматизации.

Мобильность данных БД, так же как для программ, можно характеризовать в основном длительностью и трудоемкостью их ***инсталляции, адаптации и замещаемости*** при переносе ИБД на иные аппаратные и

операционные платформы. Информация о процессах, происходящих во внешней среде, может иметь большие объем и трудоемкость первичного накопления и актуализации, что определяет необходимость ее тщательного хранения и регламентированного изменения. Так как перенос БД часто обусловлен необходимостью увеличения ресурсов ЭВМ, доступных для решения новых перспективных задач, их проект становится естественным расширением функций ИБД относительно исходной версии проекта. Для оценки качества и определения требований к мобильности ИБД, так же как для ПС, следует решать задачу сравнения достигаемого эффекта и затрат для методов переноса или повторной разработки компонентов и наполнения базы данных в конкретных условиях с учетом всех перечисленных факторов и затрат.

11.5. Характеристики защиты и безопасности функционирования программных средств

Непрерывно возрастающая сложность и вследствие этого уязвимость систем и программных продуктов от случайных и предумышленных негативных воздействий выдвинули ряд проблем, связанных с безопасностью систем и программных средств, в разряд важнейших — ***стратегических***, определяющих принципиальную возможность и эффективность применения программных продуктов в административных системах, в промышленности и в военной технике. При этом выделились области анализа и обеспечения: ***информационной безопасности***, связанные в основном с защитой от предумышленных, негативных воздействий на информационные ресурсы систем, и ***функциональной безопасности***, обусловленной отказовыми ситуациями и потерей работоспособности систем и ПС вследствие проявления непредумышленных, случайных дефектов и отказов программ, данных, аппаратуры и внешней среды. С позиции доминирующей категории обеспечения безопасности автоматизированные системы, их программные продукты и базы данных можно условно разделить на ***два крупных класса***:

— системы, в которых накапливаются, обрабатываются и хранятся большие объемы информации из внешней среды с активным участием пользователей, для которой должна обеспечиваться конфиденциальность,

целостность и доступность данных потребителям, что отражается требованиями преимущественно к характеристикам ***информационной безопасности***;

— системы и объекты автоматизации, в аппаратуру которых ***встроены комплексы программ*** управления и обработки информации в реальном времени, основные задачи которых состоят в обеспечении достоверной реализации эффективного и устойчивого управления объектами внешней среды при относительно малом (негативном) участии пользователей в их решении и высоких требованиях к характеристикам ***функциональной безопасности***.

В ряде случаев эти два ***понятия и их характеристики близки*** и связаны с нарушением выполнения требований спецификаций к функциональной пригодности объекта или системы, однако они имеют существенные особенности, которые целесообразно уточнить.

Обеспечение информационной безопасности функционирования систем в процессе разработки и эксплуатации развивается вследствие возрастания сложности и ответственности задач использования информационных ресурсов и увеличения их уязвимости от предумышленных, внешних воздействий с целью незаконного использования или искажения информации и программ, которые по своему содержанию предназначены для применения ограниченным кругом лиц. Основное внимание в современной теории и практике обеспечения безопасности информационных систем сосредоточено на защите от злоумышленных разрушений, искажений, хищений и нерегламентированного использования программных средств и информационных ресурсов баз данных. Для решения этой проблемы созданы и активно развиваются методы, средства и стандарты обеспечения информационной безопасности — защиты программ и данных от ***предумышленных негативных внешних воздействий***. При этом понятия обеспечения безопасности и защиты системы и информации зачастую не разделяются. Факторы безопасности, характерные для сложных информационных систем — целостность, доступность и конфиденциальность информационных ресурсов, а также ряд типовых процедур систем защиты — криптографическая поддержка, идентификация и аутентификация, защита и сохранность данных пользователей при предумышленных негативных воздействиях из внешней среды, ***далее не рассматриваются и не учитываются***.

Обеспечение функциональной безопасности при случайных, дестабилизирующих воздействиях и отсутствии злоумышленного влияния на системы, ПС или информацию баз данных существенно отличается от задач информационной безопасности (рис. 11.1). Функциональная безопасность объектов и систем зависит от отказовых ситуаций, негативно отражающихся на работоспособности и реализации их основных функций, причинами которых могут быть дефекты и аномалии в аппаратуре, программах, данных или вычислительных процессах. При этом катастрофически, критически или существенно искажается процесс функционирования систем, что наносит значительный ущерб при их применении. Основными источниками отказовых ситуаций могут быть некорректные исходные требования заказчика, сбои и отказы в аппаратуре, дефекты или ошибки в программах и данных функциональных задач, проявляющиеся при их исполнении в соответствии с назначением. При таких воздействиях внешняя, функциональная работоспособность систем может разрушаться не полностью, однако невозможно полноценное выполнение заданных функций и требований к качеству информации для потребителей. Безопасность их функционирования определяется проявлениями ***дестабилизирующих факторов, приносящих большой ущерб***:

— техническими отказами внешней аппаратуры и искажениями исходной информации от объектов внешней среды и от пользователей систем и обработанной информации;

— случайными сбоями и физическими разрушениями элементов и компонентов аппаратных средств вычислительных комплексов и средств телекоммуникации;

— дефектами и ошибками в комплексах программ обработки информации и в данных;

— пробелами и недостатками в средствах обнаружения опасных отказов и оперативного восстановления работоспособного состояния систем, программ и данных.

При анализе характеристик функциональной безопасности целесообразно выделять два класса систем и их ПС. ***Первый класс*** составляют системы, имеющие встроенные комплексы программ жесткого регламента реального времени, автоматизированно управляющие динамическими внешними объектами или процессами. Время необходимой реакции на

отказовые ситуации таких систем обычно исчисляется секундами или долями секунды, и процессы восстановления работоспособности должны проводиться за это время в достаточной степени автоматизированно (бортовые системы в авиации, на транспорте, в некоторых средствах вооружения, системы управления атомными электростанциями). Эти системы используют относительно небольшие информационные ресурсы, сложные логические комплексы программ управления и практически недоступны для предумышленных негативных внешних воздействий.

Системы ***второго класса*** применяются для управления процессами и обработки деловой информации из внешней среды, в которых активно участвуют специалисты-операторы (банковские, административные, штабные военные системы). Допустимое время реакции на опасные отказы в этих системах может составлять десятки секунд и минуты, и операции по восстановлению работоспособности частично могут быть доверены специалистам-администраторам по обеспечению функциональной безопасности. В этих системах возможны предумышленные негативные внешние воздействия, однако они ниже не рассматриваются.

Понятия и характеристики функциональной безопасности систем близки к понятиям надежности (см. выше п. 11.3). Основное различие состоит в том, что в показателях надежности учитываются все реализации опасных отказов, а в характеристиках функциональной безопасности следует регистрировать и учитывать только те отказы, которые привели к столь большому, катастрофическому ущербу, что отразились на безопасности системы и информации для потребителей. Статистически таких отказов может быть в несколько раз меньше, чем учитываемых в значениях надежности. Однако методы, влияющие факторы и реальные значения характеристик надежности ПС могут служить ориентирами при оценке функциональной безопасности критических систем. Поэтому способы оценки характеристик и испытаний функциональной безопасности могут ***базироваться на методах определения надежности*** функционирования комплексов программ и баз данных.

Ущерб от дефектов и ошибок программ и данных может проявляться в более или менее систематических отказах, каждый из которых отражается на надежности, но не является катастрофой с большим ущербом, влияющим на безопасность системы. Накопление таких отказов со временем

может приводить к последствиям, нарушающим функциональную безопасность систем и их применение. Таким образом, дополнительно сближаются понятия и характеристики надежности и функциональной безопасности сложных систем и ПС.

Эффективная система защиты информации и программных средств подразумевает наличие совокупности организационных и технических мероприятий, направленных на предупреждение различных угроз безопасности, их выявление, локализацию и ликвидацию. Создание такой системы предусматривает ***планирование и реализацию целенаправленной политики комплексного обеспечения безопасности систем и программных продуктов*** (см. рис. 11.1). Требования к характеристикам программных средств, обеспечивающим безопасность, обычно представляются в составе общей спецификации требований к характеристикам системы.

Наиболее полно ***степень защиты системы характеризуется величиной предотвращенного ущерба — риска*** (см. лекцию 10), возможного при проявлении дестабилизирующих факторов и реализации конкретных угроз безопасности применению программного продукта пользователями, а также средним временем между возможными проявлениями угроз, нарушающих безопасность. С этой позиции затраты ресурсов разработчиками и заказчиками на обеспечение безопасности функционирования системы должны быть соизмеримыми с возможным средним ущербом у пользователей от нарушения безопасности. Проектирование защиты систем с использованием программных средств включает подготовку комплекса взаимосвязанных мер, направленных на достижение требуемых характеристик и уровня безопасности. Для обеспечения эффективности систем ***комплекс программ обеспечения безопасности целесообразно базировать на следующих общих принципах:***

— стоимость создания и эксплуатации системы программной защиты и обеспечения безопасности должна быть меньше, чем размеры наиболее вероятного или возможного (в среднем), неприемлемого потребителями системы риска-ущерба, от любых потенциальных угроз;

— программная защита функциональных программ и данных должна быть комплексной и многоуровневой, ориентированной на все виды угроз с учетом их опасности для потребителя;

— комплекс программ защиты должен иметь целевые, индивидуальные компоненты, предназначенные для обеспечения безопасности функ-

ционирования каждого отдельно взятого объекта и функциональной задачи ПС с учетом их уязвимости и степени влияния на безопасность системы в целом;

— система программ защиты не должна приводить к ощутимым трудностям, помехам и снижению эффективности применения и решения основных, функциональных задач пользователями в целом.

Процессы проектирования программ обеспечения безопасности ПС, как самостоятельной системы, принципиально не отличаются от технологии проектирования любых других сложных программных комплексов. Для этого, прежде всего, необходимо проанализировать и конкретизировать в ***спецификации требований проекта ПС задачи, а также исходные данные и факторы, определяющие характеристики безопасности функционирования программ:***

— критерии качества и значения характеристик, отражающих необходимый и достаточный уровень безопасности применения системы пользователями в целом, и каждого из ее основных, функциональных компонентов в соответствии с условиями среды применения и требованиями спецификаций заказчика;

— перечень и характеристики возможных внутренних и внешних дестабилизирующих факторов и угроз, способных влиять на характеристики безопасности функционирования программных средств и баз данных;

— требования к методам и средствам предотвращения и снижения влияния угроз безопасности, обусловленные предумышленными негативными внешними воздействиями, а также возможными дефектами программ и данных;

— перечень подлежащих решению задач защиты, перекрывающих все потенциально возможные угрозы, и оценки характеристик решения отдельных задач, необходимых для обеспечения равнопрочной безопасности системы с заданной эффективностью;

— оперативные методы и средства повышения характеристик безопасности функционирования программ в течение всего жизненного цикла системы путем введения в комплекс программ временной, программной и информационной избыточности для реализации системы защиты от актуальных видов угроз;

— ресурсы, необходимые и доступные для разработки и размещения программной системы обеспечения безопасности (финансово-экономичес-

кие, ограниченная квалификация специалистов и вычислительные ресурсы ЭВМ);

— стандарты, нормативные документы и методики воспроизводимых измерений характеристик безопасности, а также состав и значения исходных и результирующих данных, обязательных для проведения испытаний;

— оценки комплексной эффективности защиты системы и программного продукта и их сравнение с требуемой заказчиком, с учетом реальных ограничений совокупных затрат ресурсов на обеспечение защиты.

В основу формирования требований по безопасности должно быть положено определение перечня и характеристик потенциальных угроз безопасности и установление возможных источников их возникновения (см. рис. 11.1).

Внешними дестабилизирующими факторами, создающими угрозы безопасности функционирования программных продуктов и системы, являются:

— предумышленные, негативные воздействия лиц с целью искажения, уничтожения или хищения программ, данных и документов информационной системы;

— ошибки и несанкционированные воздействия оперативного, административного и обслуживающего персонала в процессе эксплуатации системы;

— искажения в каналах телекоммуникации информации, поступающей от внешних источников и передаваемой потребителям, а также недопустимые значения и изменения характеристик потоков информации от объектов внешней среды;

— сбои и отказы в аппаратуре вычислительных средств;

— вирусы, распространяемые по каналам телекоммуникации;

— изменения состава и конфигурации комплекса взаимодействующей аппаратуры системы за пределы, проверенные при испытаниях или сертификации.

Внутренними источниками угроз безопасности функционирования сложных систем и ПС являются (см. лекцию 10):

— системные ошибки при постановке целей и задач проектирования системы, формулировке требований к функциям и характеристикам средств защиты решения задач, определении условий и параметров внешней среды, в которой предстоит применять программный продукт;

— алгоритмические ошибки проектирования при непосредственной алгоритмизации функций защиты программных средств и баз данных, при определении структуры и взаимодействия компонентов комплексов программ, а также при использовании информации баз данных;

— ошибки программирования в текстах программ и описаниях данных, а также в исходной и результирующей документации на компоненты ПС;

— недостаточная эффективность используемых методов и средств оперативной защиты программ и данных и обеспечения безопасности функционирования системы в условиях случайных и предумышленных негативных воздействий от внешней среды.

Полное устранение перечисленных ***угроз характеристикам безопасности*** функционирования критических ПС принципиально невозможно. При проектировании проблема состоит в выявлении факторов, от которых они зависят, в создании методов и средств уменьшения их влияния на безопасность ПС, а также ***в эффективном распределении ресурсов на средства защиты***. Необходимо ***оценивать уязвимость*** функциональных компонентов системы для различных предумышленных, негативных воздействий и степень их влияния на основные характеристики безопасности. В зависимости от этого следует распределять ресурсы средств защиты для создания проекта системы, равнопрочной по безопасности функционирования при любых внешних воздействиях.

Величина и рациональное распределение ресурсов ЭВМ на отдельные виды защиты оказывает значительное влияние на достигаемую комплексную безопасность системы. Наиболее общим видом ресурсов, который приходится учитывать при проектировании, являются ***допустимые финансово-экономические затраты или сметная стоимость разработки и функционирования системы обеспечения безопасности и средств программной защиты***. Для размещения средств защиты в объектной ЭВМ при проектировании должна быть предусмотрена программная и информационная избыточность в виде ресурсов внешней и внутренней памяти ЭВМ. Кроме того, для функционирования средств защиты необходима временная избыточность — дополнительная производительность ЭВМ.

При проектировании целесообразно разделять вычислительные ресурсы, необходимые для непосредственного решения основных, функциональных задач системы, и ресурсы, требующиеся для защиты и обеспече-

ния корректного, безопасного функционирования программного продукта. Соотношение между этими видами ресурсов в реальных крупномасштабных системах зависит от сложности и состава решаемых функциональных задач, степени их критичности и требований к характеристикам безопасности всей системы. В различных классах систем ресурсы на обеспечение безопасности могут составлять от 5—20% до 100—300% от ресурсов, используемых на решение основных, функциональных задач, т.е. в особых случаях (критические военные системы) могут превышать последние в 2—4 раза. В административных и организационных системах средства обеспечения безопасности обычно используют 10—20% всех видов трудовых, аппаратных и вычислительных ресурсов.

Одна из трудностей планирования процессов для достижения высокого качества защиты состоит обычно в отсутствии полной совокупности достоверных требований заказчика к характеристикам безопасности на начальных этапах проектирования и разработки, а также итерационный процесс их конкретизации в течение всего жизненного цикла ПС. В результате первично сформулированные ***требования к характеристикам качества системы защиты и обеспечения безопасности крупных ПС последовательно уточняются*** и корректируются в процессе взаимодействия заказчика и разработчика с учетом объективно изменяющихся характеристик развивающегося проекта.

Проектирование системы защиты тесно связано с определением понятия и функций администратора безопасности системы. ***Администратор безопасности*** — субъект доступа, ответственный за защиту охраняемых ресурсов и эффективное использование имеющихся функций защиты системы пользователями. Без постоянного присутствия администратора при применении крупных систем меры защиты могут быть неэффективными, так как злоумышленник получает возможность в течение неограниченного времени осуществлять попытки несанкционированного доступа. Поэтому в системы обеспечения безопасности вводятся:

— административные функции и интерфейсы, доступные администратору по безопасности;

— принципы и средства для последовательного, эффективного использования и адаптации функций компонентов системы безопасности;

— средства конфигурирования функций системы и комплекса обеспечения безопасности;

— контроль допустимого поведения пользователей и предотвращение нештатного применения процедур, влияющих на безопасность.

В системах с большим количеством объектов, требующих разных уровней защиты, может быть несколько администраторов, объединенных в ***службу администрации безопасности***. Важным свойством системы управления доступом должна являться способность создавать так называемый ***след контроля***, т.е. совокупность сведений о состоянии и функционировании средств защиты, накапливаемых во времени и предназначенных для анализа и управления средствами защиты. Для хранения этих сведений у администраторов обычно организуются контрольные журналы учета и регистрации событий защиты. Основными сведениями, накапливаемыми в этих журналах, являются данные о работе пользователей и попытках несанкционированных действий, выходящих за рамки представленных им полномочий, или от объектов внешней среды.

Чтобы гарантии безопасности достигались при минимальных затратах, необходимы целенаправленное, координируемое планирование и управление для предотвращения дефектов и ошибок проектирования, а также для их выявления и устранения на самых ранних этапах разработки. Поэтому план и мероприятия, обеспечивающие качество программ защиты, должны охватывать не только завершающие испытания, а весь жизненный цикл программ обеспечения безопасности. Для этого в процессе формирования технического задания следует сформулировать ***основные положения методологии и план последовательного повышения характеристик безопасности*** путем наращивания комплекса средств защиты, поэтапных испытаний компонентов и определения характеристик безопасности, допустимых для продолжения работ на следующих этапах.

Проекты комплексов защиты зависят от конкретных характеристик и назначения объектов, подлежащих защите, а также от применяемых нормативных документов и их требований. Проектирование средств обеспечения безопасности функционирования ПС — творческий процесс, зависящий от множества факторов, что определяет ***ограниченную стандартизацию совокупности ряда методов и задач***. Наиболее широко и детально методологические и системные задачи проектирования комплексной защиты систем изложены в ***трех частях стандарта*** **ISO 15408:1-3:1999** — Методы и средства обеспечения безопасности. Критерии оценки безопас-

ности информационных технологий. В ***первой***, относительно небольшой части представлены цели и концепция обеспечения безопасности, а также общая модель построения защиты, которая отличается гибкостью и динамичностью формирования требований и оценивания функций и компонентов системы безопасности. В ней выделены: окружающая среда; объекты защиты; требования и спецификации функций защиты; задачи инструментальных средств обеспечения системы защиты. Изложены общие требования к критериям и характеристикам оценки результатов защиты, к Профилю по безопасности, к целям оценки требований и к использованию их результатов. Предложен проект комплекса общих целей, задач и критериев обеспечения безопасности конкретных систем.

В наибольшей, ***второй*** части стандарта представлена парадигма построения и реализации, структурированных и детализированных функциональных требований к компонентам защиты систем. Выделены и классифицированы одиннадцать базовых ***классов*** требований обеспечения безопасности систем. Каждый класс детализирован функциональным ***семейством*** требований, которые реализуют соответствующую часть целей обеспечения безопасности и, в свою очередь, структурированы наборами требований к более мелким ***компонентам*** частных задач.

Профили семейств и компонентов служат базой для дальнейшей конкретизации функциональных требований в ***Задании по безопасности*** для определенного проекта системы и помогают избегать грубых ошибок и пробелов при формировании набора таких требований. Обобщения оценок спецификации требований Задания по безопасности должны обеспечивать возможность делать общий вывод заказчиками, разработчиками и испытателями проекта об уровне соответствия безопасности функциональным требованиям и требованиям гарантированности защиты. ***Профиль и Задание по безопасности*** являются основными исходными документами при сертификации на соответствие требованиям заказчика к характеристикам безопасности применения конкретной системы.

Третья часть стандарта посвящена целям, методам и уровням обеспечения гарантий качества систем защиты, при разработке и реализации требований к функциям обеспечения безопасности в системе. Определены методы и средства, которые целесообразно использовать для обеспечения корректной реализации Задания по безопасности, жизненного цикла средств

защиты и эффективного их применения. Изложены детальные требования по обеспечению гарантии качества создания и применения систем безопасности.

Таким образом, методологически решение задач обеспечения характеристик безопасности должно осуществляться как ***проектирование сложной, достаточно автономной программно-аппаратной системы*** в окружении и взаимодействии с основными, функциональными задачами и компонентами системы (см. рис. 11.1). Защита должна быть ориентирована на комплексное обеспечение эффективного решения основных, функциональных задач безопасности всей системы. При этом следует определять приоритеты и ранжировать по степени необходимой защиты функциональные компоненты, оценивать опасность различных внешних и внутренних угроз безопасности, выделять методы, средства и нормативные документы, адекватные видам угроз и требуемой защите, оценивать необходимые и доступные для этого ресурсы различных видов.

ЛЕКЦИЯ 12

ВЫБОР ХАРАКТЕРИСТИК КАЧЕСТВА В ПРОЕКТАХ ПРОГРАММНЫХ СРЕДСТВ

12.1. Принципы выбора характеристик качества в проектах программных средств

Описание в стандарте **ISO 9126:1-4** характеристик качества программных средств не содержит рекомендаций и методик выбора их значений в требованиях к конкретным проектам. Необходимо, прежде всего, установить рациональные диапазоны мер и шкал для каждой характеристики и ее атрибутов, которые можно будет использовать в качестве первичных ограничений при выборе их значений для реальных проектов. Далее должны быть разработаны процессы выбора, установления и представления в спецификациях требований к атрибутам каждой характеристики качества. Эти требования должны учитывать реальные ограничения ресурсов, доступных для обеспечения ЖЦ ПС. Ресурсы этих процессов и атрибуты характеристик качества ниже, по возможности, сводятся к трудоемкости и длительности их реализации, а также к соответствующему влиянию этих параметров в функциональную пригодность.

Улучшение каждой характеристики качества требует некоторых затрат (трудоемкости, финансов, времени), которые в той или иной степени отражаются на основной характеристике качества — на ***функциональной пригодности***. При выборе конкретных мер и шкал конструктивных характеристик качества следует учитывать возможные затраты на их достижение и результирующее повышение функциональной пригодности, желательно, в сопоставимых экономических единицах, в тех же мерах и масштабах. Такое, даже качественное ***сравнение эффекта и затрат*** позволяет избежать многих нерентабельных повышений требований к отдельным

конструктивным характеристикам качества, которые не отражаются на адекватном улучшении функций ПС. Поэтому для каждого проекта необходимо ранжировать характеристики и их атрибуты и выделять, прежде всего, те, которые могут в наибольшей степени улучшить функциональную пригодность для конкретных целей. Таким образом, при системном анализе, формировании технического задания и спецификаций требований возникает ***два класса оптимизационных задач***:

— распределение затрат на улучшение отдельных, конструктивных характеристик ПС с целью достижения его максимальной или достаточно высокой функциональной пригодности;

— определение оптимальных или допустимых затрат на улучшение каждой конструктивной характеристики ПС, обеспечивающих адекватное или достаточно существенное увеличение качества функционирования.

Решение этих задач должно быть направлено на обеспечение достаточно высокой функциональной пригодности ПС ***путем сбалансированного улучшения остальных характеристик качества в условиях ограниченных ресурсов на ЖЦ***. Для этого в процессе системного анализа при подготовке технического задания и требований спецификаций, значения, требуемых атрибутов и субхарактеристик качества должны проверяться по степени их влияния на функциональную пригодность. Излишне высокие требования к отдельным атрибутам качества, требующие для реализации больших дополнительных трудовых и вычислительных ресурсов, целесообразно снижать, если они слабо влияют на основные, функциональные характеристики ПС. Таким образом, ограниченные ресурсы трудоемкости и длительности этапов ЖЦ ПС должны распределяться по процессам улучшения отдельных характеристик и атрибутов качества с учетом их воздействия на повышение функциональной пригодности.

Строгое формализованное решение этих задач в большинстве случаев невозможно, однако качественный системный анализ может помогать выявлению основных тенденций изменения и взаимосвязей значений характеристик. Наиболее просто могут быть установлены рациональные значения стандартизированных характеристик или их номинальные категории свойств для определенных классов ПС. При определении этих границ следует учитывать корреляцию как между атрибутами определенных характеристик, так и между различными характеристиками. Так, например, надежность функционирования ПС при больших нагрузках и перегрузках

может сильно зависеть от временной эффективности использования производительности ЭВМ. Используемость ресурсов ЭВМ может ограничивать сопровождаемость и изменяемость программ, и то и другое необходимо учитывать при определении требований к характеристикам конкретных проектов ПС.

После первичного выбора характеристик качества ПС необходимо определить экономическую эффективность и реализуемость программного средства в соответствии с ***требованиями контракта по качеству*** в условиях реального ограничения экономических ресурсов, доступных для обеспечения всего жизненного цикла комплекса программ. Достижение высокого качества любых изделий не может быть бесплатным, для этого необходимы определенные затраты ресурсов, которые тем больше, чем выше требуемое качество. Многие проекты информационных систем терпели и терпят неудачу из-за отсутствия у разработчиков и заказчиков при подготовке контракта четкого представления о реальных финансовых, трудовых, временных и иных ресурсах, необходимых для их реализации. Общее понятие — доступные ресурсы разработки — включает реальные финансовые, временные, кадровые и аппаратурные ограничения, в условиях которых происходит создание и развитие сложного комплекса программ. Эти факторы проявляются как ***дополнительные показатели качества продуктов*** и рентабельности процессов, которые следует учитывать и оптимизировать в ЖЦ ПС. При выборе и определении требований к характеристикам качества проекта программного средства могут использоваться два сценария.

Первый сценарий базируется на маркетинговых исследованиях рынка программных продуктов и на стремлении поставщика занять на рынке достаточно выгодное место. Для этого ему необходимо определить наличие на рынке всей гаммы близких по назначению и качеству ПС, оценить их экономическую эффективность, стоимость и применяемость, а также возможную конкурентоспособность предполагаемого программного продукта для потенциальных пользователей и их возможное число. Кроме того, следует оценить рентабельность затрат на обеспечение всего ЖЦ нового ПС и выявить функциональные и конструктивные ***характеристики качества, которые способны привлечь*** достаточно массовых покупателей и оправдать затраты на предстоящую разработку. Для этого потенциальные покупатели-пользователи перед приобретением ПС обычно оце-

нивают конкурентоспособность продукции на рынке ***по величине отношения***:

— возможной экономической ***эффективности*** (ценности) применения и качества программного продукта и способности удовлетворить пользователями свои потребности при его использовании;

— ***к стоимости*** (цене), которую готовы заплатить пользователи при приобретении и эксплуатации данного комплекса программ или базы данных.

При выборе продукта и поставщика покупатель стремится максимизировать это отношение как за счет поиска ПС с наилучшими функциями, эффективностью и высокими характеристиками качества, так и за счет минимальной стоимости покупаемого продукта. В этом сценарии при организации проектирования вся ответственность за цели и характеристики качества проекта ложится на его руководителей, и особую роль должны играть ***специалисты по маркетинговому анализу*** на рынке предполагаемого продукта. Они должны оценить риск успешного продвижения создаваемого продукта на рынок, сроки и график выполнения этапов жизненного цикла, потребность и достаточность ресурсов для реализации проекта, а также перспективы длительного развития, модификаций и распространения версий программного продукта.

Такие проекты обычно относительно невелики по объему и срокам реализации первой версии, однако могут предполагать длительный ЖЦ и множество модификаций для адаптации к нуждам и среде пользователей. Отбраковка вариантов реализации ПС ведется по показателю эффективность/стоимость для пользователей, с учетом конкурентоспособности и распространения на рынке. Этот сценарий экономического обоснования проектов ПС требует специфических маркетинговых исследований рынка подобных продуктов и их характеристик качества. Однако при этом должны обязательно учитываться затраты ресурсов на непосредственную разработку и обеспечение ЖЦ ПС, и возможная рентабельность проекта с учетом прогноза его жизненного цикла и распространения на рынке. Для этого в начале проектирования разработчикам необходимо прогнозировать затраты на создание и весь ЖЦ ПС, что анализируется при втором сценарии.

Второй сценарий предполагает наличие определенного заказчика — потребителя проекта ПС, который определяет основные технические и

экономические требования и характеристики качества. Он выбирает конкурентоспособного поставщика-разработчика, которого оценивает на возможность реализовать проект с необходимыми характеристиками качества с учетом ограничения сроков, бюджета и других ресурсов. Этому помогают опыт и экономические характеристики ранее выполненных проектов этой фирмы, но некоторые проекты могут не иметь прецедентов, и тогда приходится использовать имеющуюся статистику в этой области. При этом предполагается, что результаты разработки не обязательно подлежат широкому тиражированию, могут не поступать на открытый рынок, вследствие чего маркетинговые исследования для таких проектов не являются доминирующими и обычно предварительно могут не проводиться.

Однако для заказчика и разработчика при заключении контракта необходимо достаточно достоверное прогнозирование и экономическое обоснование требуемых ресурсов по трудоемкости, стоимости, срокам и другим характеристикам. Противоположность интересов поставщика и потребителя при оценивании стоимости и других ресурсов проекта требует поиска компромисса, при котором разработчик не продешевит, а заказчик не переплатит за конкретные выполненные работы и весь проект. Поэтому оба партнера заинтересованы в достоверном экономическом прогнозировании затрат ресурсов на проект ПС (см. лекцию 5).

Представленные выше (см. лекцию 11) характеристики и атрибуты качества имеют различное влияние на функциональную пригодность в зависимости от назначения и функций ПС, а также от субъективных взглядов заказчиков и потребителей соответствующих характеристик. Обычно ***наиболее сильное влияние функции ПС*** оказывают на требования к атрибутам характеристик защищенность — безопасность, надежность, эффективность и практичность. Эти атрибуты могут быть ранжированы по степени воздействия на функциональную пригодность в зависимости от назначения и особенностей ПС. Конкретные меры и диапазоны шкал этих характеристик следует определять в зависимости от их влияния на метрику качества в использовании по прямому назначению ПС основными пользователями.

В то же время характеристики сопровождаемость и мобильность ***относительно слабо связаны*** с назначением и конкретной функциональной пригодностью ПС. Их меры и шкалы определяются не столько конкретными функциями комплекса программ, сколько его архитектурой и приспособ-

ленностью интерфейсов к модификации и переносу на иные операционные и аппаратные платформы. Потребителями и оценщиками этих характеристик являются специалисты, обслуживающие расширение функций и развитие применения ПС, которые зачастую могут не учитывать непосредственно конкретные функции в своей деятельности. Поэтому метрика качества в использовании для этих характеристик приобретает иное значение: ее следует использовать при сопровождении и/или при переносе программ и данных, а не при их исполнении и применении ПС по прямому назначению.

Принципиальные и технические возможности и точность реализации и измерения значений атрибутов характеристик качества для конкретного проекта всегда ограничены в соответствии с их содержанием. Это определяет рациональные диапазоны значений каждого атрибута, которые могут быть выбраны для проекта ПС на основе требований заказчика, здравого смысла, а также путем анализа пилотных проектов и прецедентов в спецификациях требований реальных проектов. В пределах этих диапазонов целесообразно определение реальной достоверности оценивания и масштаба мер для описания соответствующего атрибута.

Процессы выбора и установления шкал и мер для описания характеристик качества проектов ПС можно разделить на ***два этапа***:

— предварительный выбор, формализация и обоснование набора исходных данных, отражающих общие особенности потребителей и этапы жизненного цикла проекта ПС, каждый из которых влияет на выбор определенных характеристик качества комплекса программ;

— выбор, установление и утверждение конкретных требований характеристик и атрибутов качества проекта для их последующего оценивания и применения при сопоставлении с реализованными требованиями спецификаций в процессе квалификационных испытаний или сертификации на определенных этапах жизненного цикла ПС.

На первом этапе следует использовать всю базовую номенклатуру характеристик, субхарактеристик и атрибутов, стандартизированных в **ISO 9126:1-4**. Их описания желательно предварительно упорядочить по приоритетам с учетом назначения и сферы применения конкретного ПС. Далее необходимо выделить и ранжировать по приоритетам потребителей, которым необходимы определенные показатели качества ПС с учетом их специализации и профессиональных интересов. Широкая номенклатура характеристик, представленная в стандарте **ISO 9126:1-4**, поддерживает

разнообразные требования, из которых следует селектировать и выбирать те, которые необходимы ***с позиции различных потребителей*** этих данных (см. выше табл. 6.1).

Выбранные значения характеристик качества и их атрибутов должны быть предварительно проверены разработчиками на их реализуемость с учетом доступных ресурсов конкретного проекта и при необходимости откорректированы по составу и значениям. В результате формируется полный ***набор требуемых характеристик, атрибутов, их мер и значений качества для конкретных потребителей в ЖЦ ПС.*** Результаты анализа и выбора номенклатуры и мер характеристик качества проекта ПС должны быть документированы в спецификациях требований, согласованы с их потребителями и утверждены заказчиком проекта для реализации. Изложенные положения иллюстрированы ниже, где приводится пример выбора и формирования требований к характеристикам качества программного средства сложной административной информационной системы.

12.2. Пример выбора и формирования требований к характеристикам качества программного средства

Разнообразие функций и потребителей характеристик качества программных средств делает невозможной унификацию всей совокупности требований к их составу и значениям для всех видов программных продуктов. Поэтому целесообразно ограничиться анализом выбора и формирования требований на ***гипотетическом примере ПС*** для некоторого типа систем. Приведенные ниже примеры обоснования и выбора требований к характеристикам качества ПС ***могут служить ориентирами*** для анализа и синтеза аналогичных данных в реальных проектах.

В качестве примера ниже за основу принята сложная административная система с определенными функциями и ограниченными условиями применения, с десятками действующих операторов-пользователей, работающих в реальном времени по схеме клиент-сервер. К такой системе предъявляются достаточно конкретные и высокие требования к разнообразию и качеству решения функциональных задач при относительно больших экономических и вычислительных ресурсах. Эти требования ниже формируются с позиции заказчиков и непосредственных пользователей ПС. Аналогами такой системы можно рассматривать банковские, налого-

вые, коммунальные, таможенные и другие административные системы, имеющие наборы разнообразных функциональных задач и интенсивные потоки исходной и результирующей информации в реальном времени. Регламент реального времени в них более легкий, чем в системах управления динамическими объектами, например, самолетами.

Таблица 12.1

Пример распределения приоритетов требований к характеристикам качества программного средства

Характеристика	Субхарактеристика	Приоритет требований
Функциональность	Функциональная пригодность	Высокий
	Корректность — правильность	Высокий
	Способность к взаимодействию	Средний
	Защищенность	Высокий
Надежность	Завершенность	Низкий
	Устойчивость к дефектам	Средний
	Восстанавливаемость	Высокий
	Доступность — готовность	Высокий
Эффективность	Временная эффективность	Высокий
	Используемость ресурсов	Средний
Практичность	Понятность	Средний
	Простота использования	Низкий
	Изучаемость	Средний
	Привлекательность	Низкий
Сопровождаемость	Анализируемость	Средний
	Изменяемость	Средний
	Тестируемость	Средний
Мобильность	Адаптируемость	Средний
	Простота установки	Средний
	Замещаемость	Низкий

Стандартизированные в **ISO 9126:1-4** характеристики качества ПС различаются по степени влияния на ***системную эффективность — функциональную пригодность*** их применения по прямому назначению. Вследствие реальной ограниченности ресурсов, которые доступны в жизненном цикле ПС, их необходимо выделять с учетом влияния на эту эффективность. Для этого целесообразно в процессе системного анализа присваивать и учитывать уровень приоритета требований к каждой субхарактеристике. В таблице 12.1 представлен пример возможного распределения приоритетов требований заказчика к субхарактеристикам ПС для принятого

варианта административной системы. Высшие приоритеты, естественно, выделены функциональной пригодности и в наибольшей степени поддерживающим ее характеристикам: корректности, защищенности, надежности, системной и временной эффективности. Эти приоритеты должны учитываться при выделении доступных ресурсов для достижения требуемых значений характеристик в жизненном цикле ПС. Остальным субхарактеристикам могут быть присвоены более низкие приоритеты и на выполнение соответствующих требований заказчика могут выделяться меньшие ресурсы, что, в частности, может отражаться на неполной реализации некоторых установленных заказчиком требований вследствие ограниченности ресурсов. Подобная таблица должна подготавливаться заказчиком как дополнение к техническому заданию и ***служить ориентиром*** при выборе предельных значений требований к субхарактеристикам и их атрибутам.

Выбор требований к мерам характеристик качества, естественно, должен начинаться с определения необходимых свойств и значений группы показателей, отражающих ***функциональные возможности ПС,*** куда по стандарту **ISO 9126:1-4** входят субхарактеристики: функциональная пригодность, корректность, способность к взаимодействию и защищенность — безопасность. Пример содержания и возможных диапазонов изменений атрибутов качества этих субхарактеристик был представлен выше в таблице 11.1 (см. лекцию 11). Для конкретного проекта ПС перечень атрибутов качества следует адаптировать и уточнять с учетом его назначения и функций.

Выбор и формирование ***требований к функциональной пригодности ПС*** — наиболее ответственная задача начальных этапов проектирования и всего последующего развития жизненного цикла. В данном примере эта задача не иллюстрируется вследствие необходимости излишнего расширения и еще большей детализации примера. Схема структурирования и содержания функциональных, структурных и эксплуатационных требований к характеристикам ПС приведена выше, в лекции 11, которая может использоваться для наполнения конкретными данными в реальных проектах. Все остальные стандартизированные характеристики ПС, в той или иной степени, должны способствовать обеспечению ***тактических целей*** выбранных конструктивных требований и достижению ***стратегических целей*** функциональной пригодности программного продукта.

Требования к субхарактеристике ***корректность*** могут представляться в виде описания двух основных свойств, которым должны соответство-

вать все программные компоненты и ПС в целом. Первое требование состоит в выполнении полной прослеживаемости сверху вниз реализации требований технического задания и спецификации на ПС при последовательной детализации описаний и верификации программных компонентов вплоть до текстов и объектного кода программ. Второе требование заключается в выборе степени и стратегии покрытия тестами программных компонентов, достаточной для функционирования ПС с необходимым качеством и точностью результатов, при реальных ограничениях ресурсов на тестирование. Реализация этих требований должна обеспечивать требуемую функциональную пригодность. Она должна поддерживаться разработчиками и независимыми экспертами в процессах жизненного цикла компонентов ПС, развиваться и обобщаться в составе и содержании внутренних и внешних метрик, а также метрик в использовании.

Требования к характеристике ***способность к взаимодействию*** могут быть достаточно полно формализованы и утверждены в процессе проектирования, с некоторыми уточнениями на последующих этапах. Их основой являются нормативные документы на интерфейсы открытых систем или выбранные для конкретного проекта стандарты де-факто. При выборе элементов программных компонентов, обеспечивающих способность к взаимодействию в конкретном проекте ПС, следует учитывать величину вычислительных ресурсов, необходимых для их реализации. Чем более полно используются международные стандарты открытых систем, тем шире способность к взаимодействию с различными повторно используемыми компонентами ПС, однако требуются большие затраты вычислительных ресурсов. При формализации интерфейсов важно также учитывать перспективы продолжительного сопровождения множества версий ПС, возможность повторного использования их компонентов и переноса на различные платформы. Унификация интерфейсов на взаимодействие с внутренней, внешней средой и с пользователями должна отражаться в специальных разделах технологической документации и должна иметь возможность проверки заказчиком и/или экспертами по документам и текстам программ без их исполнения.

Выбор и формирование требований к характеристике ***защищенность — безопасность*** должны осуществляться на основе потребностей эффективной и безопасной реализации назначения и функций ПС. В процессе системного анализа и проектирования должны быть выявлены по-

тенциальные предумышленные и случайные угрозы функционированию ПС и установлен необходимый уровень защиты данного комплекса программ. В соответствие с этим уровнем заказчиком выбираются и устанавливаются стандартизированная категория защищенности ПС и необходимый набор методов и средств контрмер защиты с учетом ограниченных ресурсов на их реализацию. В результате сформированные требования должны обеспечивать равнопрочную защиту от реальных угроз применению ПС и реализацию необходимых мер контроля и подтверждения целостности и характеристик качества функциональной пригодности комплекса программ в условиях проявления угроз.

Таблица 12.2

Пример требований к количественным характеристикам качества программного средства

Характеристики качества	Мера	Требуемое значение
Надежность		
Завершенность:		
— наработка на отказ при отсутствии рестарта.	Часы	10
Устойчивость:		
— наработка на отказ при наличии автоматического рестарта;	Часы	50
— относительные ресурсы на обеспечение надежности и рестарта.	%	10
Восстанавливаемость:		
— длительность восстановления.	Минуты	5
Доступность — готовность:		
— относительное время работоспособного функционирования.	Вероятность	0,998
Эффективность		
Временная эффективность:		
— время отклика — получения результатов на типовое задание;	Секунды	5
— пропускная способность — число типовых заданий, исполняемых в единицу времени.	Число в минуту	20
Используемость ресурсов:		
— относительная величина использования ресурсов ЭВМ при нормальном функционировании программного средства	Вероятность	0,8

Тактические цели выбора конструктивных характеристик качества стандарта **ISO 9126:1-4** последовательно рассмотрены и иллюстрированы таблицами 12.2 и 12.3. Пример требований к основным ***количественным характеристикам качества ПС*** сложной административной

системы представлен в таблице 12.2. Все меры и шкалы для атрибутов характеристик выбраны в соответствии с их содержанием из таблицы 11.2 (лекция 11). Требования к атрибутам характеристики ***надежность*** могут быть выбраны с учетом следующих факторов. При отсутствии автоматического рестарта, за счет отладки и при наличии администратора, контролирующего работоспособность ПС, можно считать допустимой наработку на отказ порядка 10 часов. За счет программно-аппаратных механизмов автоматического рестарта эта наработка при проявлении отказов может быть повышена приблизительно в 5 раз, т.е. при 80% отказов возможно их автоматическое обнаружение и оперативное восстановление, вследствие чего наработка на отказ возрастет до 50 часов. По опыту, на обеспечение этого может потребоваться около 10% вычислительных ресурсов системы. Предполагается, что для оперативной работы пользователей административной системы допустимая длительность прерывания работы для полного восстановления нормального функционирования системы может составлять не более 5 минут. В результате при таких значениях атрибутов надежности коэффициент готовности — вероятность застать ПС в работоспособном состоянии — составит достаточно высокую величину 0,998.

Так же как при формировании требований к корректности, для характеристики надежности большое значение имеет установление требований к степени покрытия тестами в процессе отладки структуры программных компонентов и ПС в целом. Формализацию этой характеристики целесообразно устанавливать отдельно от общих характеристик качества для проверки на технологических этапах тестирования и испытаний. При этом следует учитывать, что в примере принято весьма малое время восстановления, обусловленное мелкими программными дефектами без учета физического разрушения компонентов, которое должно поддерживаться дублированием вычислительных средств и высокой автоматизацией процессов аппаратурного обеспечения надежности ПС.

Основные требования к атрибутам характеристики ***эффективность использования вычислительных ресурсов системы*** сосредоточены на наиболее критичных показателях производительности и длительности решения функциональных задач. В отличие от объемов памяти, временные характеристики труднее устанавливать и измерять и их ограниченность сильнее влияет на функциональную пригодность ПС. Для оперативной

работы пользователей важно иметь малое (несколько секунд) время отклика из ЭВМ после получения типового задания и начала решения требуемой функциональной задачи. Это время обычно желательно иметь в пределах нескольких (для примера принято пяти) секунд, хотя длительность полной реализации задания может быть значительно больше. Требуемая пропускная способность решения функциональных задач зависит от их содержания и числа действующих пользователей. В примере предполагается, что десять операторов могут вводить в минуту по два задания каждый, которые должны исполняться в отведенное время без дополнительной задержки, что приводит к требованию пропускной способности данного ПС на выбранной вычислительной среде — 20 заданий в минуту.

Таблица 12.3

Пример требований к качественным характеристикам программного средства

Характеристики качества	Мера	Требуемое значение
Практичность		
Простота использования:		
— среднее время ввода заданий;	Секунды	10
— среднее время отклика на задание.	Секунды	5
Изучаемость:		
— трудоемкость изучения применения ПС;	Чел.-часы	200
— продолжительность изучения;	Часы	50
— объем эксплуатационной документации;	Страницы	1000
Сопровождаемость		
Изменяемость:		
— трудоемкость подготовки изменений;	Чел.-часы	10
— длительность подготовки изменений.	Часы	5
Тестируемость:		
— трудоемкость тестирования изменений;	Чел.-часы	20
— длительность тестирования изменений.	Часы	5
Мобильность		
Адаптируемость:		
— трудоемкость адаптации;	Чел.-часы	50
— длительность адаптации.	Часы	10
Простота установки:		
— трудоемкость инсталляции;	Чел.-часы	10
— длительность инсталляции.	Часы	5
Замещаемость:		
— трудоемкость замены компонентов;	Чел.-часы	50
— длительность замены компонентов	Часы	10

Требования к используемости ресурсов памяти и производительности вычислительных средств могут устанавливаться исходя, с одной стороны, из экономической целесообразности применения наиболее дешевой, с минимальными ресурсами ЭВМ, загрузка которой будет в среднем не ниже 0,5. С другой стороны, высокая загрузка (выше 0,9) может приводить к нежелательной задержке или даже потере заданий при случайном, кратковременном повышении их интенсивностей, что может негативно отразиться на функциональной пригодности. Таким образом, в данном примере рациональная величина вероятности использования ресурсов ЭВМ в процессе нормального функционирования ПС должна находится в пределах 0,8.

Пример требований к ***качественным характеристикам ПС*** представлен в таблице 12.3, которая базируется на мерах и шкалах таблицы 11.3 (лекция 11) с небольшими сокращениями наиболее неопределенных качественных атрибутов, которые выбираются и описываются экспертами в виде наборов свойств программ. Основная часть атрибутов качества в таблице 12.3 сведена к экономическим показателям трудоемкости и длительности, требуемых для реализации соответствующих характеристик.

Требования к ***практичности*** и ее субхарактеристикам — понятности и простоты использования — зависят от назначения и функций ПС и могут качественно формализоваться заказчиками набором свойств, необходимых для удобной и комфортной эксплуатации программ. Количественно простоту использования можно в некоторой степени характеризовать требованиями ограничения средней длительности ввода типовых заданий и времени отклика на них, которое должно быть в несколько раз меньше. Требования к продолжительности изучения ПС, достаточной для эффективной эксплуатации сложной административной системы квалифицированным специалистом, в данном примере могут составить около недели или порядка 50 часов. Для коллектива из четырех человек-эксплуатационников это потребует трудоемкости около 200 человеко-часов. Для обеспечения полноценного изучения процессов применения ПС этими специалистами может быть необходима эксплуатационная документация объемом около 1000 страниц, а также желательны адекватные по содержанию электронные учебники. Малый объем эксплуатационной документации может снизить качество и полноту использования функций сложного ПС, а очень большой объем — также может ухудшить эксплуатацию из-за

трудности выделения и освоения наиболее существенных свойств и особенностей применения ПС из множества второстепенных деталей.

Требования к компонентам ***сопровождаемости*** количественно можно установить для субхарактеристик изменяемости и тестируемости. Требуемые значения зависят от четкости концепции и архитектуры ПС, от унифицированности внутренних, внешних и с пользователями интерфейсов, от качества технологической документации, а также от инструментальной оснащенности ЖЦ данного ПС и еще от некоторых факторов. Обобщенно это отражается на длительности и трудоемкости подготовки и реализации типовых модификаций, обусловленных необходимостью устранения дефектов и небольшими усовершенствованиями функций ПС. В рассматриваемом примере для подготовки и выполнения каждого изменения (без учета затрат времени на обнаружение и локализацию дефекта) можно принять среднюю продолжительность в 5 часов и суммарную трудоемкость двух специалистов около 10 человеко-часов. Требования к продолжительности тестирования таких изменений могут составить также до 5 часов, но трудоемкость может увеличиться до 20 человеко-часов, так как требуемый коллектив тестировщиков может возрасти до трех-четырех специалистов.

Выбор и установление требований к ***мобильности*** ПС в данном примере сведены к трудоемкости и длительности процессов: адаптации к характеристикам пользователей и внешней среды, инсталляции версий ПС в среде пользователей и замены крупных компонентов версий ПС по требованиям заказчиков или конкретных пользователей. Наиболее простым и легко формализуемым из перечисленных процессов является инсталляция готовой версии ПС с комплектом документации без дополнительных изменений на платформе пользователя, которая может требовать до 5 часов работы двух специалистов (10 человеко-часов). Более сложный процесс включает адаптацию ПС по формализованным инструкциям к специфической аппаратной и внешней среде конкретного пользователя, которая может потребовать вдвое большего времени и в несколько раз (в примере 5) большего числа специалистов. Еще более сложный и трудоемкий процесс замены крупных компонентов ПС и перенос их на иную аппаратурную и операционную платформу. Для этого процесса в примере требуется не менее 20 часов и коллектив около 5 человек (100 человеко-часов).

Рассмотренный пример выбора и формирования характеристик качества проекта ПС может служить ***ориентиром подходов*** при анализе факторов и реализации процессов установления требований к ним в технических заданиях и спецификациях. Обсуждение и согласование между заказчиком и разработчиком рациональных значений всего ансамбля характеристик и их атрибутов качества позволяет избегать как нецелесообразного завышения требований, так и снижения требований к отдельным характеристикам, которые могут негативно отразиться на функциональной пригодности ПС. При этом важно учитывать ограниченность ресурсов при выборе мер и шкал характеристик в жизненном цикле ПС и необходимость компромиссов между ними вследствие многочисленных связей и взаимовлияний. Подобный анализ может эффективно отражаться на снижении стоимости, трудоемкости и длительности создания ПС и на повышении экономической эффективности всего их жизненного цикла.

ЛЕКЦИЯ 13
ВЕРИФИКАЦИЯ, ТЕСТИРОВАНИЕ И ОЦЕНИВАНИЕ КОРРЕКТНОСТИ ПРОГРАММНЫХ КОМПОНЕНТОВ

13.1. Принципы верификации и тестирования программ

Верификация — это процесс для определения, выполняют ли программные средства и их компоненты требования, наложенные на них в последовательных этапах ЖЦ ПС. Анализ, просмотры (обзоры) и тестирование от требований являются важнейшей частью верификации и установления корректности программ. Основная цель верификации ПС состоит в том, чтобы обнаружить, зарегистрировать и устранить дефекты и ошибки, которые внесены во время последовательной разработки или модификации программ. Для эффективности затрат ресурсов при ее реализации верификация должна быть интегрирована как можно раньше с процессами проектирования, разработки и сопровождения. Обычно она проводится сверху вниз, начиная от общих требований, заданных в техническом задании и/или спецификации на всю информационную систему до детальных требований на программные модули и их взаимодействие.

Назначение верификации ПС — последовательно проверить, что в реализованном комплексе программ (рис. 13.1):

— общие требования к информационной системе, предназначенные для программной реализации, корректно переработаны в спецификацию требований высокого уровня к комплексу программ, удовлетворяющих исходным системным требованиям;

— требования высокого уровня правильно переработаны в архитектуру ПС и в спецификации требований к функциональным компонентам низкого уровня, которые удовлетворяют требованиям высокого уровня;

Рис. 13.1

— спецификации требований к функциональным компонентам ПС, расположенным между компонентами высокого и низкого уровня, каждый раз удовлетворяют требованиям более высокого уровня;

— архитектура ПС и спецификации требований к компонентам низкого уровня корректно переработаны в удовлетворяющие им исходные тексты программных и информационных модулей;

— исполняемый объектный код удовлетворяет требованиям к исходному тексту программных компонентов.

Кроме того, верификации на соответствие спецификации требований на конкретный проект программного средства подлежат требования к технологическому обеспечению ЖЦ ПС, а также требования к эксплуатационной и технологической документации.

Цели верификации ПС достигаются посредством последовательного выполнения комбинации из просмотров, анализов, разработки тестовых сценариев и процедур и последующего выполнения этих процедур. Тестовые сценарии предназначены для проверки внутренней непротиворечивости и полноты реализации требований. Выполнение тестовых процедур должно обеспечивать демонстрацию соответствия испытываемых программ исходным требованиям. Информация о процессе верификации включает требования к системе, требования к ПС и к его архитектуре, данные о прослеживаемости последовательного преобразования требований, исходный текст программ, исполняемый объектный код, План верификации ПС, План квалификационного тестирования ПС. Результаты верификации должны быть включены в документы: выполненные процедуры верификации ПС, описание и отчет о квалификационном тестировании ПС и его компонентов.

Просмотры и анализы требований высокого уровня предназначены для того, чтобы обнаруживать, регистрировать и устранять дефекты и ошибки, которые внесены в процессе последовательной разработки и детализации спецификаций требований к ПС. Эти просмотры и анализы должны подтвердить корректность и согласованность требований высокого уровня, а также гарантировать, что:

— полностью определены функции информационной системы, которые должно выполнять ПС;

— требования по функциональности, эффективности и к качеству системы детализированы в исходных требованиях высокого уровня к ПС и что правильно определены производные требования и обоснована их необходимость;

— каждое требование высокого уровня к ПС является точным, однозначным и достаточно детализированным и что требования не конфликтуют друг с другом;

— не существует никаких конфликтов между требованиями высокого уровня и возможностями аппаратных и программных средств объектного вычислителя, особенно такими, как время реакции системы и характеристики аппаратуры ввода/вывода;

— процесс разработки требований к ПС полностью соответствует стандартам на создание спецификаций требований и любые отклонения от стандартов обоснованны;

— функциональные и конструктивные характеристики качества, предназначенные для программной реализации, полностью включены в требования высокого уровня к ПС.

Просмотры и анализы исходных текстов программ предназначены для выявления и регистрации дефектов и ошибок, которые внесены в процессе программирования компонентов ПС. Эти процедуры должны подтверждать, что выходные результаты кодирования алгоритмов — тексты программ являются точными, полными и могут быть проверены. Прежде всего, должна определяться корректность текста программ по отношению к исходным требованиям и к архитектуре ПС и соответствие стандартам на программирование. Эти просмотры и анализы обычно ограничиваются исходным текстом программ, которые подтверждают, что он является корректным, полным и соответствует требованиям низкого уровня к компонентам, а также то, что исходный текст не содержит излишних и неописанных функций. Должно быть проверено, что процесс разработки программ полностью осуществлялся в соответствии со стандартами на программирование и отклонения от этих стандартов обоснованны, особенно в случаях ограничения сложности и использования конструкций программ, которые должны удовлетворять целям корректности системы. Сложность в данном контексте понимается как степень связности программных компонентов, уровень вложенности управляющих структур и сложность логических и числовых выражений. Этот анализ должен гарантировать, что возможные отклонения от стандартов оправданны.

Тестирование ПС от требований имеет две взаимодополняющие цели. Первая цель — показать, что ПС удовлетворяет заданным требованиям к нему. Вторая цель — показать с высокой степенью доверия, что

устранены дефекты и ошибки, которые могли бы привести к возникновению недопустимых отказовых ситуаций, влияющих на корректность и надежность системы. Тестирование интеграции программных компонентов, основанное на требованиях, должно акцентироваться на взаимосвязях между требованиями к ПС и на реализации требований к его архитектуре. Цель такого тестирования — гарантировать, что программные компоненты взаимодействуют друг с другом корректно и удовлетворяют требованиям к ПС и к его архитектуре.

Многолетний опыт показал, что единственный универсальный метод тестирования создать невозможно и следует применять упорядоченный ряд значительно различающихся методов. Каждый метод отличается целевыми задачами тестирования, проверяемыми компонентами программы и методами оценки результатов. ***Только совместное и систематическое применение различных методов тестирования позволяет достигать высокого качества функционирования сложных комплексов программ.***

При выборе и применении методов тестирования программных компонентов необходимо учитывать общие требования к ним и их ***особенности***:

— относительно высокая доля творческого труда специалистов, осуществляющих тестирование, приводит к необходимости обеспечения высокоэффективного интерактивного их взаимодействия со средствами автоматизации проверок;

— непредсказуемость видов и мест выявляемых ошибок в программах ограничивает возможность автоматического их обнаружения и определяет необходимость ориентироваться на частично автоматизированные методы и средства при творческой, высокой роли специалистов при тестировании;

— разнообразие возможных мест расположения и видов ошибок, при относительно редком их обнаружении, приводит к регистрации и анализу большого объема избыточной информации о процессе исполнения программ при тестировании;

— высокая сложность отлаживаемых программных компонентов, творческий и итерационный характер процесса тестирования затрудняют и ограничивают возможную точность оценки полноты проведенного тестирования и достигнутого качества компонентов;

— активное участие в тестировании специалистов, различающихся по квалификации, опыту, темпераменту и творческим возможностям, а также различия архитектуры, сложности и функций отлаживаемых про-

граммных компонентов не позволяют жестко регламентировать трудоемкость, методики и технологии применения видов и средств автоматизации тестирования.

В процессе разработки программ специалисты должны стремиться охватить тестированием функционирование ПС во всей доступной области изменения исходных данных и режимов применения. При этом номенклатура и диапазоны варьирования тестов ограничены либо допустимой длительностью подготовки и исполнения тестов, либо вычислительными ресурсами, доступными для использования с целью контроля и тестирования в режиме нормальной эксплуатации. Это отражается на выборе методов тестирования и последовательности анализа компонентов ПС, объеме применяемых тестов и на глубине тестирования. На выбор эффективных методов тестирования и последовательность их применения в наибольшей степени влияют основные характеристики тестируемых объектов:

— класс комплекса программ, определяющийся глубиной связи его функционирования с реальным временем и случайными воздействиями из внешней среды, а также требования к качеству обработки информации и надежности функционирования;

— сложность или масштаб (размеры) комплекса программ и его функциональных компонентов, являющихся конечными результатами разработки;

— преобладающие элементы в программах: осуществляющие вычисления сложных выражений и преобразования измеряемых величин или обрабатывающие логические и символьные данные для подготовки и отображения решений.

Тестовые варианты должны быть разработаны так, чтобы верифицировать корректность функционирования и определить условия, при которых могут проявляться потенциальные ошибки. ***Анализ покрытия тестами требований к ПС*** должен определять, какие требования не были протестированы и какие структуры программного средства не были исполнены при тестировании. Анализ тестового покрытия состоит из ***двух шагов***, включающих анализ покрытия, основанного на требованиях, и анализ структурного покрытия. ***Первый шаг*** анализирует тестовые наборы относительно требований ПС, чтобы подтвердить, что выбранные наборы тестов удовлетворяют установленным критериям. Этот анализ покрытия, основанного на требованиях, должен определить, насколько полно тести-

рование проверило реализацию всех требований в спецификации к ПС, и показать потребность в дополнительных тестовых наборах. Тестовые варианты, основанные на требованиях, могут не полностью покрыть структуру программы. Поэтому дополнительно выполняется анализ структурного покрытия и проводится его верификация. ***Второй шаг*** должен подтвердить, что процедуры тестирования, основанные на требованиях, покрыли всю структуру программы. Анализ структурного покрытия должен определять, не пропущены ли элементы структуры программы, которые не проверены тестовыми процедурами, основанными на требованиях.

Далее внимание сосредоточено на основных методах обеспечения ***качества относительно простых программ***. К ним относятся отдельные программные модули (ПМ) или их небольшие группы, решающие достаточно простую функциональную задачу. Однако ниже эти компоненты не различаются, и преимущественно используется термин — программный модуль. Эти объекты тестирования имеют ряд принципиальных особенностей, которые отличают их от сложных программных средств и непосредственно отражаются на применяемых методах и средствах автоматизации тестирования.

Невысокая размерность ПМ обеспечивает их обозримость и возможность детального анализа функций, структуры программы и процесса решения задач. Детализированный контроль ПМ может проводиться планомерно и детерминированно с точностью до любого оператора или операнда в программе. Стремление к снижению сложности тестирования реализуется частично путем укрупнения наблюдаемых и анализируемых компонентов до уровня небольших групп операторов на языке программирования в пределах одного структурного элемента программы. Тем самым контролю доступны вся логика решения задачи и все возможные маршруты и варианты исполнения программы. В результате при тестировании имеется возможность оценивать и контролировать степень проведенной проверки и достигнутое качество компонентов ПС, а также выделять следующие виды тестирования.

Тестирование для обнаружения ошибок имеет целью выявление отклонений результатов функционирования реальной программы от заданных требований и эталонных значений. Задача состоит в обнаружении максимального числа дефектов программы, в качестве которых принимается любое отклонение результатов от эталонов. Успешным является тес-

тирование, которое приводит к обнаружению существования ошибок при минимальных затратах.

Тестирования для диагностики и локализации ошибок предназначено для того, чтобы точно установить исходное место искажения программ или данных, являющегося причиной и источником отклонения результатов от эталонов при тестировании для обнаружения ошибок. Эффективными являются тесты, способствующие быстрой и точной локализации первичных дефектов программ и данных. На этой стадии затраты оправданы и тестирование можно считать успешным, если оно привело к определению элементов программы или данных, подлежащих непосредственной корректировке.

Методы и стратегии тестирования программных компонентов используются для обработки текстов программ в различной форме и для представления информации о результатах тестирования в виде, удобном для анализа специалистами. Форма представления и исполнения программы для тестирования зависит от этапов разработки, уровня языков программирования, наличия соответствующих средств автоматизации и других факторов. Программы в процессе тестирования используются в ***двух принципиально различных формах представления***:

— в виде текста ***на языке программирования*** или формализованного описания спецификаций требований (символьное представление), удобного для анализа человеком и обычно недоступного для непосредственного получения результатов функционирования на ЭВМ;

— в ***машинном коде*** конкретной ЭВМ (объектное представление), пригодном для автоматической обработки определенных кодовых исходных данных и неудобном для их анализа человеком.

Первичные и вторичные ошибки выявляются на последовательных этапах тестирования (см. лекцию 10). Вторичные ошибки являются определяющими для качества функционирования программ, так как не каждая первичная ошибка вносит заметный вклад в искажение выходных результатов. Вследствие этого ряд первичных ошибок может оставаться необнаруженным и, по существу, не влияет негативно на функциональные характеристики программ. Существенной особенностью тестирования ПМ является относительная ***близость проявления вторичных ошибок к их причинам — первичным ошибкам***. Это означает, что последствия ошибок обнаруживаются в искажениях результатов непосредственно в месте

локализации первичной ошибки или после небольшого числа шагов исполнения программы. Это значительно облегчает их диагностику и локализацию, а также контроль правильности проведенных корректировок.

Анализ программы на уровне символов и операторов языка программирования обеспечивает все виды тестирования: для обнаружения ошибок, для их локализации и для проверки правильности выполненных корректировок. Эта особенность используется не только при автономной отладке модулей, но и при комплексном тестировании и обнаружении дефектов функционирования ПС. В последнем случае приходится итерационно углубляться в проверяемые компоненты, вплоть до модуля и его операторов. При этом для локализации и исправления первичной ошибки в ряде случаев приходится возвращаться к тестированию модулей на уровне операторов программы.

Еще одной особенностью тестирования ПМ является необходимость и возможность обеспечения гарантий их высокого качества и пригодности для использования в разных проектах ПС. ***Перспективы многократного использования апробированных программных компонентов*** в различных версиях ПС могут быть обеспечены только при четкой формализации их функций и условий применения, а также при доверии к корректной реализации функций и интерфейсов в определенной среде. Вследствие этого необходима систематичность тестирования и формализованный контроль достигнутой корректности после каждого цикла проверок. Завершать отладку и испытания ПМ следует их аттестацией, с приложением перечня реализуемых требований, интерфейсов, характеристик полноты тестирования и диапазонов варьирования тестовых данных. Такая аттестация может служить гарантией качества для многократного использования ПМ в различной аппаратной и операционной среде.

Тестирование потоков управления (структуры программы) должно быть начальным этапом, так как при некорректной структуре возможны наиболее грубые искажения выходных результатов и даже отсутствие некоторых из них. Оно состоит в проверке корректности последовательностей передач управления и формирования маршрутов исполнения программы при обработке тестов. Для тестирования структуры программ в большинстве случаев требуются относительно меньшие затраты по сравнению с тестированием на потоках данных.

Тестирование потоков данных можно разделить на два этапа. Первый этап тестирования состоит в анализе обработки данных, определяющих значения предикатов в операторах выработки логических решений. Эти решения влияют на маршруты обработки информации, что сближает в этой части метод тестирования потоков данных с тестированием структуры программы. Второй этап тестирования обработки данных состоит в проверке вычислений по аналитическим формулам или численных значений результатов в зависимости от числовых или логических значений исходных данных. В качестве эталонов используются результаты ручных или автоматизированных расчетов по тем же или близким по содержанию формулам.

Любые методы тестирования ПМ, в большей или меньшей степени, ориентированы на обнаружение ошибок определенных типов. Методы тестирования потоков управления предназначены преимущественно для обнаружения ошибок в структуре ПМ и реализуемых маршрутах обработки информации. Методы тестирования потоков данных обеспечивают выявление ошибок в вычислительной части программ и в процессах преобразования различной информации. Эти методы тестирования применяются как при представлении программ на языках программирования, так и при исполнении их в объектном (машинном) коде после трансляции. Такая ориентация позволяет упорядочить последовательность применения методов с целью устранения, прежде всего, ошибок, в наибольшей степени отражающихся на корректности программ, а также сосредоточиваться на методах, позволяющих решать частные задачи достижения необходимого их качества при минимальных затратах.

Совокупность спецификаций тестов может рассматриваться как ***второе, независимое описание содержания и реализации последовательных процедур комплекса программ***. Жизненный цикл и развитие тестов при разработке и сопровождении ПС должны проходить во времени, параллельно динамике изменения и жизненному циклу текстов программ. Тесты и сценарии их реализации должны быть адекватными и полностью отражать содержание текстов компонентов комплексов программ, но в иной форме. Их следует рассматривать и анализировать как еще одну форму описания функций программ и данных ПС, которые также могут содержать дефекты и ошибки. Таким образом, программной (процедурной) форме представления ПС должно полностью соответствовать его

равноправное содержание в форме сценариев и тестов для проверки их взаимного соответствия. При этом дефекты и ошибки возможны в обеих формах описания содержания программ, определение их места и устранение является основной задачей тестирования.

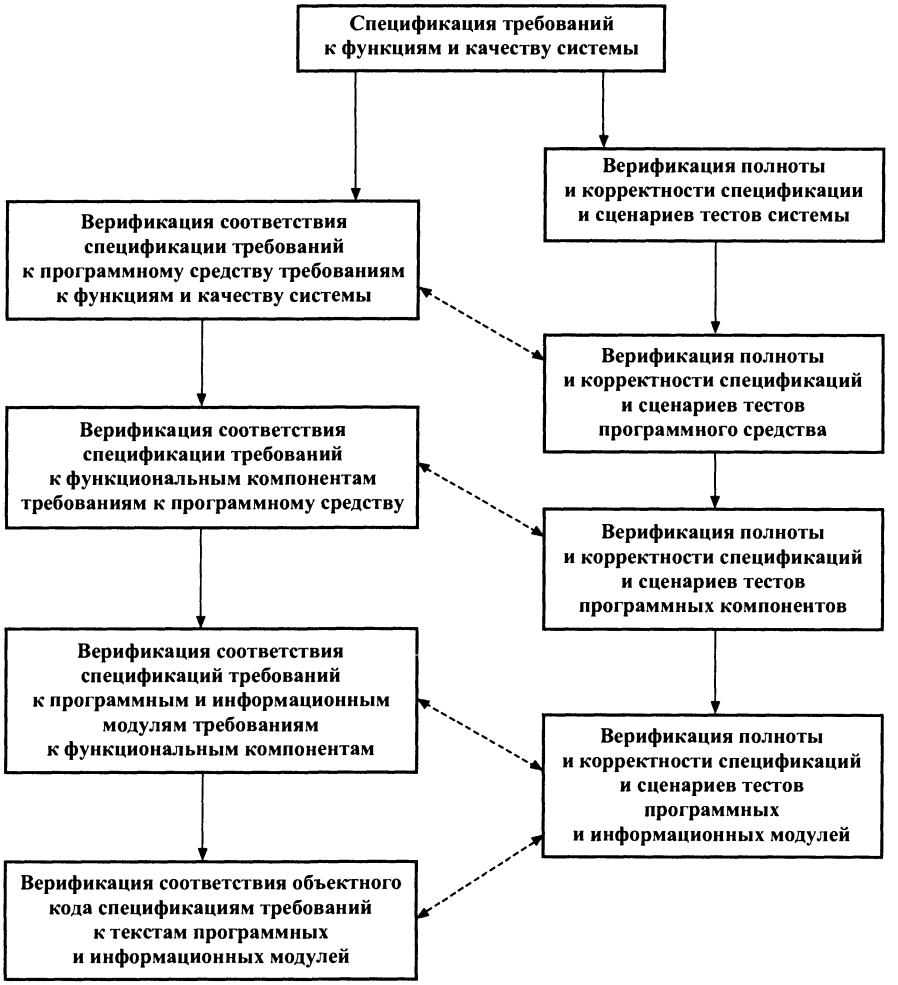

Рис. 13.2

Для обеспечения высокого качества программного продукта параллельно с ***верификацией требований*** и программированием корректировок целесообразно разрабатывать и верифицировать ***спецификации и сценарии тестов***, отражающие методы и конкретные процедуры проверки реализации изменений этих требований (рис. 13.2). Тестовые спецификации могут использоваться для проверки согласованности, внутренней непротиворечивости и полноты реализации требований без исполнения программ. Для каждого требования к комплексу программ, его архитектуре, функциональным компонентам и модулям должна быть разработана спецификация тестов, обеспечивающая проверку корректности, адекватности и возможности в последующем реализовать тестирование компонента на соответствие этому требованию. Такая взаимная проверка функций компонентов, отраженных в требованиях и в спецификациях тестов, обеспечивает повышение их качества, сокращение дефектов, ошибок, неоднозначностей и противоречий.

При тестировании программ зачастую оказывается, что не каждое требование в спецификации описано достаточно полно и корректно, чтобы оно могло быть проверено тестами. При разработке тестов на основе таких спецификаций вследствие их возможной неопределенности может обнаруживаться, что ***не для каждого требования*** к программе или данным может быть подготовлен тест. С другой стороны, ***не для каждого теста*** может оказаться в спецификации адекватное требование на функции ПС. Спецификации тестов должны обеспечивать дополнительный контроль корректности спецификаций требований и верификацию взаимодействия компонентов на соответствующем уровне описания ПС. Независимая разработка спецификаций тестов на основе спецификаций требований создает базу для обнаружения, какие требования не тестировались или принципиально не могут быть проверены тестированием. Таким образом, ***верификация спецификаций требований на*** программные компоненты и ПС могут использоваться с ***двумя целями*** (см. рис.13.2):

— для разработки, программирования и проверки текстов программ и интерфейсов взаимодействия программных компонентов разных уровней в комплексе программ;

— для создания скоординированного комплекса тестов для совокупности компонентов, обеспечивающих взаимную проверку реализации спецификаций требований на комплекс программ и компоненты.

В результате совокупности спецификаций требований к тестам могут применяться при разработке и сопровождении ***как эталоны и вторая адекватная форма описания содержания программ*** для сквозной верификации спецификаций требований к тестам сверху вниз, а также сами подвергаться верификации на корректность соответствия исходным требованиям к компонентам текстов программ и данных разного уровня. Такие ***параллельные взаимные проверки*** спецификаций требований и текстов программ и спецификаций тестов способствуют выявлению и исключению множества вторичных дефектов и ошибок в ПС. Впоследствии эти спецификации тестов должны использоваться для непосредственного тестирования исполнения требований к программным компонентам соответствующего уровня. Кроме того, параллельная и независимая разработка, с одной стороны, спецификаций программ и спецификаций тестов, а также их реализации, с другой стороны, позволяет распараллеливать работы ЖЦ ПС, что ведет к сокращению сроков создания компонентов и комплексов программ.

Реализация этих целей взаимодействия верификации и тестирования может производиться разными методами и независимыми специалистами — программистами и тестировщиками, что позволяет использовать результаты их деятельности для сравнения содержания одних и тех же программ, представленных на языках программирования и описанных на языках тестов. Особенности описаний и реализации программ, а также ***мышления программистов*** — на основе функций и процедур исполнения программ существенно ***отличаются от представлений и методов описаний тех же функций программ тестировщиками*** — создателями сценариев тестирования. Они акцентируют деятельность на конкретных процедурах проверки функционирования, возможных результатах и взаимодействии компонентов ПС. Это позволяет выявлять вторичные дефекты, появляющиеся при корректировках, и повышать качество разработки и сопровождения путем сопоставления двух методов и результатов описания одних и тех же программ, за счет того, что мала вероятность одинаковых ошибок в сценариях тестов и в реализации текстов программ.

13.2. Процессы и средства тестирования программных компонентов

Относительная простота ПМ позволяет детально анализировать их внутреннюю структуру и любой маршрут исполнения программы. Это обеспечивает возможность реализации ***двух стратегий тестирования***: от структуры и от данных. Этим двум стратегиям соответствуют два метода тестирования программ: метод анализа потоков управления и анализа потоков данных. Методы дополняют друг друга, и каждый может быть доминирующим на начальных этапах отладки в зависимости от типа объекта и условий тестирования.

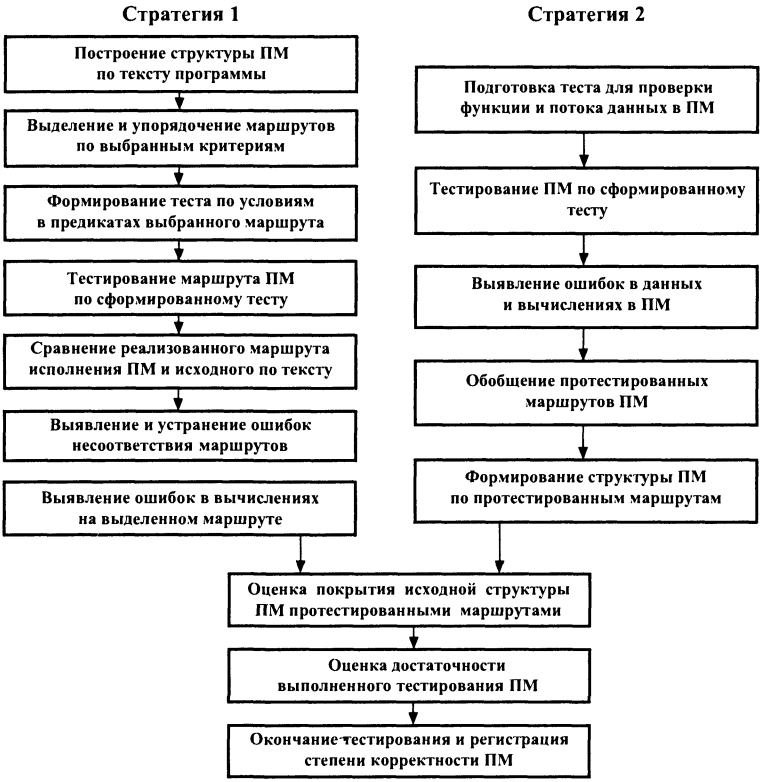

Рис. 13.3

При ***первой стратегии*** (рис. 13.3) за основу принимается структура ПМ, построенная по тексту программы в виде графа. В графе программы по некоторым критериям выделяются и упорядочиваются маршруты исполнения программы и условия-предикаты, при которых они могут быть реализованы. Эти условия используются для подготовки тестовых наборов, каждый из которых должен реализоваться по маршруту, принятому за эталон при подготовке теста. Отклонение исполнения теста от первоначально выбранного маршрута рассматривается как ошибка, причина которой может быть либо в первичной структуре ПМ, либо в реализации конкретного маршрута при данном тесте на входе.

После устранения ошибок и несоответствия выбранных и реализованных маршрутов для каждого из них проверяется процесс обработки данных и выявляются ошибки в результатах их преобразования. Затем оценивается достаточность выполненного тестирования ***по степени покрытия исходного графа программы*** проверенными маршрутами, которые выделялись по выбранному или заданному критерию. Завершается тестирование при требуемом покрытии графа программы протестированными маршрутами или при использовании ресурсов, выделенных на тестирование. В последнем случае необходимы оценка достигнутой корректности программы и регистрация этой величины.

При этой стратегии некоторые выделяемые маршруты могут оказаться принципиально нереализуемыми из-за противоречивых условий в последовательных операторах — вершинах графа программы. Вследствие этого для некоторых модулей могут подготавливаться тесты, которые являются избыточными и не отражают реальное функционирование программы. Тем не менее данная стратегия имеет преимущества при тестировании логических программ с малой долей вычислений. При этом достаточно эффективно контролируется полнота тестирования при различных критериях выделения маршрутов и методах их упорядочения.

При ***второй стратегии*** (см. рис. 13.3) за основу принимаются требования спецификаций, конкретные тестовые и эталонные значения, которые подготавливаются специалистами путем анализа переменных и предикатов в тексте программы. При каждом тесте программа исполняется по определенному маршруту, который регистрируется. При этом реализованный в соответствии с анализируемыми требованиями маршрут и поток данных рассматриваются как эталонные компоненты структуры програм-

мы. По мере тестирования отмечаются проверенные операторы и оценивается полнота ***покрытия тестами требований спецификаций*** на маршрутах тестирования. Накопление и обобщение реализованных маршрутов позволяет воспроизвести структуру программы и контролировать степень покрытия каждого компонента. Для этого на каждом этапе тестирования выделяются компоненты программы, оставшиеся непроверенными, для которых необходимо подготавливать дополнительные тесты. Однако при такой стратегии трудно оценить степень покрытия и проверки всех маршрутов исполнения программы без использования структурного графа, построенного по ее исходному тексту.

Данная стратегия имеет преимущества при сравнительно простой структуре программы и при преобладании в ней вычислительных операторов. На каждом маршруте упорядоченное варьирование переменных акцентирует внимание на вычислительной части программы, что существенно для такого класса ПМ. Однако возрастает риск пропустить сочетания предикатов, определяющих непроверенный маршрут. Поэтому практически всегда целесообразно ***совместное применение двух стратегий***, с акцентом на одну из них в зависимости от особенностей тестируемого ПМ или его частей. Программы с преобладанием сложной логической структуры и с относительно малой вычислительной частью целесообразно начинать тестировать по первой стратегии и только для маршрутов с вычислительными операторами использовать анализ потоков данных (вторую стратегию). В модулях, имеющих простую структуру и содержащих значительный объем вычислений после первичной отладки по второй стратегии, целесообразна проверка полноты тестирования структуры и завершение отладки по первой стратегии.

Известно, что в крупных ПС исчерпывающее тестирование, гарантирующее полное отсутствие в них дефектов и ошибок, принципиально невозможно и число дефектов, обнаруживаемых в программах даже независимыми тестировщиками, имеет пределы. Представленное выше прослеживание соответствия компонентов при верификации корректности сверху вниз от исходных требований к комплексу программ и тестирование покрытия компонентов на соответствие требованиям на каждом уровне их детализации может потребовать значительных вычислительных, трудовых и временных ресурсов. Реальные ограничения доступных ресурсов определяют количество неустраненных дефектов и достижимую корректность

компонентов и ПС в целом. На практике учитывать степень покрытия тестами достаточно трудоемко и зачастую желательно иметь более простой способ регламентирования и оценивания качества тестирования. Возникает проблема, как практически учесть ограничения ресурсов и ***когда прекратить верификацию и тестирование***, а также какая при этом будет достигнута ***корректность программ*** и способна ли она удовлетворить заказчика и пользователя при эксплуатации ПС.

При разработке сложных ПС верификация и тестирование требуют значительных ресурсов в течение всего ЖЦ ПС, и наиболее критичным ресурсом является допустимое время поэтапного выполнения этих процедур. Интенсивность выявления дефектов при тестировании программ практически экспоненциально убывает в зависимости от времени, затрачиваемого на их обнаружение, и соответственно возрастают интервалы между очередными проявлениями дефектов. На основе экспериментальных данных ***созданы математические модели***, которые позволяют прогнозировать интервалы времени между последовательными обнаружениями дефектов или ошибок в программных компонентах при некотором методе и этапе верификации или тестирования (см. лекцию 10). Например, если при тестировании определенного модуля или компонента ПС выявлено определенное число дефектов (например, 5 или 10), то модели позволяют оценить ресурсы (время), необходимое для обнаружения следующего дефекта, и рентабельность продолжения тестирования используемым методом. Рациональное изменение стратегии или метода выявления дефектов может приводить к кратковременному повышению интенсивности их проявления, а затем к постепенному ее снижению.

Для использования таких моделей в конкретной фирме для некоторого класса проектов при определенной квалификации разработчиков экспериментально может быть установлено число дефектов, которое должно быть обнаружено тестировщиками на определенном этапе тестирования программного компонента за выделяемое время. Если за это время выявлено число дефектов меньшее, чем задано априори, то это может означать либо очень хорошую работу программистов-разработчиков, либо недостаточное качество тестов и их исполнителей. Некоторое дополнительное тестирование, возможно, поможет решить эту альтернативу. Регулярная регистрация числа выявляемых дефектов за заданное время определенными тестировщиками, в результатах работы конкретных программистов,

позволит оценить усредненную ***интенсивность выявления дефектов при тестировании*** на предприятии при разработке определенных типов проектов. Такие оценки могут использоваться при прогнозирования достигнутого качества соответствующих программных компонентов. При этом целесообразно учитывать категории выявленных дефектов по степени влияния на результаты функционирования программы: критические — недопустимо искажающие результаты, опасные — угрожающие небольшими искажениями результатов в редких случаях, а также слабо или практически не влияющие на результаты.

Современные ***системы систематического тестирования и отладки программных компонентов*** высокого и контролируемого качества должны обеспечивать:

— удобный, дружественный диалог в символьном и графическом видах пользователей со средствами автоматизации и, в основном, безбумажное тестирование программ;

— использование достаточно развитой и эффективной базы данных проектирования (репозитория) для накопления и хранения разнообразной информации о разрабатываемых программах, их версиях, планах тестирования, тестовых и эталонных данных, выполненных корректировках;

— автоматическое обнаружение статическими методами типовых ошибок в исходных текстах программ, обусловленных нарушениями формализованных правил семантики программирования, структурного построения модулей и использования данных;

— автоматизированное планирование тестирования, выдачу рекомендаций пользователям по систематическому применению методов, стратегий и средств динамической отладки;

— эффективную реализацию отладочных заданий с целью достижения максимальной корректности программ в условиях ограниченных ресурсов на тестирование;

— оценку достигнутой корректности программ по выбранным критериям тестирования и определение основных показателей качества созданных программных компонентов;

— автоматизированную регистрацию и документирование всех выполняемых изменений в программах и учет версий программных модулей и групп программ, в которых проведены корректировки.

Для отладки программных компонентов, входящих в сложные ПС и пригодных для повторного использования в различных проектах, необходим комплекс средств автоматизации, использующий основные современные методы выявления ошибок и дефектов в программах. Эти средства можно разделить на (рис. 13.4):

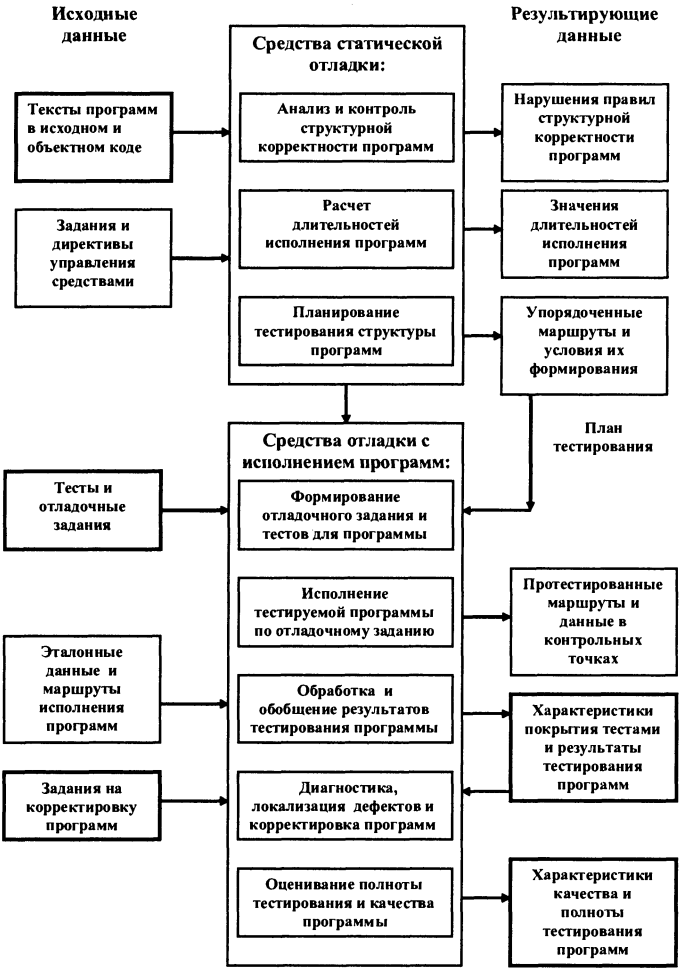

Рис. 13.4

— ***статические*** — анализирующие спецификации и исходные тексты программ без их исполнения в объектном коде;

— ***динамические***, при использовании которых программы функционируют в объектном коде и пригодны для их реального применения.

Эти средства объединяются средствами ***интерактивного диалога с пользователями и базой данных проектирования***. В группу ***статической отладки*** входят средства, автоматизирующие структурный анализ и проверку формальной корректности текстов программ и выявление соответствующих типов ошибок. Кроме того, они должны обеспечивать автоматизированное получение данных, поддерживающих выявление ошибок при последующем применении динамических средств. Средства подготовки данных для планирования тестирования должны обеспечивать выделение типовых структур в модуле (циклов, альтернатив, маршрутов исполнения) и их упорядочения по некоторым правилам. Выделение маршрутов исполнения программы и условий их формирования позволяет определить границы областей изменения переменных, влияющие на формирование тестовых наборов и их объем. В результате статического анализа программы автоматически могут создаваться упорядоченные наборы ее характеристик, которые используются для подготовки наборов тестов, а также для контроля полноты проведенного тестирования модулей при динамической отладке.

В группу средств статической отладки входят также средства расчета длительностей исполнения модулей и компонентов. Эти средства позволяют получать ориентировочные значения и распределения длительностей счета программы аналитически, без ее исполнения на ЭВМ. В результате расчетов выявляются компоненты программы, требующие избыточно большого времени счета на реализующей ЭВМ, а также подготавливаются данные для общей ***проверки реализуемости ПС в реальном времени***.

Группу средств ***динамической отладки*** можно разделить на основные средства, непосредственно обеспечивающие исполнение программ в соответствии с отладочными заданиями, и средства, анализирующие выполненное тестирование, его результаты и проведенные корректировки. В основные входят средства:

— трансляции программ, тестов и заданий с языка отладки;

— исполнения программы по отладочному заданию в соответствии с планом и сценарием тестирования;

— регистрации данных о результатах тестирования.

Средства трансляции заданий с языка отладки обеспечивают обработку и подготовку к исполнению тестов и сценария отладочного задания. В задании указывается тестируемая программа, контролируемые и регистрируемые переменные и состояния программы в процессе исполнения. Тестовые значения преобразуются в форму, пригодную для исполнения отлаживаемой программой. Операторы отладочного задания объединяются с отлаживаемой программой или подготавливаются для исполнения в режиме эмуляции или интерпретации.

Средства исполнения программы по отладочному заданию управляют реализацией операторов задания в процессе исполнения отлаживаемой программы. В процессе обработки тестов производится предварительная селекция результатов тестирования в соответствии с указаниями операторов отладочного задания. В некоторых случаях получаемые результаты могут автоматизировано сравниваться с эталонными значениями для выявления ошибок.

Средства регистрации и обобщения данных о результатах тестирования осуществляют преобразование зафиксированных данных функционирования отлаживаемой программы на язык отладки для диалогового взаимодействия с пользователем. Для сокращения избыточной информации производятся редактирование и селекция результатов исполнения операторов отладочного задания. Отображение результатов тестирования в мнемонической и графической формах может обеспечиваться унифицированными средствами интерактивного взаимодействия пользователей с ЭВМ.

Для каждого отлаженного программного компонента должно обеспечиваться ***хранение основных тестов и отладочных заданий***, с использованием которых проведено тестирование. Это позволяет поддерживать высокое качество компонентов и при необходимости вносить в них корректировки. При проведении таких изменений соответственно расширяется и модифицируется состав применявшихся тестов и заданий. Вместе с тестами целесообразно хранить результаты оценки полноты тестирования и достигнутой корректности программного компонента. Эти данные вместе с оценкой сложности компонента, структурными и информационными характеристиками могут объединяться в паспорт аттестации компонента.

Для поддержки конфигурационного управления программными компонентами может быть полезным хранение на некотором интервале вре-

мени проведенных изменений и выявленных ошибок (см. лекцию 16). Эти данные могут использоваться для прогнозирования процессов отладки программ и сроков достижения заданного их качества. Они позволяют выявлять статистические характеристики ошибок и квалифицированно управлять процессом модификации программ.

Планирование тестирования — процесс творческий, и средства автоматизации предназначены в основном для подготовки исходных данных, используемых при планировании. Эту информацию можно разделить на ***три группы данных***:

— о циклах в программе;

— о маршрутах исполнения программы;

— о предикатах, определяющих маршруты исполнения программы и границы областей изменения переменных.

Для подготовки этих данных используются текст программы на языке программирования и взвешенная графовая модель программы. Эта модель может подготавливаться специально средствами автоматизации планирования тестирования или средствами контроля структуры и определения длительности исполнения программы. Основная часть данных для планирования тестирования может быть получена автоматическим расчетом по тексту программы с использованием указаний разработчика о критерии выделения и стратегии упорядочения маршрутов. Диалог разработчика со средствами автоматизации целесообразен для уточнения критерия и стратегии образования маршрутов в зависимости от сложности тестирования, а также для исключения циклов и ациклических маршрутов, которые не реализуются по сочетаниям условий в предикатах. Диалог может также быть полезным при подготовке взвешенной графовой модели программы, когда необходимо ввести оценки значений вероятностей ветвления в вершинах графа и характеристики итераций циклов.

В программе, прежде всего, автоматически должны ***выделяться циклы***, подлежащие тестированию. Для этого используются указания разработчика о стратегии выделения маршрутов при тестировании циклов. Кроме того, следует вводить указания о количестве итераций циклов и их связях с маршрутами исполнения циклов. В результате разработчику отображаются данные о маршрутах в циклах, которые подлежат тестированию по выбранной стратегии. По данным о циклах, выделенных для автономного тестирования, рассчитываются ***суммарное число тестов в плане*** и сум-

марная сложность тестирования циклов. Выделение циклов и маршрутов в них позволяет преобразовать программу к ациклическому виду.

Для ***выделения тестируемых маршрутов*** в такой ациклической программе разработчик должен указать ***критерий***, по которому следует формировать маршруты. Кроме того, разработчик указывает ***стратегию*** для составления упорядоченного списка маршрутов, по которому надлежит планировать последовательность тестирования. Упорядочение маршрутов производится по длительности их исполнения или по вероятности реализации при случайных данных на входе программы. Если ряд маршрутов может быть нереализуемым по сочетаниям условий в вершинах графа программы, то такие маршруты следует исключать из последующего анализа.

В результате составляется список маршрутов, упорядоченных по выбранной стратегии. По этим маршрутам рассчитываются полное число тестов и суммарная сложность тестирования структуры программы в соответствии с выбранным критерием выделения маршрутов. Корректировка планов возможна за счет изменения критерия выделения маршрутов и за счет ограничения числа выделенных для тестирования маршрутов из общего упорядоченного множества. Выделенные для тестирования маршруты могут дополняться данными о значениях переменных в предикатах условий на каждом маршруте. Для этого используются текст программы на языке программирования и описание переменных. По полученным соотношениям между переменными в предикатах условий могут быть построены ***границы областей*** изменения переменных для каждого из маршрутов и для программы в целом (см. п. 13.5). По числу границ областей изменения переменных осуществляется оценка числа тестов, необходимых для проверки процессов обработки данных в анализируемой программе.

Характеристики сложности тестирования областей в совокупности с характеристиками сложности тестирования структуры программы и циклов позволяют оценить ***реализуемость плана тестирования*** конкретного программного модуля или компонента. Кроме того, рассматриваемые средства представляют разработчику достаточно полные сведения о циклах, маршрутах и переменных, которые необходимо учитывать при планировании тестирования. Автоматический расчет и упорядочение информации о характеристиках программы, а также отображение этих сведений

в компактной и наглядной форме позволяют сделать процесс тестирования эффективным и экономичным.

Документирование процессов тестирования программных компонентов следует проводить непосредственно при выполнении каждой из соответствующих работ с определенными целями. Должны быть описаны и документированы, упорядочены и конкретизированы весь технологический процесс тестирования и частные результаты на каждом этапе ЖЦ ПС:

— исходные данные: технические задания и спецификации требований; комплекс эталонных значений; критерии качества результатов тестирования; ограничения доступных ресурсов для реализации тестирования;

— совокупность выбранных методов, стратегий и сценариев тестирования при реализации этапов и процессов плана;

— план тестирования компонентов;

— методы и критерии оценки достигнутого качества программных компонентов;

— входные и результирующие данные тестирования;

— графики организации решения частных задач тестирования и необходимые для них ресурсы;

— распределение ответственности между специалистами по компонентам и видам проверок;

— протоколы результатов тестирования и обобщенные отчеты о достигнутом качестве программных компонентов.

13.3. Технологические этапы и стратегии систематического тестирования программ

Исходным эталоном для тестирования любой программы являются требования технического задания и/или спецификации требований заказчика и потенциального пользователя, предъявляемые к создаваемым программам. Подобные документы должны устанавливать состав, содержание и значения результатов, которые должен получать пользователь при определенных условиях и исходных данных. Любое отклонение результатов функционирования программы от предъявляемых к ней требований и сформированных по ним эталонов следует квалифицировать как ошибку или дефект в программе.

Эталонные значения параметров и характеристик результатов тестирования для ***вычислительных программ*** получать относительно проще, прежде всего потому, что требуется значительно меньшее количество контрольных величин. Промежуточные значения результатов можно определять интерполяцией или вообще пропускать, опираясь на предположение о гладкости вычисляемых функций, т.е. необязательна подготовка эталонов при абсолютно всех комбинациях исходных данных. При том же объеме тестируемой программы подготовка эталонных значений для ***логических программ*** усложняется, прежде всего, вследствие их комбинаторного характера и увеличения необходимого числа тестовых значений. В этом случае тесты, в принципе, необходимо рассчитывать для всех возможных сочетаний исходных данных, так как каждое из них может полностью изменять область определения и смысловое значение результирующих величин.

На математических моделях или прототипах, реализуемых на ЭВМ, наиболее эффективно получать эталонные значения и требуемые характеристики функционирования сложных программ. Возможно создание моделей двух типов: на базе более сложных и более точных алгоритмов, которые, например, не могут быть реализованы на объектной ЭВМ вследствие ограниченных ресурсов, и на базе упрощенных, обобщенных моделей для более быстрого получения эталонных значений. Модели второго типа, естественно, характеризуются меньшей потенциальной точностью результатов, однако, благодаря снижению сложности, в них менее вероятны ошибки.

Результаты функционирования реальных программ, являющихся предшественниками-прототипами или пилотными проектами разрабатываемой версии ПС, для использования в качестве эталонов при тестировании требуют обеспечения идентичности и повторяемости исходных данных, используемых проверяемыми программами при получении эталонных значений. На завершающих стадиях создания ПС формализация эталонных значений в ряде случаев не доводится до конца и в качестве эталона выступает неформализованное представление разработчика или заказчика. Во всех случаях целесообразно фиксировать и сохранять значения используемые в качестве тестовых эталонов для обеспечения повторяемости сеансов тестирования.

При сравнении результатов функционирования проверяемых программ на соответствие эталонам следует использовать ***критерии оценки допустимых отклонений от эталонов и решений о степени корректности*** программы. Величина допусков зависит от типа проверяемого алгоритма, метода и этапа проверки корректности программ. Для простых логических алгоритмов, реализующих некоторые схемы принятия решений, при анализе результатов тестирования обычно используется детерминированный критерий абсолютной идентичности проверяемых и эталонных решений при одинаковых исходных данных. В вычислительных алгоритмах и при проверке сложных логических алгоритмов сравнение с эталоном приходится осуществлять статистически.

Выделение и стратегия упорядочения компонентов для тестирования в крупных ПС зависит от их архитектуры и реального состава готовых к использованию компонентов. При ***нисходящем тестировании от требований*** оно начинается с программ организации вычислительного процесса. Первоначально тестируются управляющее ядро комплекса программ и программы решения функциональных задач, размещенные на высших иерархических уровнях, на соответствие исходным требованиям технического задания. К ним последовательно, по мере готовности, подключаются компоненты более низких иерархических уровней. Такая стратегия ***сверху вниз*** эффективна, когда имеется достаточно полный набор готовых апробированных программных компонентов и/или модулей, ранее отработанных в версиях подобных или пилотных программных комплексов.

Если некоторые программы нижних уровней не разработаны или недостаточно протестированы, то вместо них временно могут подключаться программные имитаторы — «заглушки». В результате при тестировании на начальных этапах проверяются модели функциональных групп программ или комплекса с некоторым числом имитаторов программных компонентов. Преимуществом такой стратегии тестирования является сохранение и последовательное развитие тестовых исходных данных по мере подключения компонентов. Однако тестирование групп программ с заглушками может требовать больших затрат на обнаружение простейших ошибок во вновь разработанных и подключаемых модулях, если они до этого автономно недостаточно тестировались.

При систематическом ***восходящем тестировании,*** прежде всего, проверяются программные компоненты и/или модули нижних иерархических

уровней в функциональной группе программ, к которым последовательно подключаются вызывающие их модули. В этих модулях тестирование также начинается с простейших конструкций, переменных и маршрутов обработки информации. Соответственно последовательно усложняются используемые методы тестирования и типы выявляемых при этом ошибок. Последовательное наращивание компонентов в комплексе программ ***снизу вверх*** позволяет проверять работоспособность таких групп в их естественном исполнении, без подмены и имитации компонентов нижних уровней. Основные трудности при такой стратегии состоят в необходимости непрерывного обновления и увеличения числа тестовых наборов по мере подключения каждого нового компонента более высокого уровня. Одновременно углубляется тестирование компонентов нижних иерархических уровней, что способствует систематическому повышению их качества.

Нисходящее и восходящее тестирование отличаются не только схемой анализа модулей или небольших компонентов, но также принципиальными целями всего процесса тестирования крупных комплексов программ. Основная ***цель нисходящего тестирования*** — верифицировать требования к модулям и программным компонентам, а также добиться их качества при автономном тестировании каждого из них вне реального времени. Тем самым в процессе ***декомпозиции*** должно быть обеспечено требуемое качество компонентов и их соответствие исходным требованиям. ***При восходящем тестировании главная задача*** — обеспечить укрупнение, ***интеграцию*** и корректное взаимодействие всех компонентов для полного решения задач требуемым комплексом программ. При этом предполагается достаточно высокое качество подготовленных ранее компонентов. Представленные в данной главе фрагменты этих базовых процессов тестирования схематично объединены на рис. 13.5. Подобную схему полезно иметь в виду при планировании стратегии тестирования крупных ПС.

С учетом особенностей применения методов и технологических этапов ЖЦ программных компонентов ниже в данном разделе последовательно рассматриваются задачи ***восходящего тестирования следующих объектов***:

— формализованных спецификаций требований на программные и информационные модули, на группы программ и на программные комплексы;

— программных модулей, запрограммированных и подготовленных к тестированию на уровне исходных текстов программ и на уровне объектных кодов реализующей ЭВМ;

— автономных групп программных модулей и компонентов, решающих законченные функциональные задачи;

— функциональных компонентов в составе программных средств.

Рис. 13.5

Задача ***тестирования спецификаций*** состоит в проверке полноты и взаимного соответствия функций, предписываемых программным и информационным компонентам требованиями разных иерархических уровней (см. п. 13.1). Кроме того, задачи тестирования включают проверку соответствия описаний информации на входах и выходах взаимодействующих программных модулей и групп программ, а также с описаниями информационных модулей в базе данных. В результате тестирования спе-

цификаций должна быть обеспечена их корректность и согласованность в пределах обобщенного описания требований к функциям всего ПС и взаимодействия всех его составных частей. Тестирование взаимосвязей целесообразно проводить, начиная от спецификации требований комплекса или группы программ. Последовательно по иерархическим уровням должно прослеживаться обеспечение программ верхнего уровня реализованными функциями программ нижних уровней, предписанными программными спецификациями. Одновременно проверяется полнота выполнения этих функций спецификациями информационных модулей (см. рис. 13.2).

Процесс ***тестирования программных модулей*** состоит в проверке корректности обработки модулями поступающей информации и получающихся на выходе данных в соответствии с функциями, представленными в спецификациях требований. Должна быть проверена корректность структуры модулей и примененных конструктивных элементов: циклов, блоков, переключателей и т.д. Так как на этом этапе тестирования участвует наибольшее число специалистов, зачастую не очень высокой квалификации, особое значение приобретают методики тестирования и регламентирование применения средств автоматизации.

Проверке подлежат маршруты обработки информации в каждом модуле и правильность их реализации в зависимости от исходных данных. Полнота теста определяется критериями выделения маршрутов для тестирования и степенью покрытия тестами требований спецификаций и возможных маршрутов исполнения программы. На каждом выделенном маршруте должна проверяться корректность выполняемых вычислений при некоторых фиксированных исходных данных. При этом выявляются ошибки неполного состава или некорректности условий при реализации частных маршрутов обработки данных, а также некоторые ошибки преобразования переменных. Для каждого выделенного маршрута по тексту программы формируется набор условий, определяющих его реализацию и используемый при создании соответствующего теста. Такое представление маршрутов позволяет упорядоченно контролировать достигнутый уровень проверки маршрутов и в некоторой степени предохраняет от случайного пропуска отдельных нетестировавшихся маршрутов.

Автономное тестирование функциональных компонентов с исполнением программ предназначено для проверки корректности решения отдельных достаточно крупных функциональных задач. На этом этапе про-

веряется корректность управляющих и информационных связей между группами модулей, а также корректность реализации требований в процессе обработки информации в группе программ. При этом значительно возрастает сложность тестируемых объектов и соответственно — размеры и сложность тестов. Вследствие этого возрастают требования к автоматизации тестирования и затраты на его выполнение. Детерминированным тестированием должны проверяться структура группы программ и основные маршруты обработки информации. В ряде случаев результаты следует получать методами стохастического тестирования.

Интеграционное тестирование составляет объединение программного кода, соответствующего двум или большему количеству программных модулей, и тестирование полученного в результате кода. Это должно гарантировать, что вместе они работают, как требуется, до полной интеграции и тестирования кода каждого функционального компонента. Так как отдельные модули могут включать другие модули, некоторая часть интеграции и тестирования модулей может происходить в процессе модульного тестирования. Тестовые варианты должны покрывать все требования проекта уровня функциональных компонентов ПС. После этого следует выполнять все необходимые изменения ПС, связанные с коррекцией дефектов, выявленных в процессе верификации, а также повторное тестирование в необходимом объеме и модифицировать файлы разработки ПС и другие программные продукты, основываясь на результатах интеграционного тестирования.

Тестирование функциональных компонентов в составе программных средств в процессе разработки комплексов программ и оценки полноты тестирования осуществляются преимущественно по степени выполнения требуемых функций и по характеристикам достигаемой корректности и качества функционирования ПС в целом (см. рис. 13.2). Значительную помощь в повышении качества сложных, критических ПС может оказать систематизация видов тестирования и упорядоченное их проведение при разработке. Эти виды тестирования должны быть ориентированы на ***дифференцированное*** выявление определенных классов дефектов. Для каждого вида тестирования целесообразно разрабатывать методику его выполнения с указанием проверяемых компонентов, контролируемых параметров, ожидаемых и эталонных результатов. Кроме того, при заключительных испытаниях или сертификации должно проводиться ***интегральное*** тести-

рование при максимально широком варьировании тестов в условиях, соответствующих нормальной и форсированной эксплуатации.

Тестирование полноты решения функциональных задач при типовых исходных данных предназначено для обнаружения дефектов функционирования в нормальных, штатных условиях, определенных требованиями технического задания на базовую версию ПС. Первичным эталоном являются цели и задачи создания программного продукта. Некоторая часть тестов может содержать детерминированные исходные данные, для анализа которых часто применяются различные системы графического отображения. Особое внимание целесообразно обращать на варианты тестов, позволивших обнаружить ошибки. Для этих условий следует проводить дополнительное тестирование.

Тестирование функционирования программ ***в критических ситуациях*** по условиям и логике решения задач (***стрессовое тестирование отказоустойчивости***) проводится при исполнении программ в нештатных ситуациях, которые редко реализуются, но важны для обеспечения качества и надежности функционирования системы обработки информации. Для разработки таких тестов создаются сценарии критических сочетаний значений исходных данных и условий решения задач, при которых необходимо проверить функционирование программ и можно ожидать искажения результатов и отказы. Такие стрессовые, нештатные сочетания подготавливаются вручную или предусматривается их реализация в составе данных имитаторов стохастических тестов в реальном времени. Особая важность проверки в критических ситуациях определяется опасностью проявления таких ошибок при функционировании ПС в реальных условиях. Поэтому при тестировании активно применяются имитаторы внешней среды, автоматически подготавливающие исходные данные, и средства контроля, реагирующие на аномальные результаты исполнения тестируемых программ.

Основные этапы систематического тестирования и испытаний крупного комплекса программ реального времени и его компонентов представлены на рис. 13.6. При тестировании и испытаниях ***корректности функциональных компонентов комплексов программ выделены этапы***:

— комплексирование модулей и тестирование автономных функциональных групп программ в статике без взаимодействия с другими компонентами и, возможно, без подключения к операционной системе реального времени;

— тестирование функциональных групп программ в статике с учетом взаимодействия с некоторыми другими важнейшими компонентами и с базой данных;

— тестирование отдельных программных компонентов в реальном времени во взаимодействии с другими функциональными компонентами и с основными компонентами операционной системы и базы данных.

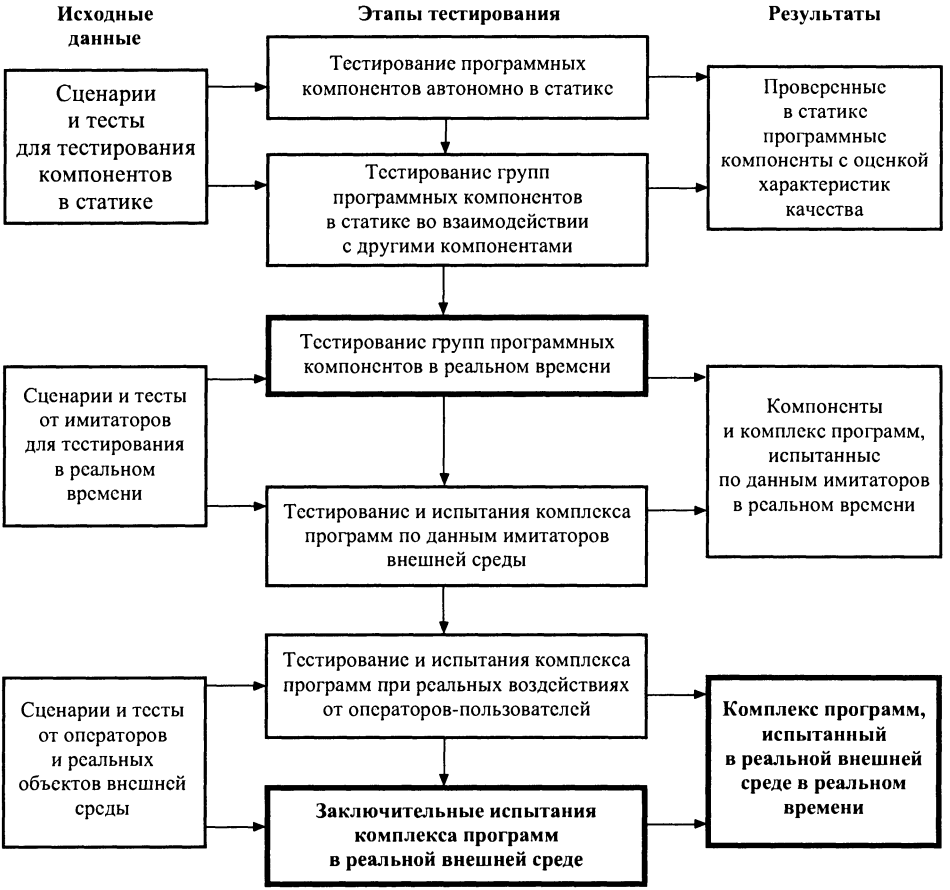

Рис. 13.6

Сложность тестирования компонентов на этих этапах в значительной степени обусловлена несинхронным процессом их разработки и отладки отдельными специалистами в коллективах. Первично спланированная логика сопряжения между собой отдельных компонентов и подключения их к операционной системе не всегда выполняется из-за задержек в разработке и автономной отладке некоторых из них. Целесообразная последовательность тестирования определенных компонентов может нарушаться неготовностью к сопряжению с ними других взаимодействующих программ.

Для обеспечения имитации объектов внешней среды и других взаимодействующих групп программ на этих этапах используются преимущественно детерминированные контрольные задачи или частные генераторы соответствующих тестов. Эти генераторы тестов целесообразно разрабатывать и оформлять как отдельные модули или группы программ, функционирующие на той же технологической ЭВМ и в той же операционной среде, что и отлаживаемые компоненты. Совместно с ними также реализуются и функционируют частные специализированные программы для обработки и обобщения отдельных результатов тестирования соответствующих групп программ.

После комплексирования основных, функциональных компонентов начинаются их тестирование и испытания в составе ПС в целом. Для них наиболее характерны следующие ***этапы квалификационного тестирования и испытаний ПС в реальном времени*** (см. рис. 13.6):

— по данным моделирующего стенда или генераторов тестов, имитирующих отдельные объекты внешней среды;

— с имитаторами отдельных объектов внешней среды и с реальными воздействиями от операторов-пользователей;

— в полностью адекватной реальной или имитированной внешней среде и с реальными воздействиями от операторов-пользователей (см. лекцию 14).

На всех этапах тестирования, кроме операций непосредственной проверки функционирования программ, можно выделить еще две важные группы работ. ***Первая группа*** — это работы по методическому обеспечению процессов тестирования и по созданию средств автоматизированной генерации тестов. ***Вторая группа*** работ должна обеспечивать возможность обработки результатов тестирования и корректной оценки достигнутых характеристик качества функционирования программ.

Средства генерации тестов и обработки результатов тестирования можно разделить на три вида (см. рис. 13.6). Одни и те же средства автоматизации тестирования в статике обычно обеспечивают отладку групп программ как автономно, так и во взаимодействии с другими компонентами. Средства, имитирующие внешнюю среду в реальном времени, чаще всего ориентированы на тестирование как функциональных компонентов, так и ПС в целом. Еще один вид генераторов тестов в той или иной степени использует реальные объекты внешней среды. Первоначально такими объектами являются имитирующие стенды с участием реального функционирования операторов-пользователей (см. лекцию 14). Затем источниками тестов могут быть комплексы реальной аппаратуры внешних объектов или их аппаратурные аналоги.

Рассмотренная схема ориентирована на тестирование и испытания комплекса программ, размещаемого на единой аппаратной платформе. При создании ***распределенных систем клиент-сервер*** возникают дополнительные, весьма сложные задачи тестирования. Программные средства, размещенные на аппаратных платформах клиентов и серверов, необходимо первоначально тестировать автономно по представленной выше полной программе, а затем дополнительно их взаимодействие. Это взаимодействие клиентской и серверной частей программного комплекса должно быть подготовлено автономным тестированием всей системы телекоммуникации и гарантированием ее качества. После этого следует повторить последние три этапа комплексного тестирования в реальном времени, представленные на рис. 13.6, но уже при полном взаимодействии обоих функциональных компонентов системы клиент-сервер.

13.4. Процессы тестирования структуры программных компонентов

Как отмечалось выше (см. п. 13.1), оценивание корректности программных средств можно представить двумя видами работ:

— верификацией — последовательным ***прослеживанием сверху вниз*** реализации требований к системе и ПС программными компонентами нижних уровней;

— определением полноты ***покрытия тестами*** их структуры и проверками выполнения исходных требований к ПС и его компонентам.

Покрытие тестами может оцениваться по степени охвата тестированием ***набора требований*** к программе или по ***покрытию тестами структуры*** программы. Прослеживание и покрытие тестами набора требований к программам трудно формализовать и оценить их влияние на достигаемую корректность ПС. Имеющийся опыт показывает, что такой анализ вполне доступен для неформального анализа, но относительно слабее влияет на корректность, чем недостаточное тестирование структуры программ. Поэтому ниже внимание сосредоточено на оценивании тестирования и корректности структурного покрытия программ.

Анализ и оценивание покрытия тестами структуры программ позволяет выявить дефекты и ошибки, угрожающие наиболее тяжелыми последствиями при функционировании ПС. Ограниченные ресурсы трудоемкости и времени на тестирование сложных комплексов программ для обеспечения их максимальной корректности приводят к необходимости рационального использования доступных ресурсов, прежде всего, для устранения самых опасных ошибок. Тестирование фрагментов структуры программы не гарантирует полное отсутствие ошибок в них, однако существенно снижает их вероятность. Пропущенные при тестировании фрагменты структуры заведомо могут содержать невыявленные ошибки, которые негативно отражаются на корректности программ. Таким образом, тестирование структурного покрытия программ является ***необходимым*** условием обеспечения их относительной корректности, однако оно ***недостаточно*** для полного обеспечения корректности их функционирования. В то же время тестирование и покрытие тестами структуры программ наиболее доступно формализации для оценивания достигаемой относительной корректности и вероятности отсутствия ошибок в программе и позволяет устранять наиболее опасные ошибки, угрожающие отсутствием или полным искажением требуемых результатов на выходе ПС.

На практике при отсутствии упорядоченного анализа потоков управления некоторые маршруты в программе оказываются пропущенными при тестировании (до 50%), поэтому первая задача, которая должна решаться при тестировании структуры программ, — это получение информации о ***полной совокупности реальных маршрутов*** исполнения в каждой программе и ее структурной сложности. Такое представление покрытия программы позволяет упорядоченно контролировать достигнутую проверку

маршрутов и, в некоторой степени, предохраняет от случайного пропуска отдельных нетестировавшихся маршрутов и их элементов.

Полное структурное покрытие программы тестами определяется числом взаимодействующих компонентов, числом связей между компонентами и сложностью их взаимодействия. Структурная сложность и корректность программного модуля зависит не столько от размера программы (числа строк текста), сколько от числа отдельных путей-маршрутов ее исполнения, существующих в программе. В пределе для обеспечения полной структурной корректности все маршруты возможной обработки данных должны быть проверены при создании программы, и тем самым определяют сложность ее тестирования.

В ряде случаев подтверждена достаточно высокая адекватность использования структурной сложности программ для ***оценки трудоемкости тестирования, вероятности невыявленных ошибок и затрат на разработку программных модулей и компонентов в целом***. Сложность тестирования структуры программных модулей можно оценивать по числу маршрутов, необходимых для их проверки, или более полно — по суммарному числу условий, которое необходимо задать в тестах для исполнения всех маршрутов программы. Анализ критериев тестирования, тестовых покрытий и выделение тестируемых маршрутов удобно проводить, используя графовые модели программ. При планировании тестирования структуры программ возникают, прежде всего, ***две задачи***: формирование критериев выделения маршрутов для тестирования и выбор стратегий упорядочения выделенных маршрутов.

Критерии выделения маршрутов для тестирования соответствуют критериям определения структурной сложности программных модулей. В основном используются следующие критерии:

X_1 — покрытие графа программы минимальным количеством маршрутов, охватывающих каждую дугу графа хотя бы один раз;

X_2 — выделение всех линейно независимых маршрутов, отличающихся хотя бы одной дугой в маршруте от остальных;

X_3 — выделение маршрутов при всех возможных комбинациях дуг, входящих в маршруты.

Планировать тестирование можно по одному из критериев или используя последовательно более жесткие критерии выделения маршрутов, при которых возрастают соответственно объем и сложность тестирования.

К значительному возрастанию числа маршрутов обычно приводят циклы в программах.

Стратегии упорядочения маршрутов. Показатель важности маршрута для тестирования программы и для оценивания ее корректности может учитывать сложность маршрута и тестов для его проверки: число операторов, условных переходов и циклов в маршруте; частость его исполнения при рабочем функционировании ПС; сложность получения соответствующих эталонных данных. В первую очередь для установления корректности программы целесообразно производить проверку основной группы маршрутов с экстремальными значениями выбранного показателя сложности в пределах ресурсов, выделенных для тестирования. При имеющихся ограничениях ресурсов некоторая часть маршрутов может оказаться непроверенной и характеризует ***достигнутую корректность*** данной программы по выбранному критерию.

Упорядочение маршрутов при планировании тестирования базируется на использовании в основном ***трех характеристик*** программных модулей:

— стратегия 1 учитывает число строк текста программы в выделенных маршрутах или расчетную длительность их исполнения при функционировании программы;

— стратегия 2 анализирует число альтернатив или условных переходов, определяющих образование каждого маршрута;

— стратегия 3 базируется на использовании вероятности исполнения маршрутов при реальном функционировании программы.

Эти стратегии тестирования позволяют сосредоточивать внимание разработчика на анализе наиболее важных для корректности компонентах программ. При ***стратегии 1*** первичному тестированию подлежат маршруты, наиболее длинные по числу строк и/или по времени исполнения. Им соответствуют обычно маршруты с наибольшим объемом вычислений и преобразований переменных. Эта стратегия целесообразна при планировании тестирования программ, имеющих вычислительный характер обработки данных при небольшом числе логических условий и маршрутов исполнения программ.

При ***стратегии 2*** приоритет отдается маршрутам, наиболее сложным по числу анализируемых условий. Такая стратегия предпочтительна при тестировании логических программ с небольшим объемом вычисле-

ний. При обеих стратегиях на завершающие этапы тестирования остаются простые по вычислениям или по логике маршруты, которые отражают потенциальную, структурную некорректность программы. Это соответствует традиционной стратегии многих разработчиков программ подготавливать вначале тесты с возможно большим охватом вычислительных или логических компонентов программы и скорейшим достижением по возможности высокого уровня их корректности.

При упорядочении маршрутов по ***стратегии 3*** основная сложность состоит в оценке и учете вероятностей ветвления в условных переходах и переключателях, а также числа исполнения циклов. Их значения должны указываться разработчиками программ, что достаточно трудоемко и субъективно. Тем не менее такая стратегия позволяет наиболее детально планировать тестирование и оценивать предельный уровень корректности программ.

Эффективность тестирования определяется полнотой проверки программного модуля или вероятностью наличия невыявленных ошибок ***в зависимости от затрат ресурсов:*** на создание тестов, исполнение программ и анализ результатов тестирования. Затраты в значительной степени зависят от суммарной сложности формирования тестов, проверяющих маршруты исполнения программы. На каждой дуге графа программы между условными переходами производятся вычисления и преобразования переменных, объем которых может изменяться в широких пределах. Для упрощения анализа и оценивания тестирования структуры программ предположим, что длительность и сложность вычислений на дугах графов программ одинаковы и относительно невелики. Некоторые вершины графа программы могут образовываться в результате схождения дуг без последующего ветвления. Такие вершины не влияют на число маршрутов, и их можно обобщать с ближайшей последующей вершиной, в которой происходит ветвление. При этих предположениях сложность теста, проверяющего каждый i-й маршрут, в первом приближении пропорциональна числу дуг графа программы, входящих в этот маршрут, или числу E_i условий, которые необходимо задать в тесте.

Экспериментально подтверждена ***адекватность использования структурной сложности программ для оценки трудоемкости тестирования***, а также вероятности наличия невыявленных ошибок и затрат на разработку программных модулей в целом. Сложность тестирования ПМ

можно оценивать по числу маршрутов M_X, необходимых для их проверки, или более полно по суммарному числу условий E_X, которое необходимо задать в тестах для прохождения всех маршрутов программы, выделенных по X-му критерию:

$$E_x = \sum_{i=1}^{M_X} E_i. \qquad (13.1)$$

где E_i — число условий-предикатов, определяющих i-й маршрут.

Маршруты исполнения программного модуля можно разделить на ***два вида***:

— маршруты исполнения преимущественно вычислительной части программы и преобразования непрерывных переменных;

— маршруты принятия логических решений и преобразования логических переменных.

Маршруты первого вида обычно логически проще и короче, чем второго, и предназначены для преобразования величин, являющихся квантованными результатами измерения некоторых непрерывных физических характеристик (непрерывные переменные). Значения таких переменных связаны условиями гладкости, т.е. условиями малых изменений производных этих переменных по времени или по другим параметрам. При оценке сложности вычислительных маршрутов программ необходим учет числа операндов, участвующих в вычислениях. Кроме того, исходные и результирующие данные при тестировании должны принимать несколько значений. Во всем диапазоне исходных переменных следует выбирать несколько характерных точек (предельные значения и несколько промежуточных), при которых проверяется программа. В особых точках значений и сочетаний переменных и в точках разрыва функции необходимо планировать дополнительные проверки. Таким образом, сложность проверки программного модуля будет определяться числом маршрутов M_X исполнения программы и числом обрабатываемых операндов l_i на каждом i-м маршруте, умноженном на число значений s_{ij} для каждой исходной j-й величины на этом маршруте:

$$S_X = \sum_{i=1}^{M_X} \sum_{j=1}^{l_i} s_{ij}. \qquad (13.2)$$

Расчет показателя сложности тестирования программного модуля по такой схеме имеет значительную неопределенность из-за произвола в выборе числа значений s_{ij} (в основном особых точек) при варьировании исходных данных в тесте. В то же время доля вычислительной части во многих сложных ПС относительно невелика.

Второй вид маршрутов является результатом функционирования схем принятия решений и преобразования логических переменных. Для логических переменных отсутствует сильная корреляционная связь между соседними значениями, и каждое изменение переменной может определять разные области результирующих значений. Такое преобразование переменных обеспечивается алгоритмами со сложной логической структурой, содержащей ряд проверок логических условий, циклов для поиска и селекции переменных, а также логические преобразования переменных. В результате в программе образуется множество маршрутов обработки исходных данных, которые определяют сложность структуры программы.

Структурная сложность программного модуля может быть рассчитана по числу маршрутов M_X в программе и сложности каждого i-го маршрута E_i. Эти показатели в совокупности определяют минимальную сложность E_X тестов для проверки программного модуля (13.1), а следовательно, трудоемкость его разработки и тестирования и вероятность пропуска логической ошибки в программе. Программные модули являются наиболее массовыми компонентами в ПС и требуют для тестирования суммарно наибольших затрат ресурсов. Затраты на тестирование каждого модуля прямо пропорциональны сложности, которая зависит от его структуры и объема вычислений l_i. При тестировании программного модуля необходимо задать и проанализировать число значений параметров:

$$B_X = \sum_{i=1}^{M_X}\left(\sum_{j=1}^{l_i} s_{ij} + E_i\right). \qquad (13.3)$$

Суммарные затраты ресурсов на тестирование модуля пропорциональны значению его сложности B_X и, с учетом удельных затрат на создание каждого теста — c, определяются выражением:

$$C_X = c\sum_{i=1}^{M_X}\left(\sum_{j=1}^{l_i} s_{ij} + E_i\right). \qquad (13.4)$$

Значение множителя *c* зависит от степени автоматизации процесса тестирования и генерации тестов. В высокоавтоматизированных системах сокращаются затраты ручного труда на подготовку и анализ тестовых данных, однако увеличиваются затраты на машинное время, необходимое для генерации тестов и для автоматической обработки результатов тестирования. В зависимости от этих факторов значения множителя *c* могут различаться в несколько раз, и их следует экспериментально определять для каждой системы автоматизации тестирования.

Сложность тестирования программ, содержащих циклы. Наличие циклов в программе способно резко увеличивать сложность их тестирования. Полное, исчерпывающее тестирование должно охватывать проверку каждого маршрута в цикле при всех возможных итерациях цикла и при всех сочетаниях циклов с маршрутами ациклической части программы. Предположим для простоты, что число маршрутов в нижней ациклической части программы равно $M_3 = 1$. Тогда полное множество маршрутов M состоит из полной совокупности всех маршрутов M_1 в верхней ациклической части программы и группы маршрутов M_2, в которой к каждому маршруту из M_1 присоединено 1..2..3... итерации (витка) цикла. При этом на каждой итерации выполняется, по крайней мере, один из внутренних маршрутов тела цикла.

Например, для графа, имеющего один цикл, требующего исполнения пяти итераций (витков) с тремя внутренними маршрутами, а также содержащего $M_1 = 10$ ациклических маршрутов, проходящих через цикл, суммарное число маршрутов для исчерпывающего тестирования равно $M = (3 \times 5) \times 10 = 150$. При возрастании любого из сомножителей (числа независимых ациклических маршрутов, проходящих через цикл, числа внутренних маршрутов тела цикла или числа его итераций) пропорционально растет их произведение, а следовательно, и сложность тестирования. Поэтому ***исчерпывающее тестирование реальных сложных программ с циклами может быть практически невозможно.***

На сложность тестирования цикла оказывают влияние его структура и два параметра: число маршрутов в теле цикла и число итераций цикла. В динамике реального исполнения простейшего цикла между его итерациями могут существовать зависимости, по крайней мере, ***трех видов***:

— на разных итерациях цикла исполняются независимо все возможные маршруты тела цикла;

— на всех итерациях цикла исполняется один и тот же маршрут тела цикла или некоторая определенная их последовательность;

— на разных итерациях цикла в силу наличия семантических связей исполняется подмножество реализуемых маршрутов тела цикла, зависящее от данных или от номера итерации.

При зависимости ***первого вида***, которая встречается наиболее редко, возникает необходимость в полном переборе всех внутренних маршрутов тела цикла в сочетании с каждым числом итераций. В этом случае сложность тестирования цикла определяется сразу обоими параметрами: числом маршрутов тела цикла и числом итераций и приближается к сложности исчерпывающего тестирования. При зависимости ***второго вида*** число маршрутов тела цикла практически не влияет на сложность цикла. Определяющим становится количество итераций, необходимых для тестирования вычислений в теле цикла (например, с учетом требуемой точности). При ***третьей***, наиболее распространенной зависимости определяющим при оценке сложности тестирования является не число итераций цикла, а число маршрутов тела цикла. Простейшие оценки сложности циклических структур целесообразно проводить в предположении, что порядок реализации маршрутов тела цикла не зависит от номера итерации. В этом случае при оценке сложности циклических структур конструктивным является подход с позиции минимально необходимого числа проверок итераций циклов.

Для оценки ***сложности структурного тестирования ациклических логических программ*** с простейшими циклами целесообразно уточнить критерии проверки. По ***первому критерию*** ациклическая часть программы покрывается минимальным числом маршрутов, в которые входят маршруты, проходящие через цикл, размыкающие его и образующие минимальное покрытие тела цикла. Кроме того, к покрытию добавляется маршрут, содержащий замыкающую дугу цикла. По ***второму критерию*** ациклическая часть программы проверяется количеством тестов, равным сложности ациклической части программы. При этом к каждому такому маршруту присоединяются все примыкающие к нему циклы. Проверка каждого цикла осуществляется одним маршрутом, содержащим столько итераций, какова сложность тела цикла, а тело цикла покрывается линейно независимыми маршрутами.

При таких оценках определяющими факторами являются полнота проверки тела цикла, условий его замыкания и размыкания. При этих критериях сложность цикла наиболее просто оценить, представив его эквивалентным ациклическим подграфом. Например, при втором критерии эквивалентным циклу с одной точкой входа и одной точкой выхода будет линейный подграф, содержащий последовательно соединенные все линейно независимые маршруты тела цикла, связывающие его точку входа с точкой выхода. Такой эквивалентный ациклический подграф добавляется к каждому маршруту в покрытии ациклической части графа по этому критерию.

При применении вложенных циклов со сложной структурой и с большим числом ветвлений в теле цикла сложность тестирования резко возрастает и повышается вероятность сохранения в программе необнаруженных ошибок. Однако циклы с простейшей структурой приводят к относительно небольшому увеличению суммарной сложности тестов. Поэтому в процессе проектирования программ необходимо в максимальной степени упрощать циклические компоненты в структуре, которые во многих случаях определяют достигаемую корректность программных модулей при их тестировании.

Выражение (13.3) целесообразно использовать для выявления сложных модулей, требующих наибольших затрат на тестирование, и для рационального распределения ограниченных ресурсов тестирования модулей при создании крупномасштабных комплексов программ. Некоторые модули могут оказаться недостаточно протестированными из-за их высокой сложности и необходимости больших затрат, которые всегда ограничены. Ограниченность ресурсов на тестирование программных модулей приводит к ***целесообразности выравнивания их структурной сложности*** и выделения затрат на проверку в соответствии со сложностью каждого модуля. Особенно сложные модули следует делить на более мелкие и простые и так перестраивать их структуру, чтобы сокращалось суммарное число маршрутов в каждом отдельном модуле и их длина.

Сложность тестирования программных компонентов (функциональных групп программ) определяется суммарной сложностью модулей и межмодульных связей по управлению и по информации. Каждый модуль должен тестироваться автономно до включения в группу программ и частично в составе группы. Затраты на тестирование модулей в составе

группы программ должны учитывать относительные суммарные затраты на тестирование всех входящих модулей с коэффициентом $d_k < 1$, зависящим от степени предшествующей проверки модуля. Если модули автономно не тестировались (например, при нисходящем тестировании), то $d_k = 1$ и затраты на тестирование каждого модуля войдут в затраты при тестировании группы программ в полном объеме. При тщательном автономном тестировании модулей можно полагать d_k = 0,1—0,01, т.е. в составе ПС затраты на тестирование каждого из модулей составляют несколько процентов.

13.5. Примеры оценок сложности тестирования программ

Приведенные выше выражения можно использовать для априорной оценки числа маршрутов и сложности тестирования (необходимого числа тестов для охвата всех выделенных маршрутов) программных модулей. Для планирования тестирования необходимо выявить зависимости этих характеристик от числа операторов в программе, ее структуры и критериев выделения маршрутов. Эту задачу можно решать экспериментально, путем подсчета соответствующих характеристик в реальных программных модулях, используемых в различных классах ПС. Для выявления общих зависимостей сложности тестирования программ целесообразно выделить типовые программные структуры, а также такие, которые позволяют получить предельные (сверху и снизу) значения этих характеристик.

На рис. 13.7 представлен пример исходного графа модуля программы, содержащего 14 вершин, 20 дуг и 3 цикла. Такая программа сравнительно невысокой сложности содержит около 30—50 операторов на языке высокого уровня и может рассматриваться как достаточно типичная. Для полной проверки модуля ***по первому критерию*** достаточно четырех маршрутов. По этому критерию гарантируется проверка всех передач управления между операторами программы и каждого оператора не менее одного раза. Самый длинный по числу вершин маршрут не охватывает только 3 вершины из 14 и только 6 дуг из 20.

После проверки еще двух маршрутов вне контроля остаются одна вершина и две дуги. Однако при этом критерии не учитывается комбина-

торика сочетания условий на разных участках маршрутов, например, при сочетаниях направлений ветвлений в вершинах 3 и 12. Сложность программы при выделении маршрутов по этому критерию характеризуется числом маршрутов равным четырем и сложностью тестирования равной 20. Эта величина характеризует ***суммарное число условий, которое необходимо задать в тестах для полной проверки*** всех маршрутов, выделенных по первому критерию. Условия в вершинах каждого маршрута могут использоваться для автоматизированного формирования предикатов в соответствующих тестах.

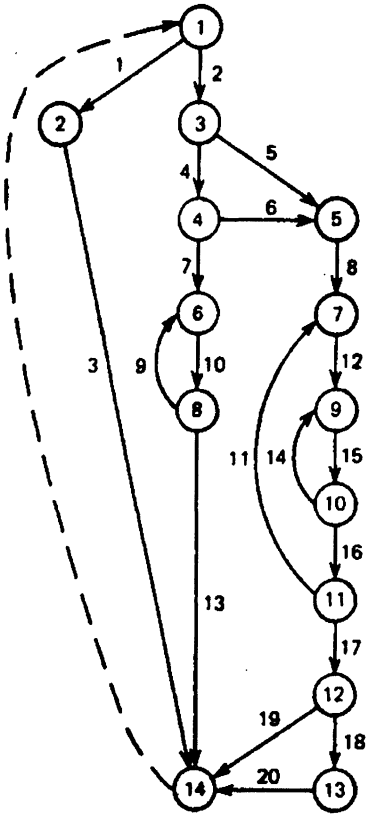

Рис. 13.7

Второй критерий выбора маршрутов при оценке сложности тестирования обеспечивает в исходном графе программы однократную проверку каждого линейно независимого ациклического маршрута и каждого линейно независимого цикла, в совокупности образующих базовые маршруты. Каждый линейно независимый маршрут или цикл отличается от всех остальных хотя бы одной вершиной или дугой. Этот критерий наиболее полно исследован при анализе корреляции сложности и трудоемкости создания программ. Множество проверяемых по этому критерию структур образуется из трех линейно независимых циклов и пяти линейно независимых ациклических структур. При этом суммарная сложность тестов, учитывающих все условия прохождения маршрутов один раз, становится равной 25.

Наиболее глубокий ***третий критерий*** проверки и определения сложности тестирования структуры программы включает требования однократной проверки не только линейно независимых, но и всех линейно зависимых циклов и ациклических маршрутов. Он заключается в анализе хотя бы один раз каждого из реальных ациклических маршрутов исходного графа программы и каждого цикла, достижимого из всех этих маршрутов. Для примера графа программы, представленного на рис. 13.7, по данному критерию необходимо исполнить 6 ациклических и 5 маршрутов, из которых достижимы элементарные циклы. Для реализации выделенных маршрутов в 11 тестах необходимо в совокупности задать 66 условий. При этом особенностью четырех последних маршрутов с циклами, так же как соответствующих им ациклических маршрутов, является полный перебор сочетаний ветвлений в 3 и 12 вершинах.

В реальных программах некоторые маршруты могут оказаться нереализуемыми из-за несовместимости условий, которые последовательно анализируется в разных вершинах (например, вершины 3 и 12 на рис. 13.7). С другой стороны, для каждого реализуемого маршрута может быть необходимой проверка при нескольких прохождениях циклов и нескольких значениях каждой обрабатываемой переменной. Особенно важно проверять циклы с условным выходом на одном-двух промежуточных, а также при максимальном и минимальном числе витков исполнения циклов. В результате показатель сложности, число необходимых тестов и длительности проверки соответственно возрастают.

Для выявления основных закономерностей и оценки ***предельных характеристик структурной сложности тестирования ПМ*** проведен анализ характеристик качества тестирования ***абстрактных ациклических программных модулей*** и представительной выборки реальных модулей сложных ПС. Исследование реальных ПМ показало, что более половины из них не содержат циклов, что позволило сосредоточить на них внимание. Предполагалось, что после тестирования по любому маршруту каждой дуги графа программы вероятность наличия ошибки в этой дуге равна нулю. Ветвления в программах объектных ЭВМ происходят через 5—10 операторов текста программ, поэтому число маршрутов исполнения ациклических ПМ пропорционально их объему, выраженному числом строк текста программ.

Анализ проведен для двух типов ациклических графов при выделении маршрутов по критерию охвата каждой дуги графа программы хотя бы один раз X_1 и по критерию выделения всех маршрутов, различающихся хотя бы одной дугой X_2. В числе выбранных содержатся структуры, охватывающие наиболее типовые варианты компонентов графов программ. Структуры различались шириной графов, вследствие чего отличались число маршрутов и сложность полного тестирования таких структур. Сложность тестирования ПМ оценивалась по числу маршрутов исполнения программы, отражающих ***число тестов***, необходимых для полной проверки корректности структуры модуля. Кроме того, в качестве показателя ***структурной сложности тестирования*** анализировалось ***суммарное число условий***, которое необходимо задать в тестах (сложность тестирования) для проверки модуля.

Граф Γ_1 представлял собой симметричное упорядоченное бинарное дерево с максимальным использованием различных ветвлений и максимальной шириной. В графе Γ_2, наоборот, ширина минимальная и всего два полностью различающихся маршрута. При выделении маршрутов по первому критерию в графе Γ_2 только два маршрута и не зависят от числа вершин ветвления. В графе Γ_1 при выделении маршрутов по первому критерию их число линейно возрастает при увеличении числа вершин. При выделении маршрутов по второму критерию их число пропорционально полному числу вершин для этих типов графов.

Структурная сложность тестирования графов ПМ изменяется в более широком диапазоне, чем число маршрутов, что определило выбор

логарифмического масштаба по оси ординат на рис. 13.8. Наименьшей структурной сложностью характеризуется граф Γ_2 при выделении маршрутов по первому критерию. Для таких графов структурная сложность возрастает практически линейно в зависимости от числа вершин. При выделении маршрутов по второму критерию тот же граф характеризуется наибольшей структурной сложностью. При 30 и более вершинах структурная сложность этого графа почти на порядок выше, чем графа Γ_1 при том же числе вершин. Относительная разница сложности тестирования этих графов при критерии X_2 приблизительно сохраняется при изменении числа вершин от 16 до 100. Такое распределение значений структурной сложности от типов графов обусловлено различием их ширины. Так как число маршрутов при критерии X_2 в зависимости от числа вершин во всех графах изменяется практически одинаково, то определяющим различия структурной сложности является число условий, анализируемых в каждом маршруте. Во всех маршрутах графа Γ_2 участвуют все его вершины, что определило его наибольшую структурную сложность.

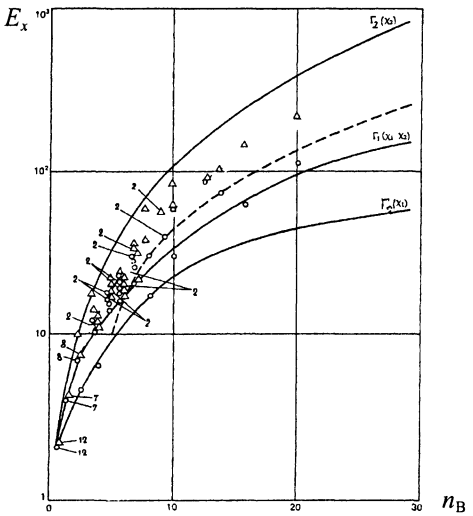

Рис. 13.8

На рис. 13.8 точками (с указанием числа совпадающих значений) отмечены значения структурной сложности тестирования около 70 реаль-

ных ациклических ПМ в двух системах. Характеристики абстрактных графов Γ_1 и Γ_2 действительно охватывают диапазон изменения показателей сложности тестов для реальных программ, которые по каждому критерию группируются приблизительно посередине. Максимальная сложность тестов по второму критерию для произвольных ациклических программ близка к числу вершин в квадрате. По тому же критерию минимальная сложность тестов для широких структурированных графов типа «дерево» на порядок меньше максимальной сложности. Для усредненных оценок сложности полных тестов произвольных ациклических программ хорошее приближение для инженерных оценок при $n_в > 10$ дает выражение $n_в{}^{2}/_{3}$ (штриховая линия на рис. 13.8). Характерно, что увеличение числа вершин в 4 раза (от 32 до 128) для рассмотренных графов приводит к возрастанию структурной сложности более чем в 10 раз. Если же программу, имеющую 128 вершин, разделить на 4 модуля, то их суммарная сложность практически равна только учетверенной сложности модулей, содержащих по 32 вершины. Исследованные реальные ПМ 80% случаев содержат не более 10 вершин и имеют структурную сложность тестирования < 50.

Показано, что при разработке ПМ целесообразно учитывать рациональное ***ограничение размеров модулей на уровне трехсот строк текста***, что соответствует приблизительно тридцати альтернативам в ациклических программах. При этом для полного покрытия таких ПМ тестами необходимо задавать до 1000 условий, что обычно достаточно трудно или невозможно реализовать практически. В среднем полное тестирование программ с 30-ю вершинами ветвления производится тестами с суммарной сложностью около 300—500. Суммарная сложность тестов, необходимых для полного тестирования программ, имеющих различные структуры, может отличаться в несколько раз.

Поэтому при разработке ПМ ***рекомендован рациональный размер программ модулей*** в пределах 100—200 строк текста, для полного тестирования которых достаточно использовать 10—20 тестов с суммарным числом условий ветвления до 100. При превышении рекомендуемых размеров ПМ их трудно протестировать полностью и ***целесообразно делить на более мелкие компоненты***, доступные для практически полного покрытия тестами.

Для получения практических оценок достигаемой корректности программы при покрытии тестами ее структуры необходимо оценить ***диапа-***

зон реальной вероятности ошибки, допускаемой квалифицированным программистом первично в каждой дуге графа программы. Экспериментально установлено, что для слабо структурированных программ число ошибок, выявляемых в процессе тестирования программных модулей, составляет около одного процента числа строк текста этих модулей. Для программ обработки информации и управления число условных переходов составляет около 10% числа строк в программе, т.е. ветвление в программе происходит в среднем после исполнения 10 строк текста линейных участков. Следовательно, порядка 10% линейных участков (или дуг в графе) программных модулей могут содержать первоначально ошибки перед тестированием, что соответствует вероятности $q_{ij} \approx 0{,}1$.

Использование правил структурного программирования, спецификаций требований на модули и группы программ, а также современной технологии программирования позволяет снизить первичную вероятность ошибок приблизительно на порядок, т.е. до уровня $q_{ij} \approx 0{,}01$. Поэтому оценивание стратегий тестирования и достигаемой при этом корректности целесообразно проводить в диапазоне $q_{ij} = 0{,}1$—$0{,}01$, соответствующем практически наихудшим и наилучшим значениям вероятностей ошибок в дуге до тестирования.

Для простейших оценок можно предположить, что все дуги в графе программы имеют одинаковую длину и эквивалентны по вероятности появления в них ошибок, т.е. $q_{ij} = \text{const}$ в начале тестирования модуля. В действительности дуги графов реальных программ содержат различные ***типы операторов, которые в разной степени подвержены искажениям и ошибкам***. Операторы в программе различаются по своей сложности и соответственно по вероятности искажения их программистами в процессе создания программ. В зависимости от типа оператора, в котором допущена первичная ошибка, различаются последствия проявления искажений (вторичные ошибки).

Планирование тестирования структуры программных модулей и оценивание их корректности в значительной степени может быть автоматизировано. Если фиксировать маршруты, по которым уже выполнено тестирование, то можно автоматически исключать их из информирования и выдавать на регистрацию тестировщикам только группу маршрутов, подлежащих первоочередной проверке. Эти же данные могут использоваться для автоматического расчета полноты проведенной проверки и для ***оцени-***

вания достигнутой структурной корректности программы по каждому из критериев выбора маршрутов.

Рассмотренный анализ структурной корректности программ в пределах модуля может быть распространен на группы взаимодействующих модулей. В этом случае каждый модуль следует рассматривать как закрытую структуру, характеризующуюся ранее оцененной структурной корректностью. В графе группы взаимодействующих модулей может быть выделена совокупность маршрутов исполнения этой группы и проконтролирована полнота их тестирования. Последовательно укрупняя выделяемые группы таких модулей снизу вверх, можно упорядоченно их тестировать и оценивать достигнутую структурную корректность групп программ и ПС в целом.

13.6. Тестирование обработки потоков данных программными компонентами

Функционирование любой программы можно рассматривать как ***обработку потока данных***, передаваемых от входа в программу к ее выходу (см. п. 13.1). Входные данные последовательно используются для определения ряда промежуточных результатов вплоть до получения необходимого набора выходных данных. Задача тестирования и анализа потока данных состоит в установлении корректности их обработки и в выявлении ошибок в тестируемой программе. Эта задача может решаться ***статически*** — без исполнения программы (анализом по ее тексту) и ***динамически*** — путем реального исполнения программы на ЭВМ в машинных кодах при различных исходных данных.

Наборы действий по преобразованию исходных данных в выходные могут быть формализованы ***диаграммами потоков данных*** (DFD — Data Flow Diagrams). Для этого применяется система графических элементов, содержащих квадратики с описаниями сущностей и номерами, а также соединяющие их стрелки процессов:

— внешние сущности — объекты, являющиеся источниками или потребителями информации, идентифицируемые их содержанием и номерами;

— процессы, перемещающие объекты от одного действия к другому, преобразующие исходные данные в результирующие;

— накопители объектов или данных, где временно они размещаются на хранение;

— потоки данных — информация, передаваемая от источника к потребителю.

Для построения DFD-диаграмм формализованы синтаксис и семантика графических элементов: отражающих движение объектов — процедуры программ; описаний внешних сущностей — источников и потребителей данных; их хранения. Рекомендуется вначале определять набор действий, описывающих, что должны выполнять процедуры программы. Затем строить модель окружения — внешние сущности, порождающие процессы и специфическое поведение при обработке данных. Наборы простейших DFD-диаграмм — операторов программы, объединяются в иерархические структуры, отражающие потоки данных в программных модулях или функциональных компонентах из ряда модулей.

Данные, участвующие в вычислениях, на языках программирования высокого уровня ***определены явно*** по имени, типу, способам доступа и использования. Это позволяет рассматривать программу в виде мультиграфа, заданного структурой передач управления (потоком управления) и графом преобразования данных, участвующих в вычислениях (поток данных). Пересечение потока управления и потока данных осуществляется в операторах ветвления: проверки условий и циклах. Совместный анализ потоков управления и данных позволяет проверять корректность реализации областей определения переменных на маршрутах исполнения программы.

Последствия ошибок в программе могут проявляться как малые изменения некоторых переменных в процессе вычислений и как полное искажение или отсутствие на выходе требующихся величин. Тестирование программного модуля целесообразно проводить на упорядоченных наборах данных с учетом степени их влияния на выходные результаты. С этой позиции для последующего анализа целесообразно выделить ***два вида обработки данных***:

— полностью изменяющей область определения и значения результатов обработки;

— изменяющей результаты в пределах некоторой ограниченной, правильной области определения.

Первому виду обработки соответствуют исходные данные в критических точках и на границах областей изменения переменных. При таких критических значениях может изменяться маршрут исполнения программы, вследствие чего возможно наибольшее изменение результатов. Поэтому обычно тестирование обработки данных, прежде всего, направлено на проверку исполнения программ при значениях переменных, влияющих на выбор маршрута и логику функционирования программ (стратегия выделения областей переменных). Граничные условия — это ситуации, возникающие в непосредственной близости к границам областей коренного изменения обрабатываемых переменных. Число таких ***критических значений*** каждой переменной может быть на несколько порядков меньше, чем число значений по всей внутренней части области изменений этой величины.

Большинство критических значений (предикатов) может существенно влиять на результаты и подлежит наиболее тщательному тестированию. В этой части тестирование обработки данных по содержанию близко к тестированию структуры программы (см. п. 13.4). При этом виде тестирования маршруты формируются в процессе анализа и обработки данных на последовательных операторах условий в тексте программы. Набор сочетаний исходных данных в тестах непосредственно влияет на степень охвата тестированием из полного набора участков программы. Путем сопоставления проверенных маршрутов с маршрутами, выделенными по графу программы при различных критериях, можно оценивать достигнутую полноту тестирования модуля и приблизительно степень его корректности.

Второму виду обработки соответствуют данные в ограниченной (или неограниченной) области определения, которая может делиться на некоторое множество сопрягающихся областей (подобластей). Изменение данных внутри такой области не влияет на маршрут исполнения программы. Поэтому для проверки функционирования программы из всего множества значений достаточно использовать при тестировании ***только несколько значений*** внутри и вблизи границ области. Количество величин, используемых для тестирования при обработке этого вида, может быть на несколько порядков меньше полного числа значений каждой переменной в области. В процессе тестирования проверяется точность осуществляемых вычислений, правильность масштабирования и размерности обрабатываемых величин, корректность формирования логических величин. При этом

тестирование должно охватывать всю область изменения каждой обрабатываемой переменной и каждой результирующей величины.

При анализе обработки данных в пределах областей их определения ***методы тестирования целесообразно применять упорядоченно в следующей последовательности***:

— тестирование при значениях данных, определяющих маршруты исполнения программы (стратегия областей);

— тестирование корректности записи и считывания переменных при вычислениях и полноты состава выходных данных на всех маршрутах исполнения программы;

— тестирование точности результатов вычислений и корректности обработки каждой переменной;

— тестирование на полное соответствие спецификации требований состава, значений и точности выходных данных.

В приведенной последовательности частные методы тестирования позволяют, прежде всего, выявлять первичные ошибки, которые способны искажать результаты в наибольшей степени. При ограниченных ресурсах и такой последовательности тестирования в программе могут оставаться ошибки, наименее влияющие на корректность выходных данных. На основе таких проверок может оцениваться степень охвата тестированием всех условий, определенных в спецификации, и дополнительное тестирование следует проводить только при отдельных недостаточно проверенных входных данных.

Тестирование при значениях данных, определяющих маршруты исполнения программы (стратегия областей). Маршруты последовательности обработки данных могут зависеть от любых типов анализируемых величин. Одной из задач тестирования является проверка сопоставимости сравниваемых типов величин и идентичности условий их кодирования (разрядности, масштабов). Критические значения — ***предикаты***, влияющие на выбор маршрутов, во многих случаях не являются фиксированными, а формируются при обработке данных и/или сравнении нескольких переменных. При этом предикаты могут образовываться во всей области изменения каждой из переменных, например, когда они оказываются равными или отличаются на некоторую постоянную величину.

Предикаты, определяющие выбор маршрутов исполнения программы, могут формироваться в результате вычислений на линейных участках

программы. Эти участки в среднем невелики и содержат около 5—10 строк текста программы. Каждая ограниченная область исходных данных соответствует определенному маршруту в программе. ***Граница области определяется интерпретациями предикатов по маршруту*** и состоит из набора участков границы, каждый из которых определяется единственным, простым предикатом, выбирающим дугу маршрута в графе программы. Каждый участок границы области может быть открытым или закрытым в зависимости от оператора условий в предикате. Закрытый участок границы принадлежит ограничиваемой области и формируется предикатами с операторами $\leq$, $\geq$ или $=$. Открытый участок границы не входит в состав области и формируется операторами $<$, $>$ и $\neq$. Общее число предикатов в маршруте — это верхний предел числа граничных участков области входных переменных данного маршрута, так как некоторые предикаты маршрута могут в действительности не создавать граничных участков. Такие случаи возникают, когда предикат требуется для нескольких путей, и в некоторых из них повторно анализируется на маршруте.

Таким образом, программа по отношению к потоку данных может рассматриваться, прежде всего, как выполняющая ***разделение пространства исходных данных на области***, каждая из которых соответствует одному исполняемому маршруту. Ошибки в программе могут быть обусловлены модификацией границы области определенного маршрута, приводящей к расширению или сужению пространства исходных данных соответствующего маршрута. Кроме того, деформация границ областей может приводить к ошибкам уничтожения некоторых областей и потери соответствующих им маршрутов. Причинами таких ошибок могут быть искажения операторов анализа условий или искажения в процессе вычисления значений предикатов при правильном содержании оператора условия. Искажения операторов анализа условий может приводить как к деформации границы области, так и к появлению новых границ или их уничтожению, вследствие чего могут разделяться или сливаться области.

Сложность тестов линейно растет с увеличением размерности пространства исходных данных (числа требований или переменных) и с ростом числа предикатов на маршрутах. Для многих типовых модулей сложность тестов оказывается допустимой для практически полной проверки модуля. Ограничения метода проверки областей могут проявляться при

сложных организациях циклов, когда резко возрастает число маршрутов и анализируемых условий.

Тестирование корректности определения и использования данных на маршрутах исполнения программы. Если маршруты исполнения программы соответствуют допустимым областям изменения входных данных, то целесообразно проверять корректность основных операций обработки данных на выделенных маршрутах. Каждая величина на маршруте исполнения программы считывается из памяти, и после использования для вычислений записывается в память ЭВМ для хранения и последующей обработки. Чередование операций чтения и записи переменных может быть нарушено в результате ошибок в программе. Для выявления таких ошибок проводится тестирование корректности записи и считывания реальных данных или статический анализ этих операций по исходному тексту программы.

Тестирование корректности обработки каждой переменной и точности результатов вычислений. Когда показано, что сочетания данных и их области определения соответствуют корректному формированию маршрутов в программе, а также нет явных ошибок в последовательностях определения и использования каждой переменной, целесообразно провести тестирование корректности обработки каждой переменной и точности вычислительной части программы. Этот вид тестирования производится преимущественно с вещественными и целыми величинами во внутренней части их областей определения при операциях с фиксированной точкой. Кроме того, может выполняться дополнительный контроль точности вычислений при граничных значениях, ранее использовавшихся для тестирования маршрутов по областям определения.

Множество тестовых значений для проверки вычислений при простых числовых переменных целесообразно строить ***упорядоченно с учетом следующих правил***:

— входные тестовые данные в области гладкого изменения, зависящих от них результатов, должны принимать, по крайней мере, значения, близкие к наибольшим и наименьшим, а также одно-два промежуточных значения;

— тестирование должно проводиться при всех особых значениях входных переменных — в точках резкого возрастания или разрыва производных, при нулевых, единичных и предельно малых численных значениях;

— входные тестовые значения должны обеспечивать проверку программы при выходных результатах, имеющих особые точки резкого изменения или разрыва производных;

— если значения некоторой переменной зависят от значений другой переменной, то их необходимо тестировать при особых значениях сочетаний переменных (равенство обеих переменных, малое и предельно большое их различие, нулевые и единичные значения).

Таким образом, для каждой простой числовой переменной, кроме трех точек вблизи и на границе области определения, обычно необходимо тестирование программы в 3—4 промежуточных и в 2—5 особых точках значений входных данных. При 10 входных переменных и сложных вычислениях в программном модуле для тестирования вычислений может потребоваться до 50 тестовых значений. Группируя и упорядочивая тестовые значения разных переменных, их общее количество можно сократить до 5—10 тестовых наборов.

ЛЕКЦИЯ 14

ИНТЕГРАЦИЯ, КВАЛИФИКАЦИОННОЕ ТЕСТИРОВАНИЕ И ИСПЫТАНИЯ КОМПЛЕКСОВ ПРОГРАММ

14.1. Процессы оценивания характеристик и испытания программных средств

Для оценивания характеристик и испытаний программных средств на различных этапах жизненного цикла в качестве методологической основы целесообразно использовать рекомендации стандарта **ISO 14598:1-6** — Оценивание программного продукта. Процесс оценивания ПС в стандарте представлен как совокупность действий, выполняемых в кооперации заказчиком и оценщиком — испытателем. Потенциальными заказчиками оценивания ПС могут быть разработчики, поставщики, покупатели, пользователи ПС, производители систем обработки информации, а также третейские испытательные лаборатории программных продуктов. Для получения наибольшего эффекта от результатов испытаний рекомендуется, чтобы оценивание было насколько возможно:

— объективным — результаты оценивания должны базироваться на реальных фактах, не окрашенных чувствами или мнениями испытателей;

— повторяемым — повторное оценивание тождественного продукта для тождественной спецификации тем же испытателем должно давать те же результаты, что и при первичном оценивании;

— воспроизводимым — оценивание того же продукта для той же спецификации различными специалистами должно давать те же результаты, как и при предыдущем испытании;

— беспристрастным — исполнители процесса оценивания должны относиться непредубежденно к любому конкретному результату.

Общую схему процессов оценивания характеристик комплексов программ составляют (рис. 14.1):

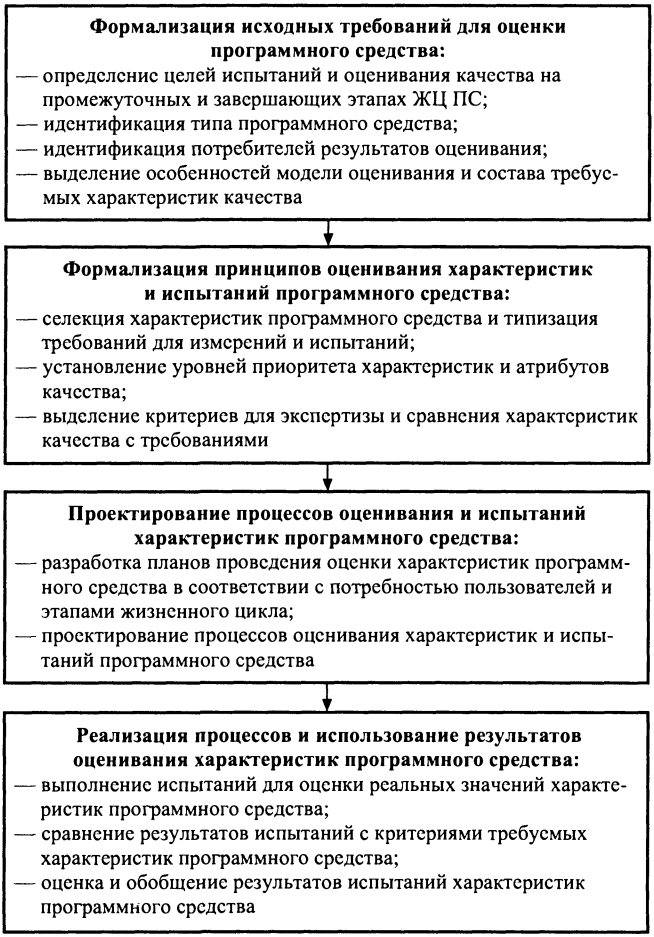

Рис. 14.1

— формализация исходных требований для оценки значений характеристик программного средства, определение целей испытаний, идентификация потребителей результатов испытаний;

— формализация принципов и особенностей оценивания при проведении экспертиз, измерений и испытаний характеристик программного средства, выделение критериев для сравнения полученных характеристик с требованиями;

— планирование и проектирование процессов оценивания характеристик в жизненном цикле программного средства в соответствии с потребностями пользователей этих характеристик;

— реализация процессов испытаний, измерений и оценивания достигнутого качества программного продукта, сравнение результатов испытаний с требованиями; оформление и использование результатов.

В ***первой части*** стандарта представлена концепция планирования и управления процессами оценивания характеристик программ, а также их связь с процессами управления жизненным циклом ПС (по **ISO 12207**). При подготовке к испытаниям рекомендуется структурировать технологию и процедуры применения конкретного ПС с целью последовательного детального оценивания групп функциональных характеристик или отдельных атрибутов качества на этапах жизненного цикла.

Решение о проведении оценивания характеристик ПС может быть принято в процессе разработки и всего ЖЦ. Если такое решение принято на начальном этапе разработки, то появляется возможность встроить в процессе разработки тесты и средства измерения для испытаний. Это обеспечивает максимальный успех в удовлетворении всех требований относительно результатов оценивания характеристик ПС и минимизирует риск ошибок в экстремальных, незапланированных ситуациях. Когда заказчиком является разработчик ПС, ранний контакт с оценщиком для испытания или экспертизы может помочь разработчику учесть некоторые специальные требования со стороны испытателя. Для очень больших, сложных проектов ПС разработчику должно быть выгодным иметь детальную кооперацию с оценщиком характеристик во время всего процесса разработки продукта для минимизации продолжительности и стоимости процесса испытаний. ***Обязанностью заказчика*** испытаний должно быть:

— установить его необходимые легальные права для выполнения испытаний ПС;

— предоставить испытателю информацию, необходимую для идентификации и описания программного средства и компонентов;

— установить начальные требования и вступить в переговоры с испытателем для выработки реальных оценочных требований;

— требования к испытаниям следует подчинять соответствующим регламентирующим документам и стандартам;

— установить требования к конфиденциальности информации, передаваемой на испытания;

— при необходимости обеспечить поддержку испытателю, включая обучение и доступ к подходящим документам;

— гарантировать своевременную поставку испытателю описания и компонентов ПС, включая документацию и другие материалы;

— информировать испытателя о любых возможных факторах, которые могут обесценить результаты испытаний.

Испытателю следует проанализировать технические ограничения, относящиеся к измерениям или верификациям, заданным в спецификации оценивания, и документировать подходящий метод. ***Метод оценивания — это процедура***, описывающая действие, выполняемое испытателем при получении результата специфицированного измерения, экспертизы или верификации для системы, компоненты или ПС. Когда метод оценивания описывается на основе программного инструментария, данный инструментарий должен быть идентифицирован в плане и методике испытаний. При этом ***обязанности оценщика — испытателя*** включают:

— проверить легальные права заказчика на систему и ПС для оценивания, для чего испытатель может потребовать соответствующие документы от заказчика;

— хранить конфиденциальность обо всей информации, передаваемой заказчиком, включая тексты ПС, записи результатов и отчет об испытаниях;

— предоставить квалифицированный и обученный персонал для выполнения испытаний;

— обеспечить инструментарий и технологию оценивания;

— выполнить испытания в соответствии с оценочными требованиями и спецификацией заказчика;

— хранить записи любой работы, выполняемой при оценивании, которые воздействуют на результаты;

— обеспечить наглядность проведения испытаний для наблюдения заказчиком и своевременную передачу отчета об оценивании заказчику.

Оценочные требования должны содержать общее описание области применения ПС и состоять из перечня требований к характеристикам. Должна быть указана относительная значимость (приоритет) конкретных характеристик в требованиях. Для каждого положения в оценочных требованиях должна быть представлена спецификация информации, содержащейся в компонентах ПС.

Спецификация оценивания — испытаний должна определять область экспертизы и измерения различных компонентов продукта, передаваемого на оценивание. Уровень детализации в спецификации испытаний должен быть таким, чтобы на ее основе гарантировались повторяемость и воспроизводимость испытаний. Заказчик должен представить описание оцениваемого продукта. Целью такого описания является определить область оценивания и идентификацию тех компонентов, которые рассматриваются как часть продукта, в отличие от компонентов ПС, на которые только ссылаются для облегчения понимания функций продукта. Это должно позволить разложить оценочные требования на субхарактеристики. Затем испытатель должен специфицировать методы измерения и экспертизы, предназначенные для анализа характеристик, субхарактеристик и атрибутов качества выбранных компонентов. Оценщик должен выполнять верификацию спецификации оценивания в соответствии с требованиями, установить, что компоненты, перечисленные в описании продукта, содержат всю необходимую информацию для выполнения оценивания в соответствии с требованиями.

План испытаний или экспертизы характеристик программных продуктов должен состоять из разделов:

— введение — постановка задачи и цели оценивания — испытаний;

— методология обеспечения объективности испытаний характеристик ПС;

— выделенные для проекта характеристики и атрибуты качества и безопасности ПС по стандарту **ISO 9126:1-4** и другим стандартам;

— требуемое качество ПС и достоверности процессов испытаний;

— расписание выполнения работ по испытаниям характеристик ПС;

— распределение обязанностей и ответственности специалистов при оценивании характеристик и атрибутов качества;

— анализ и использование результатов испытаний;

— содержание и оформление отчетов по выполнению испытаний;

— дополнительные требования по технике, инструментальным средствам и методам обслуживания испытаний, по соответствию стандартам и организационной поддержке измерений и экспертиз.

Испытатель должен обеспечить входные данные для процесса оценивания: предопределенные спецификации оценивания; методы и инструментарий. ***Процесс оценивания — испытаний*** рекомендуется в стандарте из следующих действий:

— анализ оценочных требований, при котором выделяются для идентификации реальные требования заказчика на испытания;

— спецификация процессов и возможных результатов, при которой вырабатывается спецификация оценивания на основе требований и описания программного продукта, предоставляемых заказчиком;

— проект процессов испытаний, при котором вырабатывается план на основе спецификации оценивания, компонентов ПС и методов, предлагаемых испытателем;

— процесс выполнения плана оценивания, который состоит из моделирования, экспертизы, измерения и испытания компонентов ПС в соответствии с планом, с использованием программного инструментария, а также действия, осуществляемые испытателем, которые фиксируются и результаты заносятся в отчет;

— заключение по оцениванию, которое состоит в предоставлении отчета о результатах испытаний компонентов и ПС.

В процессе испытаний может потребоваться использование программных инструментов для сбора обработанных данных или для осуществления интерпретации промежуточных данных. Штат оценщиков должен быть обучен пользоваться соответствующими инструментами. При оценивании должны быть ***получены, зафиксированы и предъявлены заказчику следующие данные***:

— требования, которые описывают цели оценивания, в частности, требуемая критичность и безопасность испытанного продукта;

— спецификация оценивания, которая определяет весь анализ и выполняемые измерения, и все компоненты ПС, которые должны анализироваться и измеряться;

— план оценивания, который описывает операционные процедуры, необходимые для выполнения спецификации оценивания, в частности, все методы и инструменты, используемые при оценивании;

— записи об оценивании, которые состоят из плана оценивания и детальных действий оценщика при выполнении плана;

— отчет по оцениванию характеристики, который содержит требования, спецификацию оценивания, результаты измерений и анализа, а также любую другую информацию, необходимую для возможности повторения или воспроизведения оценивания, а также обобщенный отчет заказчику.

Цель ***заключения*** состоит в обзоре отчета об испытаниях и передаче данных оценивания заказчику. Следует организовывать совместный обзор и анализ отчета заказчиком и испытателем. Заказчику следует дать возможность сделать комментарии по отчету. Если такие комментарии сделаны, то их следует внести в специальный раздел отчета, после чего он должен быть передан испытателю и заказчику.

С позиций разных ***потребителей результатов*** измерения и оценивания качества ПС построены ***третья, четвертая и пятая части*** стандарта **ISO 14598:1-6** — соответственно для:

— разработчиков — оценивание внутренних и внешних характеристик качества (ч. 3);

— оперативных пользователей — измерение внешних метрик и метрик в использовании (ч. 4);

— заказчиков и испытателей — определение метрик в использовании (ч. 5).

В каждой части ***выделены и детализированы подобные разделы***: особенности и потребности конкретных пользователей в результатах испытаний и номенклатуре требуемых характеристик ПС; концепция проведения испытаний и оценивания качества; определение требований к процессам испытаний характеристик программ; идентификация характеристик и атрибутов качества ПС для конкретных пользователей результатов испытаний.

Результаты оценки характеристик предлагается отражать с позиции: процессов жизненного цикла; продуктов и их компонентов; функционирования и применения ПС. Требования к процессам оценивания ***рекомендуется структурировать*** на главные (функциональные), организационные, проектные, а также выделять внутренние и внешние метрики качества и

их измерение, ориентируясь на субхарактеристики и их атрибуты в соответствующих частях стандарта **ISO 9126:1-4**. Реализация процессов испытаний программного продукта по требованиям стандарта должна проводиться квалифицированными и аттестованными специалистами, ***независимыми от разработчиков*** проекта, процессов создания ПС и его компонентов, однако коррелированно с этапами жизненного цикла конкретного проекта в соответствии с применяемой адаптированной версией стандарта **ISO 12207**.

Обращается внимание на целесообразность учета и использования истории развития и изменений эксплуатационных требований заказчиков-потребителей к определенному проекту и функциональному назначению ПС. В отчете об испытаниях рекомендуется представлять характеристики качества в соответствии с номенклатурой и мерами субхарактеристик и атрибутов, адаптированными к назначению, функциям и особенностям конкретного проекта ПС и потребителей результатов измерений.

Конфиденциальность всех компонентов и документов испытаний ПС должна защищаться оценщиком. Требования конфиденциальности охватывают многие аспекты оценочной работы: включая получение, управление, хранение и передачу всей информации, касающейся характеристик продукта. Конфиденциальность промежуточных данных испытаний должна защищаться, так же как и конфиденциальность оригиналов компонентов и документов. Испытатель должен приложить усилия, чтобы предотвратить любые случайные, ошибочные или злоумышленные модификации этих данных.

14.2. Организация и методы оценивания характеристик сложных комплексов программ

Характеристики качества функционирования программных средств зависят не только от их внутренних свойств, но и ***от свойств внешней среды***, в которой они применяются (см. **ISO 12119**). Для сокращения неопределенностей и прямых ошибок при оценивании качества ПС необходимо до начала испытаний определить основные параметры внешней среды, при которых должен функционировать комплекс программ с требуемыми характеристиками при оценивании его качества и эксплуатации.

Для этого заказчик и разработчик совместно должны структурировать, описать и согласовать ***модель внешней среды*** и ее параметры в среднем, типовом режиме применения ПС, а также в наиболее вероятных и критических режимах, в которых должны обеспечиваться требуемые характеристики качества функционирования ПС. Такая ***модель должна отражать и фиксировать характеристики*:**

— внешних потоков информации, в том числе их распределение по видам источников, характеристикам качества данных и возможности их дефектов;

— интенсивность и структуру типовых сообщений от оперативных пользователей и администраторов и их необходимую квалификацию, отражающуюся вероятностью ошибок и качеством выдаваемой информации;

— возможных негативных и несанкционированных воздействий от внешней среды при применении ПС;

— необходимые характеристики вычислительных средств, на которых предназначено функционировать комплексу программ с требуемым качеством.

При сопоставлении результатов оценивания характеристик качества с требованиями технического задания и спецификаций разработчик или поставщик ***обязан удовлетворять требования заказчиков только в пределах согласованных параметров*** модели внешней среды. Оценивание качества ПС за этими пределами должно дополнительно согласовываться испытателями с разработчиком. При этом невыполнение требований может квалифицироваться как их расширение за пределы, ограниченные контрактом, и не учитываться при оценивании заказчиком характеристик качества ПС, или как дополнительные работы, подлежащие соответствующему финансированию со стороны заказчика, для доработки программ с целью удовлетворения этих требований.

Внутренние квалификационные испытания качества программных средств (испытания главного конструктора), которые зачастую совмещаются с завершением комплексной отладки, должны оформляться документально и являются основанием при предъявлении ПС заказчику на квалификационные испытания для завершающего оценивания характеристик качества программного продукта (см. **ISO 12207**, **ISO 15504**, **ISO 16326**). Разработчик должен реализовать и оценить проект, комплекс про-

грамм, тесты, результаты тестирования и документацию для пользователя, учитывая:

— полноту охвата испытаниями всех требований спецификаций к компонентам и к ПС в целом;

— согласованность с требуемыми заказчиком и ожидаемыми результатами применения ПС;

— возможность интеграции и тестирования ПС в составе системы;

— возможность функционирования и сопровождения версий ПС в соответствии с требованиями контракта.

Любые ***испытания ограничены допустимым количеством и объемом проверок***, а также длительностью работы комиссии испытателей, поэтому не могут гарантировать абсолютную проверку качества продукта. Для повышения достоверности определения и улучшения оценивания характеристик ПС после внутренних испытаний комплекс программ целесообразно передавать некоторым пользователям на ***опытную эксплуатацию в типовых условиях***. Это позволяет более глубоко оценить эксплуатационные характеристики созданного комплекса и устранить некоторые дефекты и ошибки. Опытную эксплуатацию целесообразно проводить разработчиками с участием испытателей-заказчиков и некоторых пользователей, назначаемых заказчиком. Результаты и характеристики качества опытной эксплуатации после испытаний главного конструктора могут учитываться при проведении заказчиком квалификационных испытаний для их сокращения.

В лекции 13 рассмотрены этапы тестирования компонентов и ПС в целом с позиции последовательного увеличения функциональной сложности тестов и взаимодействия с объектами внешней среды. При этом не учитывались организационные этапы испытаний в соответствии со стандартами и их подотчетность разработчикам-поставщикам и заказчикам. Этапы и процессы квалификационного тестирования ПС ***с целью формального удостоверения для заказчика достигнутых характеристик*** качества комплекса программ и его компонентов в составе системы регламентированы в стандартах **ISO 12207, ISO 15504**. В них выделены три основных, функциональных ***этапа реализации квалификационного тестирования и испытаний*** (рис. 14.2):

— квалификационное тестирование функциональных компонентов и ПС в целом вне аппаратуры системы;

— интеграция и тестирование программного средства в целом в составе аппаратуры системы;

— квалификационное тестирование и полные испытания системы в комплексе с программным средством.

Рис. 14.2

Квалификационное тестирование функциональных компонентов и ПС в целом выполняется для того, чтобы продемонстрировать заказчику, что реализованы все требования контракта и достигнуто необходимое качество функционирования комплекса программ. Квалификационное тестирование должно покрывать все требования к компонентам в спецификации требований к ПС и в спецификациях требований к интерфейсам. Тестирование компонентов для каждой конфигурации должно показать, что полностью удовлетворены требования к компонентам, которые должны быть реализованы в данной конфигурации. Ответственными за квалификационное тестирование компонентов не должны быть лица, осуществившие выполнение рабочего проекта или программирование соответствующего компонента. Это не исключает возможность оказания помощи в проведении квалификационного тестирования со стороны лиц, выполнявших рабочий проект или программы, например, путем предоставления тестовых вариантов, основанных на знании ими внутренней реализации компонента или ПС.

Разработчик должен определить и зарегистрировать процесс подготовки к тестированию, тестовые варианты, сценарии и процедуры, которые должны использоваться для квалификационного тестирования компонента и проследить соответствие между тестовыми вариантами и требованиями контракта. Он должен подготовить тестовые данные, необходимые для выполнения тестовых вариантов, и предварительно уведомить представителя заказчика о времени и месте проведения квалификационного тестирования. Если испытание компонента должно быть засвидетельствовано представителем заказчика, то до его проведения разработчик должен проверить тестовые варианты и тестовые процедуры, чтобы гарантировать, что они полны и точны и что ПС готово для проведения тестирования в присутствии представителя заказчика. Результаты этих проверок должны быть зарегистрированы в файлах разработки ПС, а тестовые варианты и процедуры соответствующим образом модифицированы для устранения выявленных дефектов. Следует выполнить все необходимые изменения в ПС, предварительно уведомив об этом представителя заказчика, осуществить повторное тестирование в необходимом объеме, модифицировать файлы разработки ПС и другие программные продукты, основываясь на результатах интеграции и тестирования модулей и компонентов.

Результаты этих действий должны быть включены в отчет для заказчика о проведенных разработчиком предварительных испытаниях.

Интеграция и тестирование ПС в составе аппаратуры системы содержит их объединение, тестирование полученного в результате комплекса с целью определения, работают ли они вместе, как требуется по контракту, и должно продолжать этот процесс до тех пор, пока интеграция и тестирование не будут выполнены для всех компонентов программ и аппаратуры. Тестовые варианты должны покрывать все аспекты системного уровня и требуемые характеристики качества проекта. Разработчик должен выполнить необходимые корректировки программ, принять участие в повторном тестировании в необходимом объеме и модифицировать файлы разработки ПС и другие программные компоненты, основываясь на результатах интеграционного тестирования.

Квалификационное тестирование системы и программного продукта в целом выполняется, чтобы продемонстрировать представителю заказчика, что удовлетворены все требования технического задания и характеристики качества соответствуют условиям контракта. Оно должно покрывать все требования в спецификациях системы и подсистем, а также требования к интерфейсу с внешней средой. Испытания должны включать тестирование на объектной вычислительной системе или на альтернативной модели системы, одобренной представителем заказчика. Разработчик должен участвовать в разработке и регистрации процесса подготовки к тестированию, тестовых вариантов, сценариев и тестовых процедур, которые нужно использовать для полного испытания системы, и в прослеживании соответствия между тестовыми вариантами и требованиями к характеристикам качества системы. Каждое проверяемое требование должно соответствовать конкретным обоснованным характеристикам системы, иметь уникальный для проекта идентификатор, чтобы можно было провести тестирование и проследить его выполнение с помощью объективного теста. Для каждого требования должны выбираться квалификационные методы для подсистем и компонентов ПС, которые необходимо прослеживать в требованиях к системе. Степень детализации требований следует выбирать, учитывая в первую очередь те характеристики качества, которые внесены в условия приемки системы, и отдавать приоритет тем из них, которые заказчик требует обеспечить обязательно.

Все полученные результаты должны быть включены в отчет о тестировании ПС и системы. Если квалификационное тестирование системы должно быть засвидетельствовано представителем заказчика, то до его проведения разработчик должен проверить тестовые варианты и тестовые процедуры, чтобы гарантировать, что они полны и точны и что система готова для проведения тестирования в присутствии представителя заказчика. Испытания должны быть выполнены в соответствии с утвержденными заказчиком тестовыми вариантами, сценариями и процедурами.

Оценивание качества программного продукта при квалификационных, приемо-сдаточных испытаниях проводится комиссией заказчика, в которой участвуют руководитель (главный конструктор) разработки и некоторые ведущие разработчики или аттестованная сертификационная лаборатория. ***Комиссия при испытаниях должна руководствоваться следующими документами*** (см. рис. 14.2):

— утвержденными заказчиком и согласованными с разработчиком контрактом, техническим заданием и спецификациями требований на ПС;

— действующими государственными и ведомственными стандартами на жизненный цикл и испытания программ, на технологическую и эксплуатационную документацию, а также стандартами де-факто, согласованными с заказчиком для использования, — профилем стандартов и нормативных документов;

— Программой испытаний по всем требованиям контракта, технического задания и спецификаций;

— методиками испытаний, охватывающими каждый раздел требований технического задания, спецификаций и Программы испытаний;

— комплектом адекватной эксплуатационной и технологической документации на комплекс программ.

Программа испытаний является планом проведения серии экспериментов и должна разрабатываться с позиции допустимой минимизации объема тестирования в процессе проведения испытаний для проверки выполнения требований технического задания и соответствия предъявленной документации. Программа испытаний, методики их проведения и оценки результатов, разработанные совместно заказчиком и разработчиком, должны быть согласованы и утверждены. Они должны содержать уточнения и детализацию требований технического задания и спецификаций для данного ПС, а также гарантировать корректную проверку всех задан-

ных характеристик качества. ***Программа испытаний должна содержать следующие четко сформулированные разделы:***

— объект испытаний, его назначение и перечень основных документов, определивших его разработку;

— цель испытаний с указанием всех требований технического задания, характеристик и атрибутов качества, подлежащих проверке, и ограничений на проведение испытаний;

— собственно Программу испытаний, содержащую проверку комплектности спроектированного ПС в соответствие с техническим заданием и план проведения тестирования для проверки по всем разделам технического задания и дополнительных требований, формализованных отдельными решениями разработчиков и заказчика;

— методики испытаний, однозначно определяющие все понятия проверяемых характеристик качества, условия и сценарии тестирования, инструментальные средства, используемые для испытаний;

— методики обработки и оценки результатов тестирования по каждому разделу Программы испытаний.

Методика испытаний должна содержать: описание организации процесса тестирования, тестовые варианты, сценарии и процедуры, которые используются при испытаниях отдельного компонента или ПС в целом. Каждый тест должен иметь уникальный для данного проекта идентификатор; должны быть представлены инструкции для проведения тестирования; описание аппаратуры и ПС для реализации тестирования, а также инструкции для повторного тестирования. Кроме того, должны быть приведены ссылки на проверяемые требования, указаны условия выполнения (конфигурация аппаратуры и компонентов ПС), входные данные, эталонные и ожидаемые результаты, критерии оценки качества результатов, процедура проведения тестирования для каждого тестового варианта, допущения и ограничения.

План испытаний ПС должен описывать порядок квалификационного тестирования компонентов и подсистем, тестовую внешнюю среду, которая будет использоваться при тестировании, идентифицировать выполняемые тесты и указывать план-график тестовых действий. Для каждой предполагаемой тестовой реализации должны быть указаны: используемые версии аппаратных средств; интерфейсное оборудование; дополнительные внешние устройства; генераторы тестовых сообщений; устрой-

ства синхронизации тестов. Кроме того, в документе должны быть представлены план-график тестирования и матрица трассирования тестов к требованиям спецификаций на ПС или на его компоненты, а также субподрядчики, принимающие участие в тестировании, их роль и ответственность.

Большой объем разнородных данных, получаемых при испытаниях крупномасштабных ПС, и разнообразие возможных способов их обработки, интерпретации и оценки приводят к тому, что важнейшими факторами становятся ***методики обработки и оценки результатов, а также протоколы проверки по пунктам Программы испытаний.*** В соответствии с методиками испытаний средства автоматизации должны обеспечивать всю полноту проверок характеристик по каждому разделу методик. Результаты испытаний фиксируются в протоколах, которые обычно содержат следующие разделы:

— назначение тестирования и раздел требований технического задания, по которому проводились испытания;

— указания разделов методик, в соответствии с которыми проводились испытания, обработка и оценка результатов;

— условия и сценарии проведения тестирования и характеристики исходных данных;

— обобщенные результаты испытаний с оценкой их на соответствие требованиям технического задания и другим руководящим документам, а также технической документации;

— описание отличий тестовой и реальной эксплуатационной сред;

— описание обнаруженных дефектов и ошибок и рекомендуемых улучшений в испытываемом ПС;

— выводы о результатах испытаний и о соответствии созданного ПС или компонента определенному разделу требований технического задания и исходных спецификаций.

Протоколы по всей программе испытаний ***обобщаются в акте,*** в результате чего делается заключение о соответствии системы требованиям заказчика и о завершении работы с положительным или отрицательным итогом. При выполнении всех требований технического задания заказчик обязан принять комплекс программ, и работа считается завершенной.

Наиболее полным и разносторонним испытаниям должна подвергаться первая базовая версия ПС. При ***испытаниях очередных модернизиро-***

ванных версий ПС возможны значительные сокращения объемов тестирования апробированных повторно используемых компонентов. Однако комплексные и завершающие испытания каждой новой версии ПС, как правило, проводятся в полном объеме, гарантирующем проверку выполнения всех требований модифицированного технического задания. Для выявления дефектов в процессе эксплуатации серийных образцов в каждом из них должен быть предусмотрен некоторый минимум средств для проверки функционирования и обнаружения искажений результатов. Эти средства должны позволять фиксировать условия неправильной работы программ и характер проявления дефектов при применении ПС. Последующее исправление ошибок должно проводиться специалистами, осуществляющими сопровождение.

До начала квалификационных испытаний ПС подлежат проверке и ***паспортизации средства***, обеспечивающие получение эталонных данных, средства имитации тестов от внешних объектов, средства фиксирования и обработки результатов тестирования. Завершаются квалификационные испытания ПС предъявлением заказчику на утверждение ***комплекта документов***, содержащих ***результаты комплексных испытаний*** версии программных средств:

— откорректированные тексты программ и данных на языке программирования и в объектном коде, а также полные спецификации требований на программные компоненты и ПС в целом после полного завершения тестирования и испытаний;

— Программу испытаний ПС по всем требованиям технического задания;

— комплект методик испытаний и обработки результатов по всем разделам программы испытаний;

— тесты, сценарии и генераторы тестовых данных, использованные для испытаний программных и информационных компонентов и версии ПС в целом;

— результаты и протоколы квалификационного тестирования, функциональные и конструктивные характеристики ПС в реальной внешней среде;

— отчет о подтверждении заданного качества, полные характеристики достигнутого качества функционирования, а также степени покрытия тестами спецификации требований к ПС;

— план, методики и средства автоматизации обучения заказчика и пользователей применению испытанной версии ПС;

— комплект эксплуатационной документации, описание ПС и руководство пользователя в соответствии с условиями контракта;

— технические условия на версию ПС, базу данных и эксплуатационную документацию для тиражирования и серийного производства;

— руководство по инсталляции, генерации пользовательской версии ПС и загрузке базы данных в соответствии с условиями и характеристиками внешней среды;

— отчет о технико-экономических показателях завершенного проекта версии ПС, выполнении планов и использованных ресурсах;

— акт о завершении испытаний и готовности к поставке и/или предъявлению для сертификационных испытаний версии ПС.

Представленная выше организация испытаний крупных ПС ориентирована на наличие конкретного заказчика комплекса программ и ограниченное число пользователей, контролируемых заказчиком. Несколько иначе организуются испытания коммерческих пакетов прикладных программ, создаваемых по инициативе фирмы или коллектива разработчиков для продажи широкому кругу пользователей при отсутствии конкретного заказчика. Для таких коммерческих комплексов программ принято проводить квалификационные испытания на соответствие критериям, формализованным руководителем проекта в два последовательных этапа — ***Альфа- и Бета-тестирование***. Они заключаются в нормальной и форсированной (стрессовой) опытной эксплуатации конечными пользователями оформленного программного продукта в соответствии с эксплуатационной документацией и различаются количеством участвующих пользователей и уровнем их квалификации.

При Альфа-тестировании привлекаются конечные пользователи, работающие преимущественно в той же компании, но не участвовавшие непосредственно в разработке комплекса программ. Для Бета-тестирования привлекаются добровольные пользователи (потенциальные покупатели), которым бесплатно передается версия ПС для опытной эксплуатации. При этом особое значение имеет участие компетентных, заинтересованных и доброжелательных пользователей, способных выявить дефекты и своими рекомендациями улучшить качество тестируемых программ. Их деятельность стимулируется бесплатным и ранним получением и освоени-

ем нового программного продукта и собственной оценкой его качества. Эти пользователи обязуются сообщать разработчикам сведения о всех выявленных дефектах и ошибках, а также вносить изменения в программы и данные или заменять версии исключительно по указаниям разработчиков. Только после успешной эксплуатации и Бета-тестирования ограниченным контингентом пользователей руководителем проекта или фирмы разработчиков может приниматься решение о передаче ПС в продажу для широкого круга пользователей. Обобщенные результаты Бета-тестирования могут использоваться как основа для сертификационных испытаний.

При Альфа- и Бета-испытаниях принято разделять прогрессивное и регрессионное тестирование. Под прогрессивным — понимается тестирование новых программных компонентов для выявления дефектов и ошибок в исходных текстах программ и спецификациях. Регрессионное тестирование предназначено для контроля качества и корректности программ и данных после проведения корректировок. Необходимость и широта регрессионного тестирования определяются тем, что значительная доля изменений после Альфа- и Бета-тестирования, в свою очередь, содержат дефекты и ошибки. Количество тестов и длительность обоих этапов тестирования определяются экспертно разработчиками или руководителем проекта в зависимости от сложности комплекса программ и интенсивности потока изменений.

14.3. Средства для испытаний и определения характеристик сложных комплексов программ

Для обеспечения высокого качества крупных комплексов программ необходимы соответствующие ***проблемно-ориентированные интегрированные системы автоматизации тестирования***, способные достаточно полно заменить испытания программ с реальными объектами внешней среды. При этом высокая стоимость и риск испытаний с реальными объектами почти всегда оправдывают значительные затраты на такие интегрированные системы, если предстоят испытания критических ПС с высокими требованиями к качеству функционирования программ, с длительным жизненным циклом и множеством развивающихся версий. ***Инструмен-***

тальные средства автоматизации процессов тестирования и испытаний программ должны обеспечивать:

— определение тестов — реализацию процесса тестирования разработчиком: ввод тестовых наборов; генерацию тестовых данных; ввод ожидаемых, эталонных результатов;

— выполнение участка тестируемой программы между контрольными точками, для которого средство тестирования может перехватить операторский ввод (клавиатуры, мыши и т.д.) и для которого вводимые данные могут быть отредактированы и включены в последующие тестовые наборы;

— управление тестами и участком программы, для которого средство тестирования может автоматически выполнять тестовые наборы;

— анализ и обработку тестовых результатов — возможность средства тестирования автоматически анализировать тестовые результаты: сравнение ожидаемых и реальных результатов; сравнение файлов; статистическую обработку результатов;

— анализ покрытия тестами исходного кода для обнаружения: операторов, которые не были выполнены; процедур, которые не были вызваны; переменных, к которым не были обращения;

— анализ производительности программы, когда она исполняется: загрузку центрального процессора; загрузку памяти; обращения к специфицированным элементам данных и/или сегментам кода; временные характеристики функционирования испытываемой программы;

— моделирование внешней среды — поддержку процесса тестирования с помощью модели имитации данных из внешних для ПС компонентов информационной системы.

При создании генераторов тестов внешней среды применяется два принципиально различающихся подхода, которые условно можно назвать интегральным и дифференциальным. При ***интегральном*** или эмпирико-статистическом подходе основой является формальное описание входной и выходной информации имитируемого объекта, а также функциональной связи между данными на его входе и выходе. При этом структура объекта и процессы, реализующиеся при реальном функционировании его компонентов, не имеют значения и не моделируются. Исходные данные и характеристики для построения таких генераторов тестов получаются в натурных экспериментах или при исследовании более детальных — дифференциальных моделей.

Дифференциальные или имитационные модели генераторов тестов базируются на описаниях внутренних процессов функционирования компонентов объекта моделирования, его структуры и взаимодействия составляющих. Результаты функционирования таких моделей определяются адекватностью знаний о компонентах и их характеристиках, а также об их взаимосвязях. Для этого необходимы достаточно подробные сведения о всех процессах функционирования компонентов объектов внешней среды, которые, в свою очередь, могут потребовать еще более глубокого моделирования их составляющих.

В отличие от натурного эксперимента моделирование внешней среды и тестов на ЭВМ имеет большие возможности контроля как исходных данных, так и всех промежуточных и выходных результатов функционирования испытываемого объекта. В реальных системах ряд компонентов иногда оказывается недоступным для контроля их состояния, так как либо невозможно поместить измерители контролируемых сигналов в реальные подсистемы, подлежащие тестированию, либо это сопряжено с изменением характеристик самого анализируемого объекта. Преимуществом моделирования внешней среды на ЭВМ является также повторяемость результатов функционирования и возможность исследования большого количества вариантов и сценариев тестирования. В отличие от этого натурные эксперименты зачастую невозможно остановить на некоторой промежуточной фазе или повторить с абсолютно теми же исходными данными. ***Программная имитация внешней среды на ЭВМ позволяет***:

— проводить длительное непрерывное генерирование имитируемых данных для определения характеристик качества функционирования ПС в широком диапазоне изменения условий и параметров, что зачастую невозможно при использовании реальных объектов;

— расширять диапазоны характеристик имитируемых объектов за пределы реально существующих или доступных источников данных, а также генерировать потоки информации, отражающие перспективные характеристики создаваемых информационных систем и объектов внешней среды;

— создавать тестовые данные, соответствующие критическим или опасным ситуациям функционирования объектов внешней среды, которые невозможно или рискованно реализовать при натурных экспериментах;

— обеспечивать высокую повторяемость имитируемых данных при заданных условиях их генерации и возможность прекращения или приостановки имитации на любых фазах моделирования внешней среды.

Одними из наиболее сложных и дорогих имитаторов внешней среды, применяемых для испытаний комплексов программ, являются модели: полета космических аппаратов; диспетчерских пунктов управления воздушным движением; объектов систем противовоздушной обороны; сложных административных систем. Подобные моделирующие испытательные стенды (МИС) проблемно-ориентированы и объем программ, моделирующих в них внешнюю среду, может даже значительно превышать объемы соответствующих испытываемых ПС. Для их реализации выделяются достаточно мощные универсальные ***моделирующие ЭВМ*** (рис. 14.3). Кроме того, для автоматизации разработки программ могут использоваться отдельные специализированные, ***технологические ЭВМ***, что в совокупности образует инструментальную базу для обеспечения всего ЖЦ сложных комплексов программ реального времени на ***объектных реализующих ЭВМ***.

Имитация конкретных тестов с реальными характеристиками, адекватными объектам внешней среды, является основной частью типовых моделирующих стендов. В соответствии с полной номенклатурой и реальными характеристиками объектов создаются их интегральные или дифференциальные модели. Выбор типов моделей зависит от глубины знаний об алгоритмах функционирования объектов, характеристиках их компонентов и обобщенных параметрах работы объекта в целом. Кроме того, существенным для выбора типов моделей является длительность расчета имитированных данных и обеспечение возможности проводить полную имитацию внешней среды на моделирующей ЭВМ в реальном времени с учетом затрат времени на обработку результатов.

При затруднениях реализации реального масштаба времени при имеющейся производительности ЭВМ следует выделять модели объектов, на которые отсутствуют воздействия обратной связи от испытываемого ПС. Для таких объектов возможно основную часть имитации производить предварительно вне реального времени с регистрацией у каждого тестового сообщения значений реального времени, когда его следует выдать на обработку ПС. Модели объектов, результаты которых зависят от функционирования испытываемого ПС, иногда можно упрощать путем выделения алгоритма реализации обратной связи в реальном времени и дополнения

этими поправками основной части тестов, подготовленных вне реального времени.

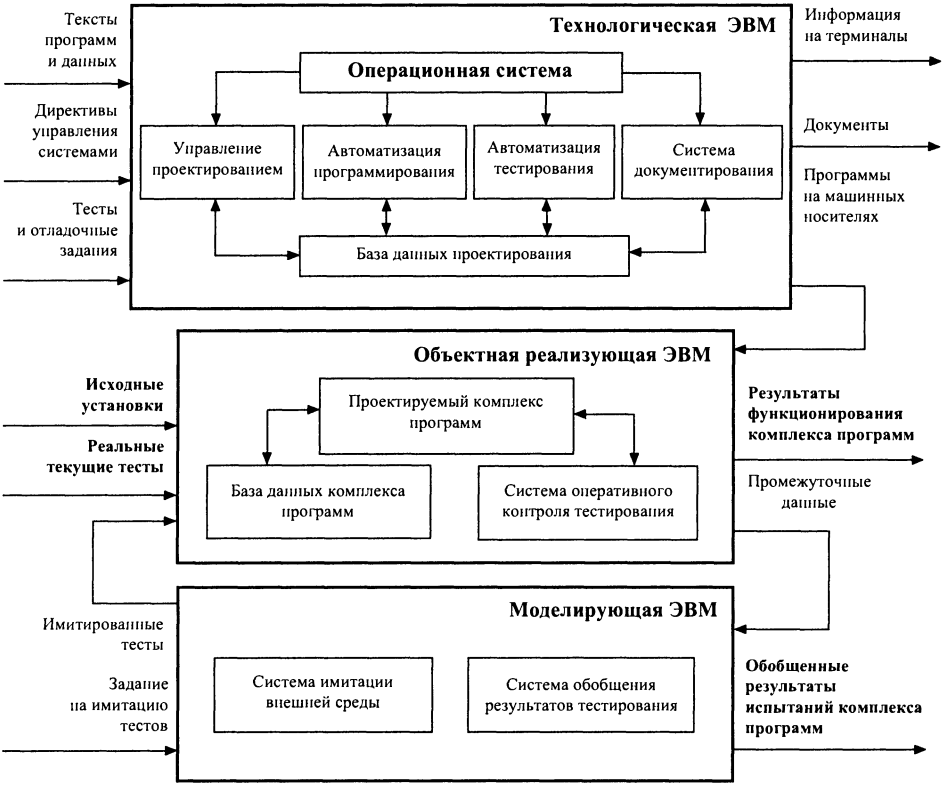

Рис. 14.3

Трудность адекватного моделирования некоторых объектов внешней среды, особенно если при их функционировании активно участвует оператор-пользователь, не позволяет сосредоточить и полностью автоматизировать для крупномасштабных ПС всю имитацию тестовых данных на моделирующих ЭВМ. Поэтому при реализации интегрированных проблемно-ориентированных МИС для испытаний качества функционирования сложных ПС приходится использовать аналоги реальных объектов внешней среды для формирования части данных. Разумное ***сочетание части***

реальных объектов внешней среды и имитаторов на ЭВМ обеспечивает создание высокоэффективных МИС с комплексными моделями совокупностей объектов внешней среды, необходимых для испытаний качества ПС в реальном времени. Такие стенды позволяют автоматическую генерацию тестов с помощью имитаторов на ЭВМ и аналогов реальной аппаратуры дополнять реальными данными от операторов пользователей, контролирующих и корректирующих функционирование системы обработки информации.

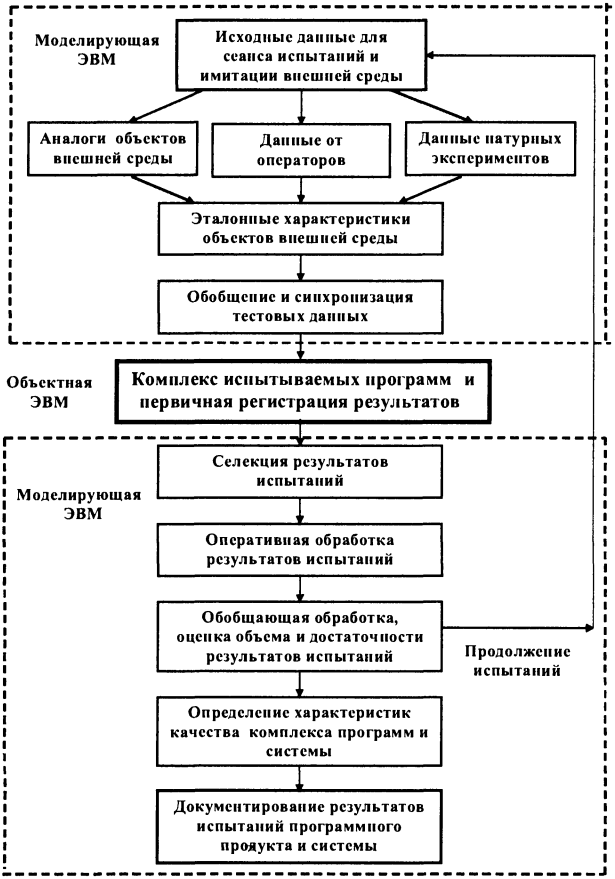

Рис. 14.4

В схеме типового МИС можно выделить ряд базовых компонентов, назначение и функции которых представлены на рис. 14.4. Для каждого эксперимента по испытаниям ПС реального времени следует подготавливать план сценариев тестирования и ***обобщенные исходные данные.*** В моделирующей ЭВМ план и обобщенные исходные данные преобразуются в конкретные значения параметров для задания функционирования каждого имитатора или реального объекта внешней среды. Эти данные вводятся и преобразуются на моделирующей ЭВМ вне реального масштаба времени и подготавливают старт сеанса функционирования стенда и испытываемого ПС в реальном времени. После этого начинают генерироваться тестовые данные.

Аналоги объектов внешней среды используются преимущественно для генерации тестов, представляющих коррелированные логические переменные, которые трудно описать и смоделировать на ЭВМ. Кроме того, они позволяют проверить и аттестовать некоторые программные имитаторы внешней среды, которые впоследствии играют основную роль при испытаниях. В ряде случаев такие аналоги не могут отразить все особенности объектов внешней среды, и имитаторы на ЭВМ остаются единственными источниками соответствующей части данных для проверки качества ПС.

Данные с рабочих мест операторов-пользователей должны отражать реальные характеристики воздействий на тестируемое ПС с учетом особенностей и квалификации человека, которому предстоит использовать испытываемые программы в реальной системе обработки информации. На эту часть МИС кроме первичных исходных данных от моделирующей ЭВМ могут вводиться данные обработки ряда тестов испытываемой системой. В результате через человека и его характеристики замыкается контур обратной связи ручного и автоматизированного управления объектами внешней среды. Такое же замыкание контура автоматизированного управления возможно в аналогах и имитаторах реальных объектов. Иногда необходимо взаимодействие реальных операторов-пользователей и испытываемого ПС непосредственно через штатные устройства сопряжения и визуализации информационной системы. В этих случаях для обеспечения испытаний приходится применять регистрацию данных, поступающих на ПС и выдаваемых программами на реальные объекты. Это дает возможность контроля и обработки тестовых данных либо на моделиру-

ющей ЭВМ, либо на отдельной ЭВМ после записи на промежуточные носители вне реального времени.

Данные натурных экспериментов с объектами внешней среды могут подготавливаться заранее вне сеансов испытаний ПС, например, при отладке аппаратной части системы обработки информации. Эти данные отражают характеристики и динамику функционирования объектов, которые трудно или опасно подключать для непосредственного взаимодействия с недостаточно проверенным ПС. Кроме того, такие данные могут использоваться для аттестации адекватности имитаторов некоторых объектов внешней среды. Они полезны в тех случаях, когда создание определенных условий функционирования объектов внешней среды очень дорого или опасно и может быть выполнено только в исключительных случаях. Однако данные натурных экспериментов не всегда удается адекватно описать условиями и обобщенными характеристиками их поведения. Наличие ряда случайных неконтролируемых факторов усложняет картину исходных данных и может затруднять сопоставление результатов натурных экспериментов с полученными при программной имитации тестов. В этом случае невозможно изменять и учитывать обратную связь от испытываемого ПС.

При тестировании в ряде случаев необходимо иметь ***эталонные характеристики данных***, поступающих на испытываемое ПС. При работе с реальными объектами зачастую приходится создавать специальные измерительные комплексы, которые определяют, регистрируют и подготавливают к обработке все необходимые характеристики в процессе реального функционирования этих объектов (например, координаты движения самолетов при испытаниях систем УВД). Такие измерения проводятся при автономном функционировании объектов или при их взаимодействии с ПС в реальном времени. Результаты измерений используются для определения характеристик качества ПС при работе с реальными объектами.

Имитация эталонных характеристик объектов внешней среды служит для определения качества функционирования ПС в идеальных условиях — при отсутствии искажений исходных данных, ошибок в измерениях их параметров, сбоев и предумышленных отказов. Проверка при таких исходных данных позволяет оценить характеристики дефектов и ошибок результатов, обусловленные недостаточным качеством ПС. Затем эталонные данные объединяются с определенными калиброванными иска-

жениями и дефектами исходных данных, что обеспечивает подготовку тестов с динамическими и статистическими характеристиками, максимально приближающимися к реальным. На промежуточных стадиях проверки с эталонными характеристиками помогают разделять причины дефектов результатов, зависящие от искажений исходных данных и от качества испытываемых программ обработки информации.

Синхронизация и обобщение тестовых данных предназначены для упорядочения тестов от источников различных типов в соответствии с реальным временем их поступления на ПС и для распределения между моделирующей и объектной ЭВМ. В результате формируются потоки тестовых данных каждого реального объекта внешней среды, которые вводятся в объектную ЭВМ в соответствии с логикой функционирования системы обработки информации через соответствующие устройства сопряжения с моделирующей ЭВМ.

Повторяемость сеансов испытаний при автоматической имитации тестов обеспечивается фиксированием всех исходных данных и применением программного формирования псевдослучайных чисел. При надежной работе аналогов реальных объектов и моделирующей ЭВМ, в принципе, можно добиться почти абсолютной повторяемости весьма длительных экспериментов и сценариев тестирования. Некоторая неидентичность результатов при повторных экспериментах может быть обусловлена сбоями и частичными отказами аппаратуры. Труднее обеспечивать повторяемость сценариев испытаний, в которых активно участвует оператор-пользователь. В этом случае необходимо регистрировать действия оператора в зависимости от времени, а затем повторять их в соответствии с записанным сценарием. При необходимости временная диаграмма может соблюдаться с точностью около 0,5—1 с, однако ошибки в действиях оператора и вводимых им параметрах могут отличаться в каждом сценарии тестирования. Вследствие этого повторяемость тестов реализуется только статистически.

Регистрация и обработка характеристик тестовых данных должна обеспечивать их контроль на соответствие заданным обобщенным характеристикам каждого объекта внешней среды и исходным данным сеанса испытаний. Часть этих характеристик используется для сопоставления с результатами функционирования ПС при последующем определении показателей его качества. Так проводится процесс испытаний по конечным

результатам функционирования ПС, выдаваемым внешним абонентам и для определения интегральных характеристик качества. Однако для диагностики и локализации отказов, дефектов и ошибок, а также для оценки частных характеристик качества необходимы промежуточные данные исполнения программ на объектной ЭВМ. Для регистрации промежуточных данных следует иметь возможность разорвать процесс естественного исполнения программы на любом заданном операторе или при обращении на запись или чтение к конкретным данным в объектной ЭВМ.

Разрыв исполнения программ для анализа промежуточных данных может производиться методом вставок специальных контрольных программ при подготовке ПС к конкретному сеансу испытаний. Такие вставки при их небольшом числе почти не искажают реальный масштаб времени. Они обычно размещаются в завершающей части отдельных модулей или групп программ и позволяют контролировать и локализовать причины отказов и дефекты с точностью до достаточно крупных участков программ. В точках контроля и разрыва естественного процесса исполнения программ обычно размещаются только операторы ухода на специализированную группу программ регистрации и оперативной обработки промежуточных данных в объектной ЭВМ. Далее эти данные либо накапливаются и предварительно обрабатываются в объектной ЭВМ, либо оперативно передаются в моделирующую ЭВМ для более глубокой обработки.

Селекция результатов испытаний может основываться на стратегии контроля функционирования программ ***снизу вверх***, т.е. от анализа исполнения отдельных операторов программы и далее до стохастических результатов функционирования всего ПС в динамике реального времени. При этом регистрируется избыточное количество данных, из которых затем отбирается минимум, необходимый для анализа. Может использоваться стратегия ***сверху вниз***, т.е. упорядоченное, иерархическое выделение в первую очередь обобщенных результатов функционирования программ с последующим уточнением регистрируемых и анализируемых результатов вплоть до детального контроля исполнения отмеченных программных модулей и отдельных операторов. В этом случае регистрируются только те данные, которые необходимы для анализа в конкретном сеансе тестирования. При обеих стратегиях необходимо иметь возможность управлять объемом и видом выделяемой и регистрируемой информации тестирования в зависимости от целей испытаний. Данные, получаемые и выделяемые в

процессе испытаний качества ПС, целесообразно делить на следующие ***группы***:

— данные, характеризующие исходную тестовую информацию и выходные результаты тестирования;

— маршруты исполнения программных компонентов и их операторов при некоторых фиксированных тестовых данных;

— аномальные события, сбои, отказы и данные, характеризующиеся отклонением результатов тестирования от эталонов за допустимые пределы и ограничения;

— характеристики использования различных ресурсов объектной ЭВМ.

Регистрация промежуточных данных обычно соответствует некоторым достаточно завершенным этапам функционирования ПС. Вызовы регистрирующих программ должны подчиняться определенной системе контроля динамического функционирования ПС при исходной гипотезе, что ***некоторые ошибки и дефекты в программах и данных могут проявиться на любой стадии тестирования.*** Однако количество вызовов регистрирующих программ и контроль промежуточных результатов, требующих нарушения целостности исполнения функциональных программ, следует ограничивать, учитывая допустимые расходы ресурсов времени на их реализацию. Так как основная задача регистрации при тестировании в реальном времени состоит в обнаружении и локализации ошибок и причин отказов с точностью до функциональной группы программ или модуля, то более точное определение места дефекта следует переносить на тестирование в статике вне реального времени.

Так как испытания современных крупномасштабных систем обработки информации позволяют получать такое большое количество контрольных данных, что достаточно полный их анализ представляет трудную методическую и техническую задачу, ***обработка результатов должна осуществляться иерархически и дифференцированно***. При избытке контролируемых величин снижается общее быстродействие имитаторов и ПС в результате затрат времени на контроль и регистрацию. Это затрудняет анализ качества функционирования программ в реальном времени. При переходе к массовым экспериментам испытаний качества приходится значительно сокращать количество анализируемых параметров и по возможности представлять их в обобщенном виде. В каждом конкретном

случае необходимо стремиться к компромиссу между полнотой регистрации промежуточных данных тестирования и удобством анализа обобщенных результатов.

Обработка результатов испытаний ПС реального времени может быть разделена на две достаточно автономные части: оперативную и обобщающую. ***Оперативная обработка результатов тестирования*** должна производиться по упрощенным алгоритмам с большой пропускной способностью, обеспечивающим сохранение реального масштаба времени для всего испытываемого комплекса программ. Основная часть оперативной обработки результатов связана с замыканием контура обратной связи для имитации динамики функционирования управляемых объектов внешней среды. Оперативно следует производить также селекцию некоторых результатов тестирования и их предварительную обработку для значительного сокращения объема сохраняемых результатов.

В оперативную обработку целесообразно включать расчет части интегральных данных, позволяющих контролировать текущий процесс обработки информации испытываемым ПС. Желательно выделять, регистрировать и отображать критические значения параметров или ситуации, угрожающие надежности и безопасности функционирования ПС. Объем таких оперативно отображаемых данных должен быть максимально сокращенным и в то же время достаточным для анализа критических ситуаций, отражающихся на качестве функционирования ПС. Эти данные должны позволять специалистам, ведущим испытания, фиксировать условия, при которых проявляются дефекты в функционировании программ, с учетом того, что автоматическая регистрация всегда имеет пробелы в составе фиксируемых параметров.

Обобщающая обработка накопленных результатов испытаний может производиться вне реального времени после завершения одного или серии испытаний. Основная задача при этом состоит в расчете различных интегральных характеристик качества функционирования ПС. При натурных экспериментах с внешними объектами для получения эталонных данных в реальном времени используются специальные ***измерительные комплексы***. Особые трудности при этом могут встретиться в связи с необходимостью совмещать во времени результаты исполнения испытываемых программ и данных, получаемых от внешних измерительных комплексов, информация которых используется как эталонная. Решение этой задачи

возможно путем либо жесткой синхронизации функционирования испытываемой и измерительной систем, либо использованием для них системы единого времени.

Зарегистрированные и обработанные результаты испытаний должны использоваться для установления соответствия полученных характеристик качества заданным требованиям. При выявлении их отклонения от требований технического задания заказчика, спецификаций или декларируемых в документации должны разрабатываться корректировки программ для устранения несоответствия. Для этого все этапы тестирования и испытаний ПС должны быть поддержаны системой конфигурационного управления версиями программных компонентов и базой данных документирования тестов, результатов испытаний и выполненных корректировок программ. ***Средства накопления сообщений*** об отказах, ошибках, предложениях на изменения, выполненных корректировках и оцененных характеристиках качества версий являются основой для конфигурационного управления развитием и совершенствованием комплекса программ.

Примером сложного испытательного стенда и моделей внешней среды является комплекс для проверки программ ***управления полетами воздушных судов и диспетчерских систем в центрах управления воздушным движением***. Для комплексной отладки, тестирования, испытаний и сертификации программ управления воздушным движением (УВД) проводится имитация в реальном времени всей информации, поступающей из внешней среды. Источниками информации для центров УВД являются радиолокационные станции, летный состав на борту воздушных судов (ВС), диспетчеры управления воздушным движением и исходные планы полетов. Вследствие этого необходимо имитировать ряд разнородных объектов с учетом интенсивных случайных воздействий, а также при наличии управления со стороны диспетчеров центра УВД и летного состава на борту ВС. Некоторые редкие проявления ошибок в программах могут компенсироваться диспетчерами, контролирующими функционирование центра УВД. Имитировать реакцию и действия диспетчеров автоматически на ЭВМ очень трудно, так как они в значительной степени зависят от квалификации и конкретных психологических особенностей поведения диспетчеров при различных ситуациях воздушной обстановки. Поэтому при комплексной отладке и испытаниях ПС центров УВД обычно участвуют реальные диспетчеры и их средства управления.

Приведенные выше требования и рекомендации по функциям и применению МИС ориентированы на создание крупномасштабных комплексов программ, их тестирование и испытания, в основном до передачи в регулярную эксплуатацию. После приемки заказчиком или приобретения пользователями в процессе функционирования и применения ПС должно обеспечиваться их регулярное тестирование и оценка текущего качества. Для этого в ***составе комплекса программ необходимы средства, обеспечивающие:***

— генерацию тестовых наборов или хранения тестов для контроля работоспособности, сохранности и целостности ПС при функционировании и применении;

— оперативный контроль и обнаружение дефектов исполнения программ и обработки данных при использовании ПС по прямому назначению;

— реализацию процедур предварительного анализа выявленных дефектов и оперативное восстановление вычислительного процесса, программ и данных (рестарт) после обнаружения аномалий функционирования ПС;

— мониторинг, накопление и хранение данных о выявленных дефектах, сбоях и отказах в процессе исполнения программ и обработки данных.

Средства генерации тестов и имитации внешней среды в составе комплекса программ предназначены для оперативной подготовки исходных данных при проверке различных режимов функционирования в процессе применения ПС и при диагностике проявившихся дефектов. Минимальный состав средств генерации тестов должен передаваться пользователям для контроля использования рабочих версий ПС в реальном времени и входить в комплект поставки каждой пользовательской версии. Для размещения таких средств мониторинга и контроля качества функционирования ПС необходимы ресурсы внешней и оперативной памяти, а также дополнительная производительность ЭВМ. Более глубокие испытания функционирования версий и локализации ошибок следует проводить на базе комплекса средств имитации внешней среды высшего уровня (МИС) на моделирующей ЭВМ, которые используются специалистами по испытаниям и сертификации. Часть этих средств имитации может применяться как средства нижнего уровня (пользовательские) на объектной ЭВМ для диагностики и обеспечения полного повторения ситуаций, при которых пользователем могут быть обнаружены дефекты функционирования.

Важной функцией испытательных стендов является их использование в качестве ***тренажеров для операторов-пользователей***. Так как качество функционирования ПС может существенно зависеть от характеристик конкретного человека, участвующего в обработке информации, то необходимо измерять эти характеристики. Необходимо также иметь возможность их улучшать до уровня, обеспечивающего выполнение заданных требований к ПС. Поэтому в процесс испытаний ПС органически входит процесс тренировки и измерения характеристик реальной реакции операторов, а также использование МИС для обучения и регулярной подготовки операторов-пользователей в процессе тиражирования и эксплуатации ПС. Кроме того, испытательный стенд может служить прототипом для разработки тренажеров в серийных системах обработки информации.

Автоматизированная имитация тестов и применение МИС может не только значительно повышать качество разрабатываемого ПС, но и снижать трудоемкость его создания. Даже приближенные оценки соотношения совокупных затрат на программную имитацию с затратами на подготовку тестовых данных при реальном функционировании объектов в большинстве случаев показывают высокую рентабельность программных имитаторов внешней среды. В пределе эффективность применения имитаторов внешней среды приближается к отношению затрат в единицу времени на функционирование реальных объектов и на программную имитацию тестовых данных в тех же условиях.

При использовании программных моделей на ЭВМ ***достоверность генерации тестов*** определяется следующими факторами:

— адекватностью имитатора моделируемому объекту внешней среды или источнику информации;

— инструментальной точностью средств, реализующих имитатор внешней среды;

— статистической точностью процесса имитации и объемом тестовых данных, учитываемых при статистическом обобщении результатов тестирования;

— точностью дискретизации имитаторами реальных непрерывных процессов в моделируемых объектах внешней среды.

Важнейшее значение для определения характеристик ПС имеет ***адекватность имитаторов***, которая зависит от степени учета второстепенных факторов, характеризующих функционирование реальных объектов или

источников информации, при создании их моделей. Точность моделей на ЭВМ прежде всего определяется алгоритмами, на которых они базируются, и полнотой учета в них всех особенностей моделируемых объектов. Кроме того, на адекватность имитации влияют качество программирования и уровень дефектов и ошибок в программах имитации. Каждый не учитываемый в имитаторе элемент или фактор моделируемой системы необходимо оценивать путем сопоставления частных имитируемых данных с результатами аналитических исследований или с данными, полученными на реальных системах, и определять его возможное влияние на полную требуемую точность модели и генерируемых тестов с учетом других составляющих, отражающихся на достоверности имитации.

Перечисленные факторы, влияющие на ***достоверность генерации тестов*** в МИС, взаимозависимы, и повышение достоверности имитации за счет одного из факторов при ограниченных ресурсах приводит, как правило, к снижению достоверности вследствие влияния остальных. Поэтому важной задачей при создании имитационных моделей является достижение наибольшей суммарной достоверности имитации и определения значений характеристик качества функционирования ПС при сбалансированном влиянии каждого из факторов. Достигаемая достоверность имитации внешней среды, а следовательно, и определения качества функционирования испытываемого ПС, естественно, зависят от ресурсов памяти, производительности и других характеристик ЭВМ, на которой реализуется имитатор. Параметры моделирующей ЭВМ в наибольшей степени влияют на статистическую и инструментальную точности, достигаемые в процессе эксплуатации модели. Адекватность моделей и точность дискретизации зависят от сложности моделирующих алгоритмов и, следовательно, от затрат на разработку имитаторов. Поэтому при создании сложных генераторов тестов необходимо достигать, по возможности, равного влияния отмеченных факторов на суммарную достоверность оценки качества ПС при квалификационных испытаниях.

Опыт разработки крупных ПС показал, что качество методов решения задач на отдельных этапах управления может быть объективно оценено лишь в комплексе со всей цепью управления и имитации необходимой информации: внешней обстановки, характеристик входной информации с учетом ошибок, взаимодействующих и обеспечивающих систем. Это возможно за счет создания имитационно-моделирующих стендов и обеспече-

ния взаимодействия по типовым каналам связи с опытными образцами компонентов систем и их комплексами программ. Такой подход оказался ***наиболее целесообразным, способствующим повышению эффективности опытно-конструкторских разработок для ПС реального времени***.

14.4. Оценивание надежности и безопасности функционирования сложных программных средств

Оценивание надежности ПС включает измерение количественных субхарактеристик и их атрибутов: завершенности, устойчивости к дефектам, восстанавливаемости и доступности-готовности (см. табл. 11.2). При этом предполагается, что в контракте, техническом задании или спецификации требований зафиксированы и утверждены заказчиком определенные значения этих атрибутов и их приоритеты. Измерения проводятся при функционировании готового программного продукта для сопоставления с заданными требованиями и для оценивания степени соответствия этим спецификациям требований.

Значения надежности коррелированны с субхарактеристикой корректность, однако можно достигать высокой надежности функционирования программ при относительно невысокой их корректности за счет сокращения времени восстановления при отказах. Кроме того, надежность ПС можно оценивать косвенно в процессе разработки по прогнозируемой плотности обнаружения скрытых дефектов и ошибок, а также по плотности выявляемых и устраняемых ошибок выходных результатов при тестировании рабочего функционирования комплекса программ (см. п. 13.1). Степень покрытия тестами структуры функциональных компонентов и ПС в целом при отладке может служить ориентиром для прогнозирования их потенциальной надежности (см. п. 13.4). Распределение реальных длительностей и эффективности восстановления при ограниченных ресурсах для функционирования программ может рассматриваться как дополнительная составляющая при оценивании надежности.

Для прямых, количественных измерений атрибутов надежности необходимы инструментальные средства, встроенные в операционную систему или в соответствующие компоненты ПС. Эти средства должны в динамике реального функционирования ПС автоматически селектировать

и регистрировать аномальные ситуации, дефекты и искажения вычислительного процесса программ и данных, выявляемые аппаратным, программно-алгоритмическим контролем или пользователями. Накопление и систематизация проявлений дефектов при исполнении программ позволяет оценивать основные показатели надежности, помогает определять причины сбоев и отказов и подготавливать данные для улучшения надежности ПС. Регулярная регистрация и обобщение таких данных способствуют устранению ситуаций, негативно влияющих на функциональную пригодность и другие важные характеристики ПС.

Прямые экспериментальные методы оценивания интегральных характеристик надежности ПС в ряде случаев весьма трудно реализовать при нормальных условиях функционирования крупномасштабных комплексов программ из-за больших значений времени наработки на отказ (сотни и тысячи часов), которые необходимо достигать при разработке и фиксировать при испытаниях. Сложность выявления и регистрации редких отказов, а также высокая стоимость экспериментов при длительном многосуточном функционировании крупномасштабных ПС приводят к тому, что на испытаниях получаются малые выборки зарегистрированных отказов и низка достоверность оценки показателей надежности. Кроме того, при таких экспериментах трудно гарантировать полную представительность выборки исходных данных, так как проверки определяются конкретными условиями применения данного ПС на испытаниях.

При испытаниях надежности ПС в первую очередь обнаруживаются отказы — потери работоспособности. Однако в большинстве случаев первоначально остается неизвестной причина происшедшего отказа. Для выявления фактора, вызвавшего отказ (первичной ошибки или дефекта) и устранения его причины необходимо, прежде всего, определить, каким компонентом информационной системы стимулирован данный отказ. Наиболее крупными источниками отказов являются частичные физические неисправности или сбои аппаратуры ЭВМ, а также дефекты и ошибки программных средств. Стабильные неисправности аппаратуры диагностируются достаточно просто, соответствующими аппаратными тестами, после чего должен следовать ремонт или замена определенных блоков. Однако при возникновении случайного отказа, после которого происходит автоматически полное восстановление нормального функционирования, во

многих случаях трудно однозначно выявить его первичный источник, особенно при очень редких отказах.

Для диагностики и устранения случайных редких отказов должна быть организована служба их регистрации с максимально полным фиксированием характеристик ситуаций, при которых проявился каждый. Сбои в аппаратуре носят более или менее случайный характер и полное повторение отказовой ситуации маловероятно. Ошибки и дефекты программ содержатся в определенном месте и регулярно проявляются при полном повторении внешних ситуаций. На основе таких признаков и, по возможности, детального описания ситуаций возникновения отказа могут строиться предположения о его причине. Эти гипотезы должны использоваться, прежде всего, для дополнительного, интенсивного тестирования всей информационной системы. Если в аппаратуре не обнаруживается причина отказа, то следует провести углубленное тестирование функционального компонента ПС, в котором, по предположению, может содержаться дефект, вызвавший отказ. Для повышения надежности ПС при высокой наработке на отказ необходима тщательная, систематическая работа специалистов, накапливающих, регистрирующих и анализирующих все отказовые ситуации при функционировании комплекса программ. Эти специалисты должны также регистрировать все проведенные корректировки для прогнозирования причин появления возможных дополнительных источников отказов, вызванных дефектами корректировок.

Для ***выявления тенденции изменения показателей надежности*** их зарегистрированные значения необходимо связывать во времени с моментами корректировки программ и данных. Анализируя корреляцию между значениями надежности и процессом изменения программ, можно выявлять некоторые корректировки, которые содержат ошибки и снижают надежность. Получающиеся при этом показатели позволяют прогнозировать число ошибок, подлежащих исправлению для достижения требуемых значений надежности в зависимости от длительности испытаний. В результате может быть оценена наработка до следующего выявления ошибки или отказа.

При заключительных приемо-сдаточных и сертификационных испытаниях для достоверного определения надежности ПС ***организуются многочасовые и многосуточные прогоны функционирования комплекса программ*** в реальной и/или имитированной внешней среде в условиях

широкого варьирования исходных данных с акцентом на стрессовые ситуации, стимулирующие проявления угроз надежности. Такие прогоны позволяют измерять достигнутые характеристики надежности и определять степень их соответствия требованиям технического задания, а также закреплять их в технических условиях и документации на ПС.

Если интенсивное тестирование программ в течение достаточно длительного времени не приводит к обнаружению дефектов или ошибок, то у специалистов, ведущих испытания, создается ощущение бесполезности дальнейшего тестирования данной программы, и она передается на эксплуатацию (см. п. 13.1). Экспериментальное исследование характеристик сложных ПС позволило оценить ***темп обнаружения дефектов, при котором крупномасштабные комплексы программ передаются на регулярную эксплуатацию***: 0,002—0,005 дефекта в день на человека, т.е. специалисты по испытаниям или все пользователи в совокупности выявляют только около одной ошибки или дефекта каждые два-три месяца использования ПС. Интенсивность обнаружения ошибок ниже 0,001 ошибки в день на человека, т.е. меньше одной ошибки в год на трех-четырех специалистов, непосредственно выполняющих тестирование и эксплуатацию ПС, по-видимому, может служить эталоном высокой надежности для обработки информации ПС. Если функционирование программ происходит непрерывно, то эти показатели соответствуют высокой наработке на обнаружение дефекта или отказа порядка 5—10 тысяч часов и коэффициенту готовности выше 0,99. При использовании этого критерия обычно учитывается календарное время испытаний, включающее длительность непосредственного тестирования как для обнаружения, так и для локализации дефектов, а также длительность корректировки программ и других вспомогательных работ для восстановления нормального функционирования ПС.

Форсированные испытания для оценивания надежности программных средств значительно отличаются от традиционных методов испытаний аппаратуры. Основными факторами, влияющими на надежность ПС, являются исходные данные и их взаимодействие с дефектами и ошибками программ или сбоями в аппаратуре ЭВМ. Поэтому форсирование испытаний надежности осуществляется повышением интенсивности искажений исходных данных и расширением варьирования их значений, а также спе-

циальным увеличением интенсивности потоков информации и загрузки программ на ЭВМ выше нормальной.

Планирование форсированных испытаний должно предусматривать последующий пересчет полученных атрибутов надежности на условия нормального функционирования. Для этого необходимо оценивать надежность испытываемых программ в зависимости от интенсивности искажений данных или от характеристик перегрузки ЭВМ, а также применять способы корректного пересчета получаемых показателей на нормальные условия эксплуатации. При форсированных испытаниях целесообразно выделять следующие ***режимы тестирования***:

— полное искажение, предельные и критические значения ключевых параметров каждого типа внешней информации и воздействий пользователей;

— предельные и критические сочетания значений различных взаимодействующих параметров эксплуатации ПС;

— предельно большие и малые интенсивности суммарного потока и каждого типа внешней информации;

— умышленные нарушения пользователями определенных положений инструкций и рекомендаций эксплутационной документации.

Как вид форсированных испытаний можно рассматривать тестирование и контроль результатов функционирования одних и тех же ПС при увеличении числа испытываемых экземпляров и нормальных исходных данных — ***Бета-тестирование*** (см. п. 14.2). На этапе опытной эксплуатации пользователями некоторого предварительного тиража ПС происходит естественное расширение вариантов исходных данных, если они взаимно независимы. Это увеличивает наборы тестов и тем самым дает возможность оценивать наработки на отказ в сотни и тысячи часов. Они позволяют выявлять и устранять значительное число дефектов за относительно небольшое календарное время и тем самым доводить надежность до требуемого уровня. Однако следует учитывать, что при этом пропорционально возрастает суммарная трудоемкость таких испытаний.

Особым видом форсированных испытаний является целенаправленное тестирование эффективности средств оперативного контроля и восстановления программ, данных и вычислительного процесса ***для оценивания восстанавливаемости***. При таких испытаниях основная задача состоит в оценивании качества функционирования средств автоматического повы-

шения надежности и в измерении характеристик восстанавливаемости. Для этого имитируются запланированные условия функционирования программ, при которых в наибольшей степени стимулируется срабатывание средств программного рестарта и оперативного, автоматического восстановления работоспособности.

Следует особо отметить трудности достижения и оценивания надежности программ, характеризующейся наработкой на отказ ≈ 100 ч. При такой надежности ПС резко возрастают сложность обнаружения возникающих отказов и диагностирования их причин. Наработка на отказ в тысячи часов в ряде случаев достигалась только при эксплуатации и сопровождении сложных ПС в течение нескольких лет. При требовании особо высокой надежности функционирования ПС суммарные затраты ресурсов на ее достижение и оценивание могут увеличиваться на порядок. Однако требующееся увеличение затрат для получения такой высокой надежности ПС в процессе разработки трудно обеспечить практически. Поэтому для ее достижения активно применяются различные методы программной защиты от сбоев и отказов программ (методы оперативного рестарта). Они позволяют замедлить рост затрат ресурсов на разработку при повышении требований к их надежности.

14.5. Оценивание эффективности использования ресурсов ЭВМ программным продуктом

Оценивание ресурсной эффективности состоит в измерении количественных субхарактеристик и их атрибутов: временной эффективности и используемости ресурсов ЭВМ комплексом программ (см. таблицу 11.2). При этом предполагается, что в контракте, техническом задании и спецификации требований зафиксированы и утверждены требуемые значения атрибутов и их приоритетов. В стандарте **ISO 9126:2** эту характеристику качества ПС рекомендуется отражать десятком атрибутов, каждый из которых оценивать для средних и наихудших сценариев функционирования комплекса программ. В таблице 11.3 сохранены только три атрибута, важнейшие для функциональной пригодности, которые наиболее доступны количественным измерениям. Оценивание этих атрибутов может проводиться при функционировании готового программного продукта или рас-

четными методами, при разработке для сопоставления с заданными требованиями и оценки степени соответствия этим требованиям.

Для измерения атрибутов временной эффективности необходимы инструментальные средства, встроенные в операционную систему или в соответствующее ПС. Эти средства должны в динамике реального функционирования программ регистрировать: загрузку вычислительной системы; значения интенсивности потоков данных от внешних абонентов; длительность исполнения заданий; характеристики функционирования устройств ввода/вывода; время ожидания результатов (отклика) на задания пользователей; заполнение памяти обмена с внешними абонентами в различных режимах применения комплекса программ. Значения этих характеристик зависят не только от свойств и функций ПС, но также от особенностей архитектуры и операционной системы ЭВМ. Регулярная регистрация и обобщение таких данных позволяют выявлять ситуации, негативно влияющие на функциональную пригодность, надежность и другие конструктивные характеристики качества ПС.

Потребность в ресурсах памяти и производительности ЭВМ в процессе решения задач может значительно изменяться в зависимости от их свойств, а также от потока, состава и объема исходных данных. Степень использования памяти и производительности ЭВМ в некоторых пределах не влияет на качество решения функциональных задач комплексом программ. При излишне высокой интенсивности поступления исходных данных может ***нарушаться временной баланс*** между длительностью решения полной совокупности задач ПС в реальном масштабе времени и производительностью ЭВМ при решении этих задач — ***нагрузочное тестирование***. Также возможно нарушение баланса между имеющейся в ЭВМ памятью и памятью, необходимой для хранения всей поступившей и обрабатываемой информации. Для выявления подобных ситуаций и определения характеристик ПС в условиях недостаточности ресурсов ЭВМ проводятся испытания при высокой, но допустимой интенсивности поступления исходных данных.

Наиболее сложным является ***оценивание эффективности использования ресурсов производительности ЭВМ в реальном времени***. При этом должна быть определена зависимость качества решения задач от интенсивности поступающей информации различных типов. Основная задача испытаний состоит в определении вероятностей, с которыми будет

нарушаться соответствие между потребностями в производительности для решения всей требуемой совокупности задач и реальными возможностями ЭВМ и других компонентов информационной системы. Если эта вероятность невелика и можно считать допустимым эпизодическое снижение качества за счет получающихся задержек и пропусков в обработке сообщений или заданий, то делается вывод о соответствии производительности ЭВМ функциям данного ПС.

Для оценивания использования ресурсов производительности должны быть ***измерены***:

— реальные значения интенсивностей поступающих исходных данных и заданий на вызов функциональных программ, а также распределения вероятностей этих интенсивностей для различных источников и типов заданий;

— длительности автономного решения отдельно каждой из функциональных задач, обрабатывающей исходные данные или включаемой внешними заданиями, а также периодически;

— загрузка ЭВМ в нормальном режиме поступления сообщений и заданий, а также вероятность перегрузки заданиями различных типов и распределения длительностей перегрузки в реальных условиях;

— влияние пропуска в обработке заданий или сообщений каждого типа и снижения темпа решения определенных задач на функциональную пригодность и другие характеристики качества ПС.

Перечисленные задачи могут быть решены экспериментально в процессе тестирования завершенной разработкой системы, однако при этом ***велик риск***, что производительность ЭВМ окажется недостаточной для решения заданной совокупности задач в реальном времени, что отразится на качестве использования ПС. Кроме того, не всегда условия испытаний или опытной эксплуатации системы соответствуют режимам массового ее применения. Поэтому при оценивании требуется принимать специальные меры для создания реальных, а также контролируемых, наиболее тяжелых по загрузке условий функционирования ПС и внешней среды. Такие критические ситуации могут быть в значительной степени предотвращены в процессе разработки ПС путем расчета длительностей исполнения модулей по тексту программ, и объединения этих характеристик в соответствии со структурой программных компонентов и всего комплекса программ.

Для корректного ***оценивания предельной пропускной способности*** системы с данным ПС необходимо измерять следующие характеристики функциональных групп программ:

— экстремальные значения длительностей их исполнения и маршруты, на которых эти значения достигаются;

— среднее значение длительности исполнения каждой функциональной группы программ на всем возможном множестве маршрутов ПС и его дисперсию;

— распределение вероятностей и значений длительности исполнения функциональных групп программ.

В общем случае для оценивания длительностей исполнения и определения качества функционирования программ в зависимости от загрузки необходимо задавать вероятность каждой комбинации тестовых данных и измерять соответствующую ей длительность. После упорядочения значений длительностей можно получить распределение вероятностей в зависимости от длительностей исполнения. Однако для сложных групп программ весьма трудно определить вероятность каждой комбинации исходных данных. Поэтому на практике в ряде случаев ограничиваются некоторыми средними или наиболее вероятными значениями тестовых данных, а также одним или несколькими сочетаниями исходных данных, при которых ожидаются предельные значения потоков заданий и длительностей исполнения программ, способные негативно отразиться на качестве функционирования ПС.

Влияние таких ситуаций ***перегрузки ЭВМ по производительности*** может быть ослаблено путем применения приоритетных дисциплин оперативной диспетчеризации исполнения заданий на решение функциональных задач. В зависимости от характеристик потоков заданий и предполагаемых длительностей их реализации могут распределяться приоритеты на их решения и тем самым повышаться эффективность использования ограниченной производительности вычислительной системы для определенного комплекса программ. Быстрый рост количества решаемых задач, их сложности и требуемой производительности вычислительных средств стимулировал поиск путей удовлетворения потребностей заказчиков в ресурсах для решения таких задач. Значительное внимание было уделено анализу ***эффективности дисциплин диспетчеризации с относительными и абсолютными приоритетами***. Эти дисциплины активно приме-

нялись при организации вычислений в специализированных, объектных ЭВМ реального времени. Они позволяли повышать эквивалентную производительность ЭВМ на 10—20% по сравнению с бесприоритетными дисциплинами диспетчеризации. Показано, что во многих случаях целесообразно применять при диспетчеризации функциональных задач не более 10—15 уровней приоритета, при загрузке ЭВМ на 80—95% и при значительном различии длительностей и коэффициентов важности (10—100) приоритетных задач. Для практического использования характеристик и методов расчета рационального распределения производительности ВС ***созданы методики и типовые модели, позволяющие анализировать и оптимизировать диспетчеризацию в конкретных системах***. При ограничениях ресурсов вследствие требований минимизации весов и габаритов специализированных, объектных ЭВМ в авиационных, ракетных и космических системах их ***экономное использование остается актуальным***. Кроме того, в некоторых случаях полезно выделение высоких приоритетов для особо важных или коротких задач, например, для обмена с внешними абонентами.

По результатам испытаний могут быть решены перечисленные задачи ***оценивания ресурсной эффективности ПС***, что позволяет анализировать факторы, определяющие необходимую пропускную способность ЭВМ, и разрабатывать меры для приведения ее в соответствие с потребностями. Если предварительно в процессе проектирования производительность ЭВМ не оценивалась или определялась слишком грубо, то велик риск, что доработки будут большими или может понадобиться заменить ЭВМ на более быстродействующую. Это обусловлено, как правило, «оптимизмом» разработчиков, что приводит к занижению интуитивных оценок длительностей решения задач и возможных предельных интенсивностей потоков информации. Длительная регистрация и накопление значений ресурсной эффективности способствуют выявлению ситуаций, при которых проявляются некоторые дефекты функциональной пригодности в ПС.

Достоверность оценивания пропускной способности ЭВМ с конкретным ПС зависит от корректности моделирования потоков внешних сообщений, а также от используемых распределений длительности исполнения программ. Для оценивания ресурсной эффективности при подготовке технического задания и спецификаций требований на ПС ***следует согласовывать с заказчиком модель и характеристики внешней среды***, в которой

будет применяться комплекс программ, а также динамики приема и передачи данных (см. п. 14.3). Эти условия следует детализировать до уровня, позволяющего однозначно определять требуемые значения ***интенсивности решения задач***:

— в среднем, нормальном режиме работы ПС с наибольшим качеством функциональной пригодности;

— в режиме предельной загрузки, реализующемся с определенной вероятностью и с допустимым снижением функциональной пригодности и некоторых конструктивных характеристик качества;

— в режиме кратковременной, аварийной перегрузки, способной критически отражаться на функциональной пригодности, надежности и безопасности применения ПС.

Для определения ***использования комплексами программ временных ресурсов ЭВМ*** полезно применять рекомендации стандарта **ISO 14756** — Измерение и оценивание производительности программных средств компьютерных вычислительных систем. Стандарт ориентирован на оценивание: прикладных программных средств, операционных систем и вычислительных комплексов, включающих все аппаратные и программные средства. Основные рекомендации сосредоточены в двух крупных разделах и четырех нормативных приложениях. Раздел 2 содержит общее описание методов измерений, а раздел 3 — детальные процедуры измерений и оценивания производительности ПС в составе информационной системы. Описание метода измерения производительности начинается с эмуляции — имитации пользователей и потоков данных из внешней среды: их случайных характеристик и процессов; функционирования терминалов; установления параметров рабочих нагрузок пользователей и вычислительных средств. ***Процедуры измерений*** должны содержать рекомендации по:

— формированию тестов;

— распределению их по временным фазам;

— определению и регистрации результатов тестирования;

— контролю корректности эмуляции внешней среды;

— статистической обработке измерений.

Оценивание величины производительности рекомендуется для определения загрузки операторов-пользователей, пропускной способности ПС по числу задач в единицу времени, временной шкалы событий обработки заданий и данных. Эти результаты предлагается сравнивать с требования-

ми заказчика и пользователей для оценивания рабочих нагрузок и достаточности производительности ПС в конкретной внешней среде. Детальные процедуры измерений и оценивания распределены по шести подразделам: исходные требования; процессы измерений; результирующие данные; проверка корректности результатов; расчеты производительности; оценивание достоверности измерений производительности. Стандарт рекомендуется для использования: испытателями; разработчиками и покупателями ПС; а также системными интеграторами сложных вычислительных систем.

ЛЕКЦИЯ 15

СОПРОВОЖДЕНИЕ И МОНИТОРИНГ ПРОГРАММНЫХ СРЕДСТВ

15.1. Организация и методы сопровождения программных средств

В процессе эксплуатации версий программного продукта у каждого пользователя могут появляться некоторые претензии к функционированию, которые квалифицируются им как ошибки или дефекты эталонной (базовой) или собственной версии. От пользователей или заказчика могут поступать также предложения по внесению изменений в базовую версию для улучшения эксплуатационных характеристик и расширения функциональных возможностей системы и комплекса программ. Аналогичные предложения могут поступать от разработчиков ПС. Для общения с пользователями и накопления информации о выявляемых недостатках в тиражируемых сложных ПС целесообразно выделение группы специалистов высокой квалификации, овладевших всеми функциями системы и программного продукта.

При организации сопровождения крупных ПС следует учитывать важные ***психологические факторы, усложняющие привлечение и деятельность менеджеров и квалифицированных специалистов*** в этой области:

— эта деятельность требует очень высокой квалификации и больших умственных затрат, связанных, прежде всего, с необходимостью одновременного, широкого охвата и анализа множества компонентов ПС и их взаимосвязей, находящихся в различных состояниях завершенности модификаций;

— корректируемые компоненты зачастую разрабатывались в прошлом в разное время, различными специалистами, в различном стиле и с неоди-

наковой полнотой документирования, что усложняет освоение их содержания при внесении изменений и устранении дефектов;

— сложная, творческая сторона работ при сопровождении вуалируется тем, что приходится овладевать и анализировать программы, разработанные ранее другими специалистами, которые зачастую, может быть, проще не корректировать, а разработать заново;

— комплексы программ, прошедшие широкие испытания и эксплуатацию у заказчиков гарантируют достигнутое качество результатов функционирования, и любые в них изменения имеют высокий риск внесения дополнительных ошибок и ухудшения этого качества, что ограничивает возможность коренных модификаций;

— выполняемые работы требуют особой, скоординированной тщательности корректировок и четкого регламентированного взаимодействия ряда специалистов, различающихся квалификацией и уровнем ответственности;

— процессы и результаты сопровождения не отличаются наглядностью и внешним эффектом, проявлением их размера и сложности, вследствие чего непрестижны среди рядовых программистов и недооцениваются руководителями проектов.

По мере развития применения сложных программных продуктов стало ясно, что интегральные затраты на их сопровождение и создание новых версий ***могут значительно превосходить затраты на разработку*** их первой версии. Опыт последних лет показал, что во многих случаях для сопровождения и мониторинга версий необходимо практически такое же, или даже большее, число специалистов, чем разработало первую версию ПС. При создании сложных ПС перемещение специалистов с разработки новых программных компонентов и ПС на развитие и сопровождение версий имеет систематический характер. Это приводит к тому, что по мере накопления эксплуатируемых ПС и их компонентов все большее число специалистов переходит из области непосредственного программирования новых программ в область системного проектирования и создания новых версий ПС ***на базе повторно используемых компонентов***.

Только после завершения создания нескольких версий ПС может прекратиться переход дополнительных кадров в сферу сопровождения и управления конфигурацией и установиться стабильное соотношение между числом специалистов, занятых первичной разработкой новых проектов и

сопровождением версий ПС. Очень часто разработчики нового ПС не предусматривают этот процесс и требующиеся ресурсы, что значительно снижает эффективность последующего применения созданного программного продукта. По некоторым оценкам, непосредственным программированием ***новых компонентов*** в мире занято около 15—20% специалистов, участвующих в создании программных продуктов.

Целью сопровождения являются выявление и устранение обнаруженных дефектов и ошибок в программах и данных, введение новых функций и компонентов в ПС, анализ состояния и корректировка документации, тиражирование и контроль распространения версий ПС, актуализация и обеспечение сохранности документации и физических носителей — рис. 15.1. Основная задача — изменить и улучшить существующий программный продукт, сохраняя его целостность и функциональную пригодность. Для сохранения и повышения качества сложных комплексов программ ***необходимо регламентировать процессы модификации и совершенствования программных средств***, а также поддерживать их соответствующим тестированием и контролем качества.

Широкое применение прототипирования и повторного использования готовых апробированных программных компонентов способствовало превращению сопровождения в особый раздел методов и средств обеспечения жизненного цикла ПС. Технология сопровождения должна обеспечивать координированное развитие множества версий ПС и их компонентов, каждая из которых имеет достаточно высокое качество и специфические функции, а также, возможно, различных пользователей. Благодаря этому со временем программные средства должны улучшаться и совершенствоваться как по функциональным возможностям, так и по качеству решения каждой задачи.

Сопровождаемость — возможность регламентированной модификации — является важной характеристикой ПС для заказчиков, поставщиков и пользователей, отражающей возможность и простоту внесения изменений в программный продукт после его ввода в эксплуатацию. Требования к сопровождаемости должны включаться в ***подготовку*** в процессе заказа, а их выполнение следует оценивать в процессе разработки модификаций ПС. Для определения и оценки качества модифицированного программного средства могут быть использованы различные показатели,

качественные и количественные стандартизированные метрики в соответствии с **ISO 9126**.

Рис. 15.1

Сопровождаемость должна быть определена до начала первичной разработки проекта ПС соответствующим ***соглашением между заказчиком и разработчиком-поставщиком***. Разработчик должен подготовить план обеспечения сопровождения, в котором отражены конкретные методы, соответствующие ресурсы и последовательности работ. Следует опреде-

лить необходимые усилия по обеспечению мониторинга и оценки аспектов сопровождаемости в процессе разработки. Требования к процессам сопровождения определяются группой основных факторов, влияющих на реализацию модификации программных средств, образующих концептуальную цепочку: ***требования на изменения — изменяемые функции — размер (масштаб) изменений — стратегия модификаций — ресурсы, необходимые для их реализации.*** Эта логическая схема обычно используется при последовательном анализе процессов сопровождения сложных ПС. При этом основным критерием оценки сопровождения является ***совершенствование функциональной пригодности*** и улучшение характеристик качества программного продукта.

Основной процесс эксплуатации в жизненном цикле может ***инициировать процесс сопровождения*** ПС путем представления предложений о модификации (изменении) или отчетов о дефектах. Процесс сопровождения программного средства в соответствии со стандартом **ISO 12207** (п. 5.5) и детализацией этого раздела в стандарте **ISO 14764** использует основной стандартизированный процесс разработки комплексов программ и вспомогательные процессы документирования, управления конфигурацией, обеспечения качества, верификации, аттестации, совместного анализа, аудита и устранения дефектов. Организационные процессы управления, создания инфраструктуры и обучения должны определяться сопроводителем в начале каждого проекта сопровождения.

Стоимость процесса сопровождения может составлять значительную (даже наибольшую) часть стоимости жизненного цикла программного продукта. Период значительного изменения размера, функций и характеристик качества в крупных проектах комплексов программ составляет обычно 1—2 года. В результате исследований появилось понятие «критической сложности и расширения размера» модифицируемой части версии ПС при сопровождении. Если при модернизации и выпуске очередной версии размер доработок заметно превышает «критический», то велика вероятность частичного ухудшения характеристик системы или необходимости выпуска нескольких промежуточных версий для устранения ошибок в изменениях и достижения высокого качества проведенной модернизации.

Характеристики, описывающие качественные и количественные требования к сопровождаемости программного средства, устанавливает заказчик. При реализации процессов разработки, эксплуатации и сопровожде-

ния любые обнаруженные дефекты должны быть описаны и проконтролированы посредством процессов, рекомендуемых в стандарте **ISO 14476**. При этом следует подготавливать соответствующие предложения о модификациях или отчеты о выявленных дефектах. В этом процессе также определяют, отражаются ли представленные дефекты на потребности в модернизации программного продукта. Процесс управления конфигурацией (УК) регистрирует и документирует состояния предложений о модификациях или отчетов о дефектах.

Заказчик может заключить соглашение с разработчиком базовой версии ПС об ***организации сопровождения*** или выбрать в качестве сопроводителя третью сторону (помимо разработчика) (см. рис. 15.1). Сопровождение может также быть проведено по соглашению между двумя сторонами внутри одного предприятия. Эти положения должны быть использованы независимо от того, принадлежит ли заказчик или поставщик к одному или к разным предприятиям.

Передача программного средства для сопровождения является контролируемой и координируемой последовательностью действий, при выполнении которых разработанный продукт переходит от предприятия, выполнившего его первоначальную разработку, к специалистам или предприятию, проводящему его сопровождение. В процессе передачи должны быть отражены:

— требования к передаче технических и программных средств, данных и знаний (опыта) от разработчика к сопроводителю;

— задачи сопроводителя, необходимые для реализации стратегии сопровождения программного продукта: комплектование персонала, его обучение, ввод в действие версий программного продукта, распространение опыта по сопровождению.

Сопроводители иногда встречаются с необходимостью сопровождать программный продукт ***с минимальным набором документов*** или даже при отсутствии их. При отсутствии необходимых документов сопроводитель должен их создать, что является обязательной частью полного корректного сопровождения. В подобной ситуации сопроводитель при подготовке к сопровождению должен:

— определить проблемную область (тип программного продукта);

— изучить любые доступные документы, по возможности обсудить программный продукт с разработчиками и поработать с данным продуктом;

— изучить структуру и организацию программного продукта;

— провести его инвентаризацию, подвергнуть продукт управлению конфигурацией, выстроить продукт в соответствии с библиотеками управления конфигурацией, создать сценарии и деревья вызовов и проанализировать структуру данного продукта;

— определить функции, реализуемые программным продуктом; по возможности рассмотреть технические требования (спецификации) к данному продукту, его общую структуру, проанализировать деревья вызовов, прочитать программные коды, предоставить данный продукт другим сопроводителям и прокомментировать программные коды;

— установить приоритеты предложений о модификации.

Сопроводители должны документально описать программный продукт в соответствии с приведенными рекомендациями. Должны быть обновлены или разработаны (при необходимости) следующие документы: технические требования (спецификации), руководства специалиста по сопровождению, руководства пользователя и руководства по вводу в действие и инсталляции.

Сопроводитель и заказчик должны ***заключить договор на сопровождение*** и указать в нем возможные процедуры внесения изменений в сопровождаемые программные продукты (см. рис. 15.1). Процедуры могут быть использованы как разработчиком оригинала, базовой версии ПС, так и независимым сопроводителем и ***охватывать***:

— основные требования и правила, используемые для определения того, когда ПС может быть локально откорректировано, а когда необходима новая базовая версия программного продукта с использованием для ее подготовки и инсталляции процесса разработки;

— описания типов редакций версий, в зависимости от частоты их появления или влияния на эксплуатацию программного продукта (например, экстренные редакции, периодические редакции);

— способы информирования заказчика о состояниях текущих или намечаемых изменений;

— методы, подтверждающие невозможность появления дополнительных проблем и дефектов в связи с внесением конкретных изменений в данное программное средство;

— классификацию типа изменения, его очередности (приоритетности) и взаимосвязи с другими предложенными изменениями.

Для реализации изменений должен планироваться и использоваться основной процесс разработки ПС и его компонентов, требования к которому ***дополнены***: установленными и документально оформленными критериями проведения испытаний модификаций, оценки их результатов, а также оценки измененных и неизмененных объектов (программных модулей, компонентов и элементов конфигурации); а также полнотой и правильностью реализации новых и измененных требований, чтобы исходные, неизмененные требования к программному продукту не исказились.

Персонал сопровождения должен проводить проверку внесенного изменения совместно с заказчиком, утвердившим модификацию в целях подтверждения функциональной пригодности и работоспособности откорректированного программного продукта, и получить подтверждение того, что внесенное изменение удовлетворяет требованиям, установленным в договоре.

Спецификация требований на изменения программного средства должна исчерпывающе и однозначно описывать обязательные требования к программному средству и к его модификациям и отражать характеристики качества, требуемые стандартами. При этом должны быть учтены следующие ***факторы, влияющие на сопровождаемость:***

— определение и описание новых функций;

— точность и логическая организация данных;

— интерфейсы (системные, компонентов и пользователей), особенно новые и перспективные интерфейсы;

— требования к функциям и рабочим характеристикам, включая влияния корректировок и дополнений;

— требования, налагаемые запланированной внешней средой;

— план обеспечения качества модифицированного программного продукта, в котором особое внимание должно быть уделено документам на изменения и их согласованность.

Разработку концепции сопровождения целесообразно начинать с формализации и обоснования набора исходных данных, отражающих общие особенности класса, назначения и функции ПС, потребителей и этапов жизненного цикла проекта, каждый из которых влияет на выбор определенных характеристик изменения комплекса программ (см. рис. 15.1). Для этого первоначально целесообразно использовать классификацию программных средств по стандарту **ISO 12182** и всю базовую номенклатуру

функциональных характеристик и качества, стандартизированных в **ISO 9126**. Их описания желательно предварительно упорядочить по приоритетам с учетом особенностей назначения, сферы модификации и применения конкретного ПС.

На этапе создания концепции и системного анализа следует сформировать цели сопровождения, выбрать методы и алгоритмы модификации основных, функциональных задач, а также сформулировать предварительные критерии качества создаваемых новых программных компонентов и данных. При этом, естественно, встает вопрос о ресурсах, которые потребуются для достижения этих целей, и о возможности их реализации. Целенаправленная и методичная экспертная ***оценка возможного масштаба и ресурсов*** на изменения уменьшает величину ошибок, однако обычно она остается все-таки довольно большой. Для обеспечения рациональной достоверности первичное прогнозирование целесообразно проводить путем экстраполяции на базе накопленных конкретных данных об отдельных аналогичных модификациях ПС.

До завершения разработки новой базовой версии программного продукта могут быть сформулированы только ***приближенные исходные требования,*** отражающие объекты модификаций и условия их создания. Тем не менее экспертный опрос ведущих специалистов позволяет составить первичный сценарий масштаба и условий очередной модификации ПС. Даже качественная классификация и описание характеристик сценариев изменений значительно повышает точность прогнозов спецификаций требований.

В ***концепции сопровождения*** заказчик и специалисты-разработчики должны представить требования, документально оформить планы и процедуры для проведения работ и реализации задач этого процесса. Они должны определить процедуры для получения, документирования и контроля сообщений о дефектах и заявок на внесение изменений от пользователей, а также для обеспечения обратной связи с пользователями. Всякий раз, когда возникают проблемы (дефекты), они должны быть документально оформлены и введены в процесс решения. Для анализа и ликвидации их следует реализовать процесс управления изменениями и конфигурацией существующего ПС и определить организационные процедуры взаимодействия с данным процессом. Необходимо проанализировать сообщение о дефектах и заявках на внесение изменений по их влиянию на

организационные процессы, существующую систему и интерфейсные связи с другими системами и установить:

— корректировку, модернизацию, профилактику или адаптацию к новым условиям;

— размер изменения, стоимость, время на реализацию изменения;

— критичность, влияние на основные функции, производительность, безопасность или защиту.

На основе проведенного анализа персонал сопровождения должен разработать варианты реализации процессов изменения и документально оформить: сообщение о дефекте или заявку на модификацию; результаты их анализа и требования к реализации изменений. Следует согласовать с заказчиком выбранные варианты изменений в соответствии с договором. Сопроводитель должен провести анализ и определить, какие документы, программные модули, компоненты или их версии требуют изменения. Полученные результаты должны быть документально оформлены.

Описание концепции сопровождения должно быть первым шагом при разработке политики сопровождения ПС. Она должна быть разработана при первом выпуске исходного ***программного продукта и отражать***:

— область сопровождения и изменений программного продукта;

— практическое применение (адаптацию) данного процесса;

— определение предприятия и лиц, ответственных за сопровождение;

— оценку стоимости и длительности сопровождения.

Область сопровождения должна определять обязанности сопроводителя и какую поддержку программному продукту он обязан обеспечить. Она зачастую определяется наличием соответствующих ***ресурсных и бюджетных ограничений и должна охватывать***:

— типы допустимых изменений и процедур сопровождения;

— уровень качества сопровождаемых документов;

— реакцию (чувствительность) пользователей на сопровождение;

— обеспечиваемый уровень обучения персонала сопровождения;

— обеспечение поставки модифицированных версий программного продукта;

— возможность организации справочной службы — «***горячей линии***».

Концепция должна учитывать задачи сопровождения программного продукта после его поставки. Важной частью концепции сопровождения является ***определение ресурсов и специалистов*** (физических или юриди-

ческих), отвечающих за сопровождение продукта. Это в равной степени справедливо и в случае внутреннего сопровождения в самой организации. При выполнении сопровождения по соглашению с третьей стороной это должно быть отмечено в концепции сопровождения. Выбор сопроводителя должен быть основан на ***ряде факторов***:

— срок службы программного средства;

— размер первичных и долгосрочных затрат на сопровождение;

— квалификация сопровождающего персонала;

— функциональная пригодность и работоспособность исходной, базовой версии программного продукта;

— программа (график) модификаций и сопровождения;

— знание сопроводителем предметной области применения программного продукта.

Должна быть проведена ***оценка условий финансирования и стоимости сопровождения***. Стоимость зависит от размера области сопровождения. Дополнительными факторами, подлежащими учету, являются стоимость: обучения как сопроводителей, так и пользователей; среды программного инструментария, тестирования и их ежегодного сопровождения. При разработке концепции сопровождения стоимость оценивают на основе ограниченных исходных данных. В дальнейшем эти оценки должны быть уточнены.

Разработку и утверждение в концепции ***спецификаций требований к функциональным характеристикам и качеству программного продукта*** с учетом выполненных изменений целесообразно проводить итерационно. Полная и однократная формализация требований к характеристикам каждой крупной модификации в начале жизненного цикла ПС обычно невозможна, прежде всего, из-за разных представлений заказчика и разработчиков о деталях ее назначения, функций и возможностей реализации при доступных ресурсах. Чем крупнее и сложнее проект ПС и соответственно выше его стоимость, тем тщательнее следует разрабатывать требования к его характеристикам сопровождения и распределять ресурсы на их реализацию.

При первоначальном определении требований к функциональной пригодности и к конструктивным характеристикам заданные заказчиком ***ограничения ресурсов*** не всегда могут учитывать ряд особенностей сопровождения проекта, что обусловит недопустимое снижение (или завыше-

ние) требований к некоторым характеристикам модифицированного ПС. Кроме того, возможно, что некоторые характеристики или их изменения противоречивы или принципиально нереализуемы в данном проекте. В результате ***несбалансированные требования*** и доступные ресурсы проявятся в виде потерь в качестве или в потребности выделения дополнительных ресурсов.

В зависимости от сложности проекта окончательным результатом работ при прогнозировании изменений комплекса программ должны быть детализированные и утвержденные требования к номенклатуре, свойствам и значениям качества программного продукта, которые достаточны для его полноценного сопровождения и последующей эффективной эксплуатации. Эти ***требования закрепляются в концепции, контракте и техническом задании***, по которым сопроводитель впоследствии должен отчитываться перед заказчиком при завершении модификаций. Однако на последующих этапах жизненного цикла и при конфигурационном управлении требования могут изменяться по согласованию между заказчиком и разработчиком, которые чаще всего приурочиваются к подготовке новой базовой версии ПС. Для этого необходим мониторинг функциональной пригодности, масштаба проекта, требований и реализаций характеристик в течение всего ЖЦ ПС.

Принципиальные и технические возможности, точность реализации свойств и измерения значений характеристик ПС, а также общие ресурсы конкретного проекта всегда ограничены в соответствии с их содержанием и возможностями заказчика и разработчиков. Это определяет ***рациональные диапазоны значений каждого изменения***, которые могут быть выбраны в концепции сопровождения ПС на основе требований заказчика, здравого смысла, а также путем анализа пилотных проектов и прецедентов в спецификациях требований реализованных модификаций. При ограниченности ресурсов проекта крупного ПС распределение приоритетов должно становиться более строгим и могут снижаться приоритеты изменений характеристик, для реализации которых ресурсов недостаточно. В результате формируется полный ***набор требуемых функциональных и конструктивных характеристик качества в процессе сопровождения ПС***.

Требования к функциональным характеристикам и качеству, ***утвержденные после проектирования концепции***, могут быть закреплены в техническом задании как обязательные для детального и рабочего проек-

тирования модификаций (см. рис. 15.1). Эти данные могут использоваться при последующем оценивании качества и при их сопоставлении с требованиями в процессе квалификационных испытаний, сертификации модификаций или новой базовой версии программного продукта.

Для заказчика и пользователей при сопровождении может иметь значение не только определение функциональной пригодности, но и оценка потенциального спроса на рынке конкретного программного продукта, а также его ***конкурентоспособности*** с другими аналогичными по функциям ПС с учетом его качества и стоимости. Это обстоятельство может определять необходимость контроля реализации и уточнения требований к отдельным характеристикам не только при сопровождении в ЖЦ ПС, но также для оценивания интегрального качества нового программного продукта, поставляемого на рынок.

15.2. Этапы и процедуры при сопровождении программных средств

В соответствии с требованиями стандарта **ISO 12207** по развитию и модификации программного продукта в жизненном цикле ***должен быть организован процесс его сопровождения*** (см. п. 5.5). Работы, обеспечивающие сопровождение ПС, включают:

— подготовку процесса;
— анализ проблем и изменений;
— внесение изменений;
— проверку и приемку при сопровождении;
— перенос;
— снятие с эксплуатации.

Эти разделы и соответствующие процессы детализированы в стандарте **ISO 14764** и с рядом комментариев изложены ниже. После активизации процесса следует разработать план сопровождения и соответствующие процедуры, а также выделить конкретные ресурсы для сопровождения. После поставки заказчику программного продукта сопроводитель, в соответствии с договором и предложением о модификации или отчетом о дефекте, должен изменить соответствующие программы и документы. Исходные данные преобразуют или используют в работах по сопровожде-

нию для получения выходных результатов — ***модифицированных версий программного продукта***. Рекомендуется проводить регулярный контроль с целью проверки корректности выходных результатов конкретных работ по сопровождению.

Рис. 15.2

При ***подготовке процесса*** сопроводитель должен создать планы и определить процедуры, выполняемые при реализации сопровождения (рис. 15.2). План сопровождения целесообразно создавать параллельно с планом разработки первой, базовой версии ПС. При выполнении данной ра-

боты сопроводитель также должен определить необходимые организационные интерфейсы и взаимосвязи между специалистами и с другими предприятиями. Исходными данными для работ по подготовке процесса являются: старая (исходная) базовая версия программного продукта; системные документы; предложения о модификациях и отчеты о дефектах. Для обеспечения эффективной реализации процесса сопровождения сопроводителю следует разработать и документально оформить ***стратегию проведения сопровождения***, один из ключевых факторов в применении и развитии ПС. При реализации этой деятельности сопроводитель должен: разработать планы и процедуры сопровождения; установить процедуры рассмотрения предложений о модификации и отчетов о дефектах; применить управление конфигурацией.

Сопроводитель должен разработать, документально оформить и выполнить планы и процедуры для проведения работ и решения задач процесса сопровождения. В плане сопровождения следует описать стратегию сопровождения системы, а в процедурах сопровождения должны быть определены подробности выполнения этапов и процессов сопровождения. Для обеспечения создания эффективных планов и процедур сопровождения сопроводитель должен:

— выполнить оценку сопровождаемой системы;

— гарантировать официальное подтверждение принятия на себя обязанностей сопроводителя программного продукта;

— провести анализ доступных ресурсов для сопровождения;

— оценить и согласовать с заказчиком финансирование и стоимость сопровождения;

— установить требования к процессу передачи программного продукта сопроводителю;

— определить подлежащие реализации процессы сопровождения;

— документально оформить процесс сопровождения в виде планов и процедур, согласованных с заказчиком.

Стратегия сопровождения должна быть ориентирована на людские и материальные ресурсы, необходимые и доступные для обеспечения развития и модификаций программного продукта. Политика сопровождения ПС должна охватывать следующие основные компоненты: концепцию сопровождения; план сопровождения; анализ ресурсов. Процесс разработки изменений включает в себя ряд работ, связанных с планированием сопро-

вождения ПС. Эти виды деятельности должны быть определены в стратегии сопровождения программного продукта: определена пороговая величина изменения в стоимостном выражении, позволяющая вносить соответствующее изменение в ПС без пересмотра конкретного договора с заказчиком; соглашения по интерфейсам для всего проекта в части постоянных проблем, связанных с неясностью, неточностью, изменчивостью или непроверяемостью требований заказчика и спецификаций.

Целью планирования сопровождения является подготовка плана работ по сопровождению и обеспечение ресурсов, необходимых для проведения этих работ после передачи программного продукта на сопровождение. Планирование начинается после определения концепции сопровождения ПС и завершается разработкой плана сопровождения, используемого в качестве основы при сопровождении. ***Общий план сопровождения должен определять***:

— причины необходимости сопровождения;

— состав исполнителей работ по сопровождению;

— роли и обязанности каждого субъекта, вовлеченного в сопровождение;

— как должны быть выполнены основные процессы и работы;

— какие имеются и необходимы ресурсы для сопровождения;

— методы и средства организации работ по управлению, выпуску продукта и синхронизации работ;

— перечень всех проектных результатов и продуктов, подлежащих поставке заказчику;

— критерии завершения соответствующей деятельности, работ и задач;

— состав отчетных материалов по этапам, затратам и графикам проведения работ;

— периодичность и способы выдачи отчетных материалов;

— состав отчетных материалов по проблемам и устраненным дефектам;

— время начала и длительность сопровождения.

Рекомендуется сопроводителям формализовать ***конкретный план сопровождения ПС*** из представленного общего состава процессов ЖЦ, который уточнить и адаптировать с учетом объема и особенностей проекта, содержащий разделы:

— описание сопровождаемой системы, в которую входит ПС;

— концепция сопровождения комплекса программ; описание уровня сопровождения системы и ПС; установление длительности процессов сопровождения; адаптация стандартизированных процессов сопровождения;

— организационные работы по сопровождению, роли и обязанности специалистов;

— ресурсы: состав специалистов; инструментальные средства; технические средства; документы и планы;

— процессы — как должна быть выполнена конкретная деятельность;

— определение уровня обучения, необходимого для сопроводителей и для пользователей;

— протоколы и отчеты по сопровождению; контрольные данные, собранные при работах по сопровождению.

Проектирование архитектуры модификаций определяет функции и компоненты модифицированного программного средства. Основными особенностями данной работы среди процессов ЖЦ ПС, влияющими на сопровождаемость, являются выбор структуры программы, разбиение ее на компоненты (модули) и поток данных, циркулирующий между ними. При модификациях важно использовать знания коллектива специалистов по обработке данных, относящиеся к возможности использования частей существующих программ или библиотек, доказавших высокое функциональное качество. Основными средствами, способствующими обеспечению требований сопровождаемости, являются модульная архитектура в сочетании с нисходящим анализом и соответствующие документы, в которые при необходимости могут быть внесены дополнения.

При проектировании ПС создаются версии каждого компонента программного средства, интерфейсов и баз данных. Составляются точные, подробные описания каждой функции для реализации предложенных изменений. Сопровождаемость программного средства может быть улучшена при учете характеристик качества, регламентированных в стандарте **ISO 9126**. Сопроводителю следует определить процедуры для: получения, документирования и контроля отчетов о дефектах и предложений о модификациях от пользователей; обеспечения обратной связи с пользователями. Каждая возникающая проблема и дефект должны быть документально оформлены и введены в процесс анализа изменений, для чего следует:

— разработать схему классификации и присвоения приоритетов для предложенных модификаций и описаний дефектов;

— разработать процедуры проведения целевых анализов изменений;

— определить процедуры представления предложенных модификаций и описаний дефектов оператором;

— определить организацию обратной связи с пользователями при анализе изменений;

— определить, как пользователей будут обслуживать в период реализации сопровождения;

— определить, как будут введены предлагаемые модификации в базу данных учета состояний изменений и используемых ресурсов.

Анализ дефектов и модификаций в стандарте **ISO 14764** рекомендуется реализовать в следующем порядке:

— анализируются предложения о модификации и отчеты о дефектах;

— дублируется или проверяется реальность каждого дефекта;

— разрабатываются варианты реализации изменения;

— документально оформляются: предложения о модификации и отчеты о дефектах, результаты их рассмотрения и варианты реализации изменений;

— проводится согласование выбранного варианта реализации изменения с заказчиком.

До внесения изменений в систему и ПС сопроводитель должен: проанализировать возможные изменения с точки зрения их влияния на деятельность предприятия, существующую систему и взаимосвязанные с ней системы; разработать и документально оформить рекомендуемые альтернативные решения по внесению корректировок и согласовать принятое решение по внесению изменений с заказчиком. Сопроводителю необходимо проанализировать отчет о проблеме — дефекте или предложение о модификации по их влиянию на организационные вопросы, существующую систему и интерфейсные связи с другими системами по типу: корректировка, модернизация, профилактика или адаптация к новым условиям или среде; по масштабу: размеру изменения, стоимости, времени на реализацию изменения; по критичности: влиянию на рабочие характеристики и производительность, безопасность или защиту продукта.

Для обеспечения реализации представленного предложения на изменение сопроводитель должен определить:

— наличие соответствующего персонала, способного реализовать предлагаемое изменение;

— наличие достаточного финансирования для реализации предлагаемого изменения в программе;

— наличие соответствующих ресурсов ЭВМ и степень влияния модификации на реализуемую или уже реализованные версии программного продукта;

— влияет ли отсутствие предполагаемых изменений на требования к системным интерфейсам, ожидаемый срок службы системы, приоритеты;

— влияние изменений на безопасность и защиту системы при эксплуатации;

— единовременные и долгосрочные затраты на корректировку;

— преимущества, получаемые после проведения модификации;

— влияние реализации изменений на графики проведения работ по версии программного продукта;

— необходимые процессы верификации, тестирования и оценки характеристик системы и программного продукта после внесения корректировки.

Для того чтобы подтвердить актуальность представленных отчетов о дефектах, сопроводитель должен ***продублировать и верифицировать*** возникшие проблемы — дефекты, выполнив следующие этапы решения данной задачи: разработать стратегию верификации и квалификационного тестирования для проверки устранения конкретной проблемы — дефекта; провести тестирование для проверки наличия проблемы — дефекта; документально оформить результаты квалификационного тестирования. Если конкретная проблема не может быть повторена сопроводителем, он должен проверить правила, политики и документы обеспечения ЖЦ ПС на предприятии. На основе проведенного анализа сопроводитель должен разработать ***варианты реализации изменения***:

— присвоить соответствующий приоритет проблеме (дефекту) или предложению о модификации;

— установить наличие возможностей и средств для решения проблемы;

— оценить масштаб и трудоемкость данной модификации;

— разработать варианты реализации конкретного изменения;

— определить влияние данных вариантов на функциональную пригодность и технические средства системы;

— выполнить анализы риска для каждого варианта модификации.

Сопроводитель должен реализовать ***процесс управления конфигурацией*** для управления изменениями существующей системы или определить организационный интерфейс с данным процессом (см. лекцию 16). Результатами данной работы являются: план и процедуры сопровождения; процедуры решения проблем и устранения дефектов; планы организации обратной связи с пользователями; план передачи модификаций заказчику и пользователям. До внесения изменений в систему и программный продукт в соответствии с договором с заказчиком сопроводитель должен ***согласовать выбранный вариант корректировки***.

Контроль за рассматриваемыми работами следует проводить посредством процесса совместного анализа. В конце работ должен быть проведен ***анализ риска***. На основании выходных результатов анализа может быть пересмотрена предварительная оценка требуемых ресурсов и с привлечением пользователей или заказчика принято решение о целесообразности перехода к работе по внесению изменений в базовую версию программного продукта. Результатами этой работы являются: анализ влияния изменений; рекомендуемый вариант и согласованные изменения; обновленные и исправленные документы.

При внесении изменений в ПС сопроводитель разрабатывает и тестирует конкретные изменения программного продукта. Исходными данными для проведения работы при внесении изменений должны быть: базовая версия программного продукта; согласованные с заказчиком предложения о модификации; согласованные документы на реализацию корректировки; отчет о влиянии корректировки и выходные результаты работы по анализу изменений. Сопроводитель должен выполнить анализ ***использования процессов разработки комплекса программ*** при внесении изменений. После согласования корректировки сопроводителю следует провести анализ и определить, какие документы, программные модули и их версии требуют изменения. Результаты такого дополнительного анализа должны быть оформлены в комплекте документов процесса разработки базовой версии программного продукта:

— определены компоненты в существующей системе, подлежащие изменению;

— определены компоненты конкретного интерфейса, затрагиваемые данным изменением;

— определены документы, подлежащие обновлению;

— обновлен комплект документов базовой версии программного продукта;

— установлены и документально оформлены критерии проведения квалификационного тестирования и испытаний, оценки их результатов в измененных и неизмененных объектах (программных модулях, компонентах и элементах конфигурации) системы;

— обеспечены полнота и правильность реализации новых и измененных требований, обеспечено, чтобы исходные, неизмененные требования и целостность системы сохранились.

Результаты испытаний корректировок должны быть документально оформлены. Контроль за рассматриваемой работой должен быть проведен посредством процесса совместного анализа. Результатами данной работы являются: обновленные планы, документы и процедуры тестирования; измененные исходные программы; отчеты о квалификационном тестировании; показатели, характеризующие качество внесенных изменений. Обновленные документы должны включать: подробный отчет о проведенном анализе; обновленные требования; обновленные планы, процедуры и отчеты о тестировании; обновленные учебные материалы.

Проверка и приемка модификаций при эксплуатации обеспечивает подтверждение корректности изменений, внесенных в систему, в соответствии с принятыми стандартами и по установленной методологии. Исходными данными для проведения работы по проверке и приемке при сопровождении являются: измененный программный продукт; результаты квалификационного тестирования внесенных изменений. Проверки проводятся для гарантирования правильности изменений и их согласованности с точки зрения выполнения установленных требований заказчика к программному продукту. Сопроводитель должен провести проверки каждого внесенного изменения совместно с заказчиком, утвердившим изменение в целях подтверждения целостности и работоспособности измененной системы:

— отслеживание реализованных предложений о модификации и отчетов о дефектах относительно требований предыдущей базовой версии проекта и программных кодов;

— проверку тестируемости текста (кодов) программы;

— проверку соблюдения стандартов на ЖЦ ПС и системы;

— проверку того, что изменены только нужные компоненты программного средства;

— проверку правильности сборки новых компонентов программного продукта;

— контроль обновления документов версии программного продукта;

— проверку полноты проведения тестирования и отчетов о тестировании.

Сопроводитель должен получить согласование и подтверждение того, что внесенное изменение удовлетворяет требованиям заказчика, установленным в договоре: посредством вспомогательного процесса обеспечения качества; проверки выполнения этого процесса; проведения аудита функциональной и физической конфигурации. Результатами данной работы являются: новая базовая версия программного продукта, включающая в себя принятые изменения; отклоненные изменения; отчет о приемке версии; отчеты о проверках и аудитах; отчет о квалификационном тестировании программного продукта.

Большую роль для успешного внедрения новых версий играет ***психологический аспект***. Благоприятные условия внедрения обеспечиваются там, где имеется нормальное взаимодействие заказчика, пользователей и разработчиков во время создания и изменения версий программного продукта. Это способствует достаточно высокой степени отработки документации, инструментальных средств разработчиками и своевременному уяснению функционального назначения компонентов ПС, его особенностей и новых возможностей пользователями. Основная психологическая трудность состоит в том, что большие коллективы специалистов необходимо перевести на новые методы работы. Особенно большие сложности возникают при внедрении версии программного продукта на стадии опытной эксплуатации, когда значителен поток ошибок по разным причинам (неопытность пользователя, некачественная документация, неотработанная система). Дополнительная трудность может быть связана с наличием определенных ограничений, свойственных применению для сопровождения и реализации изменений новой технологии и инструментальных средств, зачастую отличающихся от привычных.

Сопроводитель должен ***документально оформить*** и представить заказчику:

— отчеты о проблемах (дефектах) и предложения о модификациях; результаты их анализа и варианты реализации изменений;

— результаты приемочных испытаний, верификации, аттестации и измерений характеристик качества новой версии программного продукта;

— отчеты об обеспечении характеристик качества программного продукта и результаты их тестирования;

— результаты аудиторских проверок версии программного продукта;

— замечания заказчика и результаты взаимодействия с ним по устранению дефектов версии программного продукта;

— комплект актуальных проектных документов и документов результатов сопровождения;

— оценки корректности реализованной политики, графика и Программы квалификационного тестирования версии программного продукта;

— соотношение оценок необходимых и использованных ресурсов;

— официальные рекомендации с указаниями о целесообразных последующих модификациях и создании новых версий ПС.

Внедрение новой версии программного продукта для массового применения (см. рис. 15.2) осуществляется, как правило, в два этапа; силами разработчиков модификаций в целях обкатки, проверки и выявления ошибок в изменениях на стадии опытной эксплуатации и посредством использования специализированных коллективов сопровождения для тиражирования и распространения. Основные обязанности сопроводителей сводятся к передаче физических носителей с кодами ПС и комплектом эксплуатационной документации, а также к проведению консультаций для выделенной группы специалистов-пользователей. Сопроводители в этом случае получают возможность непосредственно контролировать работу пользователей с системой и документацией, что обеспечивает высокую оперативность отработки замечаний и рекламаций, формирование квалифицированных предложений для изменений, оценку эффективности применения версии программного продукта. Кроме того, разрабатывается учебно-методический план, подготавливаются учебные пособия, необходимые для обучения пользователей на курсах, а также проводится обучение выделенной группы специалистов, ответственных за последующее обучение коллективов пользователей и сопровождение ПС.

Применение версий программного продукта у пользователей регламентируется установленными правилами и закрепляется соответствующими договорами. Эти договоры определяют порядок поставки, инсталляции, ввода в строй и сопровождения версий ПС, а также ***порядок обучения пользователей***. Наиболее благоприятные условия для успешного внедрения создаются, когда разработка модификаций ПС идет с самого начала проекта по новой технологии. Тем не менее известны примеры подключения новой технологии сопровождения к ПС с определенным ***унаследованным заделом***. Обычно такая ситуация характерна для сложных, эксплуатируемых ПС, подвергающихся серьезной модернизации или развитию, либо когда сроки разработки проекта истекают и предпринимаются попытки повысить эффективность разработки с помощью применения новой прогрессивной технологии сопровождения.

При обучении сопроводителей основное внимание должно уделяться изложению методологических основ и стандартизированных, технологических операций по разработке модификаций программ. Таким образом, обучение специалистов целесообразно вести от технологии и стандартов к инструментальным средствам. Этот подход позволяет наиболее рационально использовать средства автоматизации в процессе разработки изменений ПС различного типа и назначения. Внедрение методологических принципов разработки модификаций программ и технологии сопровождения обеспечивает их унифицированное и эффективное использование разными специалистами, работающими как в пределах одного комплекса программ, так и над разными и независимыми проектами.

Снятие программного средства с эксплуатации и сопровождения должно быть подготовлено анализом, обосновывающим это решение. В анализе следует определить и экономически обосновать: возможность сохранения устаревшей версии комплекса программ, а также необходимость создания и применения новой версии программного продукта. При снятии программного продукта с сопровождения следует определить необходимые для этого действия, а затем разработать и документально оформить этапы работ, обеспечивающие их эффективное выполнение. Должны быть предусмотрены возможности доступа к архивным данным снятого с сопровождения базового программного продукта.

Специалисты, выполняющие снятие программного продукта с сопровождения и эксплуатации, должны разработать план, предупредить пользо-

вателей об этом, провести соответствующее обучение персонала, уведомить всех заинтересованных субъектов о завершении сопровождения и архивировать соответствующие данные. В ***содержание плана*** необходимо включить:

— анализ требований к снятию с сопровождения и эксплуатации;

— оценку влияние снятия с сопровождения программного продукта на систему;

— установить программный продукт, заменяющий снимаемый (при его наличии);

— график и Программу снятия программного продукта с сопровождения и эксплуатации;

— определить и документировать все процедуры по снятию с сопровождения и эксплуатации;

— сроки прекращения полной или частичной поддержки сопровождения;

— требования по архивации версии и модификаций программного продукта и соответствующих документов;

— сроки перехода, при необходимости, к новой версии программного продукта;

— требования по доступу к архивным копиям данных проекта программного продукта.

Для плавного перехода к новой базовой версии программного продукта должна быть обеспечена параллельная эксплуатация прежнего и нового программных продуктов. В течение некоторого периода времени следует провести необходимое обучение пользователей новой версии в соответствии с условиями договора. После выполнения запланированного снятия с эксплуатации должно быть послано соответствующее уведомление всем заинтересованным сторонам. Все связанное с прежней версией ПС: документы разработки, журналы регистрации и программы — должно быть помещено в архивы. Данные, использованные или связанные со снятым с эксплуатации программным продуктом, следует сохранять доступными для аудиторской проверки. Целесообразно также сохранять старые версии ПС и некоторые данные, полученные при решении предыдущих задач в качестве тестов; создавать копии старых программных средств и данных, полученных при решении предыдущих задач; хранить соответствующие носители в безопасном месте.

15.3. Задачи и процессы переноса программ и данных на иные платформы

Многочисленное дублирование, по существу, одних и тех же программных средств и информации баз данных на подобных или разных платформах сопряжено со значительными нерациональными затратами на их разработку и с увеличением длительностей создания информационных систем. Для их сокращения необходимы организация, технология и инструментарий, обеспечивающие ***эффективный перенос*** готовых программ и данных в пределах одной операционной и аппаратной среды или с иных платформ. Для этого создаются методологии и технологии переноса, а также стандарты, поддерживающие процессы и разработку переносимых программ и данных. В каждом конкретном случае ***необходима оценка рентабельности переноса*** с учетом ряда факторов, характеризующих мобильность программ и данных и их среды, по сравнению с полной разработкой аналогичного программного продукта на новой платформе. Использование методического, технологического, алгоритмического и программного задела из предшествующих проектов обеспечивает многократное повышение производительности труда разработчиков систем, сокращение сроков их создания и высокое качество проектов. Под ***мобильностью — переносимостью*** понимаются:

— процессы переноса программ и данных из одной аппаратной, операционной и пользовательской среды в иную по архитектуре и характеристикам среду с сохранением их целостности или небольшими изменениями функций системы;

— процессы повторного использования готовых программных компонентов и средств, а также информации баз данных возможно в пределах одной архитектуры аппаратной и операционной среды для расширения и изменения функций системы и программного продукта.

Основным стимулом развития и применения мобильных программ и данных явилась необходимость улучшения экономических показателей при создании и эксплуатации сложных систем, а также повышения их качества. Объективные требования заказчиков и пользователей по совершенствованию и снижению затрат на информатизацию объектов и процессов отразились на формировании основных ***целей создания и применения мобильных программ и данных***, которые состоят в следующем:

— обеспечение сохранения инвестиций, вложенных в реализованные и апробированные программные продукты и базы данных, в процессе развития, модификации и появления новых требований к ним, а также при совершенствовании архитектур и возрастании ресурсов и функций аппаратных и операционных платформ;

— снижение трудоемкости, стоимости и длительности непосредственной разработки сложных распределенных программных средств и баз данных;

— обеспечение высокого качества, надежности и безопасности функционирования программных средств и баз данных в системах;

— обеспечение возможности эффективного по экономическим показателям и качеству переноса апробированных программных продуктов, разработанных должным образом, с минимальными изменениями на различные операционные системы и аппаратные платформы;

— экономная реализация совместной работы и расширения функций ПС (интероперабельность) во взаимодействии с другими программами и данными при решении единой целевой задачи на различных локальных и распределенных платформах;

— обеспечение взаимодействия с пользователями в унифицированном стиле, облегчающем им переход к использованию новых или с расширенными функциями системам и программным продуктам (мобильность пользователей);

— снижения зависимости заказчиков конкретных систем от определенных поставщиков аппаратных и операционных платформ, а также от разработчиков некоторых программных продуктов.

Для достижения перечисленных целей в обеспечении мобильности ПС требуются различные ресурсы при их реализации. Потребность в конкретных ресурсах и рентабельность их использования зависят от ряда параметров, которые образуют широкий спектр ситуаций для анализа и применения свойства мобильности программ и данных. Такими ***ресурсами являются***:

— трудовые затраты специалистов и время на создание, приобретение и эксплуатацию инструментальных средств, автоматизирующих разработку и сопровождение мобильных ПС и БД;

— трудовые затраты специалистов и время на создание дополнительных интерфейсных компонентов в программах и данных, обеспечиваю-

щих их эффективную мобильность на определенные типы платформ, например, в соответствии с концепцией и стандартами открытых систем;

— дополнительные ресурсы памяти и производительности вычислительных средств, необходимые для реализации и функционирования компонентов в программах и данных, обеспечивающих их высокую мобильность, например, для реализации стандартизированных интерфейсов с внешней и внутренней средой.

Задачи повторного использования и переноса программ и данных охватывают:

— встраивание готового программного средства и информации базы данных в создаваемую новую систему при условии, что их поставщики гарантируют функционирование на выбранной платформе;

— перенос программ и данных с платформ, в среде которых они были ранее реализованы, на выбранную для системы новую платформу;

— обеспечение доступа к информационным ресурсам других распределенных систем и сетей.

При переносе свойства программных продуктов практически всегда изменяются, что следует учитывать при анализе целесообразности переноса, а также могут быть необходимы их отладка, испытания и сертификация в новой среде. Следует также учитывать, что любой ***перенос связан с затратами***, которые чаще всего требуются для:

— системного анализа рентабельности переноса на иную или ту же платформу и оценки технико-экономических показателей этого процесса;

— реализации самого процесса переноса и интеграции с операционной и внешней средой на новой аппаратной платформе или в существующей среде;

— квалификационного тестирования, испытаний и комплексной проверки функционирования программного продукта в новом окружении или на новой платформе;

— сертификации перенесенного на новую платформу продукта и функционирующего в иной операционной и внешней среде;

— корректировки или дополнения эксплуатационной и технологической документации.

Два последних вида работ могут выполняться в процессе создания мобильных ПС и БД и отсутствовать при непосредственной реализации переноса. Однако и в этом случае ***следует избегать излишнего оптимизма***

при оценке затрат на перенос, так как при создании мобильных ПС трудно предусмотреть все возможные особенности различных платформ и внешней среды, для которых декларируется мобильность конкретных средств. Эти особенности и возможное расширение окружающих прикладных программ и данных могут преподносить неприятные сюрпризы нестыковки, для ликвидации которых потребуются дополнительные затраты.

Интеграционные тенденции развития современных ПС и БД связаны с сочетанием в них задач, относящихся к разным классам. Это определяет необходимость применения разных инструментальных средств для реализации программ, относящихся к разным классам, а также необходимость обеспечения функционирования создаваемых разными методами программ в единой целевой системе. Анализ мобильности программ и данных в системах касается различных классов задач, однако далее, для определенности, конкретные методы и средства обеспечения мобильности рассматриваются преимущественно применительно к задачам обработки данных в распределенных системах. При этом ***объектами анализа переносимости являются***:

— программные модули и функциональные компоненты ПС;

— готовые (покупные) программные продукты и пакеты прикладных программ;

— крупные программные комплексы определенного функционального назначения;

— системы управления базами данных;

— файлы и информационные массивы баз данных;

— электронные документы на программы и данные.

Основные особенности повторного использования программ и данных в системах определяют две группы задач: структурирование программ и данных на стадиях анализа и проектирования систем, предполагающее последовательную декомпозицию заданных функций системы, что позволяет выделять компоненты, которые могут быть применены повторно как готовые, и описание их взаимодействия с другими компонентами; а также сборку или интеграцию компонентов и комплексное, квалификационное тестирование системы в целом. На обеспечение мобильности программ и данных направлена значительная часть ***методов и средств современной программной инженерии***. Четкое разделение результатов работ,

выполняемых на каждой стадии жизненного цикла ПС, и определенные условия переходов между этапами ЖЦ позволяют выделить следующие ***уровни переноса и повторного использования*** ПС:

— модели предметной области и спецификаций требований, возможно, реализуемые разными способами на этапе проектирования ПС;

— проектные спецификации требований на этапе разработки ПС;

— исходные тексты программ на языках программирования, применявшихся при разработке повторно используемых программных компонентов;

— объектные коды программ, когда обеспечена структурная, аппаратная совместимость между исходной и целевой (реализующей) платформами;

— структуры файлов и информация баз данных;

— тесты проверки функционирования компонентов, тесты проверки соответствия повторно используемых программ стандартизированным интерфейсам, комплексных тестов.

Процессы переноса программ и данных на иные платформы, выбор методов обеспечения мобильности ПС и характеристики используемых ресурсов для их реализации, прежде всего, зависят от параметров компонентов, предполагаемых для переноса. ***Ресурсы требуются*** в той или иной степени на ***двух фазах*** процессов переноса программ и данных:

— ***при создании*** потенциально переносимых ПС и БД, когда свойства эффективной мобильности предусматриваются и реализуются при их разработке и определяются возможные платформы и области повторного использования таких программ и данных;

— ***при непосредственной реализации*** с соответствующими затратами процессов переноса ПС и БД, в различной степени подготовленных для переноса на иные платформы или для повторного использования на той же платформе.

При анализе первой фазы следует учитывать, что к настоящему времени накоплен большой объем комплексов и компонентов программ и информации в базах данных, при создании которых не учитывалась возможность их последующего переноса на иные платформы — так называемые ***унаследованные системы***. Более того, из-за ограниченных ресурсов вычислительных средств при реализации крупных функциональных задач

программы и данные в таких системах обычно в максимальной степени адаптировались при разработке к особенностям и параметрам ЭВМ для эффективного использования их ресурсов. Предельным случаем могут служить: бортовые ЭВМ, с соответствующими ПС, для авиационных, ракетных и космических систем, в которых ограничения веса и габаритов аппаратуры вызывают повышенные требования к эффективности использования вычислительных ресурсов.

В зависимости от степени программной совместимости между исходной и новой, целевой платформами можно рассматривать следующие ***варианты применения мобильности***:

— при полной несовместимости платформ может потребоваться разработка всего комплекса программ заново (возможно, с использованием имеющихся спецификаций требований и методов реинжиниринга);

— при несовместимости языков программирования или диалектов одного языка требуется переписывание программ ПС на том языке, который принят для проекта новой системы (возможно, с использованием имеющихся проектных спецификаций и встраиваемых повторно используемых компонентов);

— при несовместимости аппаратно-программных платформ, поддерживающих один и тот же язык программирования, требуется перекомпиляция текстов ПС на новой платформе (возможно, с автоматической оптимизацией, обеспечиваемой применяемой системой программирования);

— при обеспечении двоичной совместимости архитектуры исходной и новой платформ перенос достигается непосредственным исполнением ПС на новой платформе (возможно, использующей средства эмуляции некоторых компонентов архитектуры исходной платформы).

Задачи и объекты, связанные с мобильностью ПС и БД в системах и подлежащие рассмотрению ***при выборе методов и средств обеспечения переносимости***, включают:

— унифицированные протоколы и интерфейсы взаимодействия ПС между собой, с пользователями, с внешней средой, к которым относятся, прежде всего, интерфейсы прикладного программирования, определяемые выбранной архитектурой среды системы, включающей интерфейсы операционных систем, сетевые протоколы, спецификации служб организации процессов, функционирующих поверх операционных систем;

— языки программирования и инструментальные средства, поддерживающие создание переносимых ПС и БД систем и средства программной инженерии — CASE-системы;

— языки баз данных и системы управления базами данных;

— форматы данных, форматы внешних электронных сообщений;

— форматы переносимых электронных документов.

Эффективность выбора и выделения компонентов для повторного использования и переноса на другие аппаратные и операционные платформы зависит, прежде всего, от их размера и от кратности возможного применения. При разработке ПС небольшого масштаба (порядка тысячи строк исходного текста) поиск и подбор готовых компонентов для их применения в новом ПС чаще всего оказываются нерентабельными. Таким образом, существует некоторый диапазон малых размеров программ и информации баз данных, для которых ***нецелесообразно применять*** ранее созданные программы и массивы данных. По этому параметру можно выделить ***методологии переноса***:

— комплексов программных и информационных компонентов, а также операционной среды в целом, решающих все функциональные задачи определенной сложной системы и полностью сохраняющих свою структуру на новой аппаратной платформе;

— достаточно автономных, крупных ПС и массивов информации баз данных, решающих дополнительные, функциональные задачи во взаимодействии с имеющимися на новой аппаратной платформе операционными средствами;

— отдельных модулей или небольших функциональных компонентов программ и информационных массивов данных для расширения и совершенствования функций, ранее реализованных функциональных задач на той же аппаратной и операционной платформах.

Проектирование систем ***с использованием повторно применяемых компонентов*** становится ***особенно рентабельным для крупных ПС***, содержащих сотни или тысячи модулей, и с большими объемами обрабатываемой информации. Кратность применения компонентов также значительно влияет на эффективность их переноса. Особенно тщательную отладку, унификацию интерфейсов и оформление документации целесообразно проводить для тех компонентов, которые в перспективе будут использоваться

многократно различными специалистами, в различных вариантах ПС и на той же или различных платформах.

Наиболее широко применяется перенос программ на ЭВМ с иной архитектурой и операционной средой на уровне исходных текстов программ и данных на алгоритмических языках программирования высокого уровня. На практике приходится встречаться с множеством ситуаций переноса программ и данных между несовпадающими аппаратными и операционными платформами, а также с различающимися степенью мобильности исходных ПС и БД, подлежащих переносу, и применяемыми технологиями их создания. Это разнообразие ситуаций определяет широкий диапазон значений эффективности переноса по потребным ресурсам на его проведение и выбор рациональных методов реализации. Поэтому в каждом конкретном случае целесообразно проводить факторный и технико-экономический ***анализ эффективности переноса и планировать его проведение***.

Процессы переноса программных средств и баз данных регламентируются рядом процедур и документов, стандартизированных в **ISO 14764** (п. 8.5), который детализирует требования к процессам переноса, определенным в базовом стандарте на жизненный цикл ПС (см. **ISO 12207**, п. 5.5.5). Специалисты, которые проводят перенос по рекомендациям этих стандартов, должны разработать план переноса, известить пользователей, обучить персонал, выдать предупреждения о завершении переноса, оценить влияние новой версии и внешней среды и архивировать соответствующие данные. Если систему или программный продукт (включая данные) переносят из старой в новую эксплуатационную среду, следует обеспечить, чтобы программный продукт и данные были ***корректно изменены*** при переносе. Для этого необходимо решить следующие ***основные задачи***: определить все добавляемые или изменяемые программные компоненты, продукты или данные; проверить соответствие реализации конкретных задач спецификациям требований заказчика на перенесенную версию ПС и БД.

Для контроля переноса системы необходимо разработать, документально оформить и выполнить ***план переноса программного продукта***. К планируемым работам могут быть привлечены пользователи. В содержание плана должны быть включены:

— анализ и формирование требований к результатам переноса;

— разработка (или приобретение) инструментальных средств для выполнения переноса;

— настройка программного продукта и данных к новым условиям и среде эксплуатации;

— выполнение процессов переноса;

— верификация и тестирование результатов переноса;

— обеспечение последующей поддержки прежней среды и программного продукта.

Разработка плана переноса должна быть основана на исходных данных и требованиях заказчика или потенциальных пользователей. После завершения сопроводителем планирования переноса заказчику и пользователям должно быть ***направлено уведомление*** о планах и работах по переносу программного продукта и базы данных. В содержание уведомления целесообразно включить: объяснение того, почему прежнюю среду, ПС нельзя больше сопровождать и поддерживать; описание новой среды и программного продукта с указанием даты, с которой они доступны для заказчика и пользователей.

Сопроводитель должен также представить пользователям ***план процедуры и график (Программу) переноса*** и решить следующие задачи:

— определить все компоненты, затрагиваемые переносом;

— отработать обратную связь и информацию с заказчиком и пользователями;

— определить специфику пользователей;

— опубликовать график (Программу) переноса.

Для плавного перехода в новую среду пользователями параллельно могут выполняться работы в прежней и новой среде с соответствующими версиями ПС и БД. Сопроводитель должен выполнить следующие ***работы по обучению*** персонала:

— определить требования по обучению для реализации переноса;

— запланировать реализацию требований по обучению переноса;

— выполнить проверку результатов обучения после выполнения переноса;

— обновить и откорректировать планы обучения.

После завершения запланированного переноса должны быть посланы соответствующие уведомления всем заинтересованным сторонам. Все связанные с прежней средой документы, журналы регистрации и програм-

мы следует поместить в архивы. После завершения переноса целесообразно провести ***итоговый анализ*** для оценки влияния перехода к новой аппаратно-операционной среде на различные аспекты эксплуатации перенесенного программного продукта. Результаты анализа должны быть разосланы соответствующим заинтересованным сторонам для информации, руководства и использования в работе.

Отчетными результатами работ по переносу ПС и БД являются:

— перенесенный программный продукт на новой платформе;
— план реализации переноса;
— инструментальные средства для переноса;
— извещения о намерениях по переносу;
— уведомление о завершении переноса;
— архивные данные процессов и результатов переноса.

Предельным по сложности и трудоемкости случаем является ***перенос унаследованного комплекса программ и информации базы данных*** на несовместимую, совершенно новую аппаратную и операционную платформу. При этом может требоваться сохранение большой накопленной информации базы данных и пользовательского интерфейса. В этом случае могут сохраняться и переноситься алгоритмы или спецификации задач обработки информации, а также должен быть специально организован перенос накопленной информации базы данных. Изменение платформы и расширение ее параметров может привести к целесообразности модернизации алгоритмов. В результате будет создаваться, по существу, новая система с использованием общего системного задела и информации баз данных.

Существует множество пакетов мобильных прикладных программ, созданных с применением современного инструментария. Эти средства автоматизации разработки ПС и БД резко ***упростили процессы переноса*** и свели их в ряде случаев к автоматизированной трансляции и адаптации выбранных пакетов на платформы определенных типов. При этом технология создания ПС и БД путем переноса их на другие аппаратные и операционные платформы претерпела качественное изменение и активно развивается на основе ***концепции и стандартов открытых систем*** (см. лекцию 3).

При анализе баз данных, как объектов переноса, целесообразно рассматривать ***два компонента***: системы управления данными (СУБД) и

совокупность данных, упорядоченных по некоторым правилам. Простейший вариант переноса информации БД на иную аппаратную платформу реализуется, когда на обеих платформах имеются апробированные СУБД одного типа и версий. При этом считается, что отсутствуют дополнительные технические ограничения для размещения всей информации БД и нет необходимости изменять и адаптировать ее структуру на новой аппаратной платформе. В этом случае основные работы сводятся к переносу всего объема информации БД и к испытаниям после этого, функционирования СУБД на новой платформе, на соответствие документации исходной версии СУБД с перенесенной информацией. При этом трудоемкость и длительность создания БД на новой платформе определяются в основном работами по переносу информации БД и испытаниями новой системы.

Однако чаще последующий перенос БД не предусматривается при ее первичном формировании и наполнении и возникает после длительной эксплуатации ***унаследованной системы***. Причиной обычно являются неудовлетворительные показатели качества функционирования БД, потребность в дополнительных ресурсах памяти и производительности ЭВМ, недостаточное время реакции на запросы данных. При этом возможна крайняя ситуация, когда необходимо перенести информацию БД с не полностью известной структурой и связями под управление совершенно другого типа СУБД на иную платформу, с большими ресурсами и возможностями. Сложность, трудоемкость и длительность переноса БД в этом случае значительно возрастают и требуют тщательного планирования и организации работ, приближающихся к созданию совершенно новой БД.

Затраты и сложность переноса информации базы данных зависят прежде всего от ее характеристик, которые отражают ***форматную, лингвистическую и физическую совместимость содержания переносимой БД*** между рассматриваемыми платформами:

— форматная совместимость характеризуется степенью соответствия данных в БД анализируемых платформ требованиям стандартов на форматы представления данных для документальных, фактографических, словарных или иных баз данных;

— лингвистическая совместимость определяется степенью использования в рассматриваемых БД единых лингвистических средств (классификаторов, рубрикаторов, словарей), формализованных соответствующими стандартами;

— физическая совместимость заключается в степени соответствия кодировки информации БД одинаковым стандартам на машиночитаемые носители информации.

Если одновременно с информацией БД переносятся полностью программные средства СУБД, то это обеспечивает сохранение функций управления данными. Однако может потребоваться перенос информации БД на иную аппаратную платформу с другим типом СУБД. Тогда задача переноса усложняется.

Для средних и крупных проектов системный анализ переноса целесообразно завершать ***оценкой суммарной трудоемкости и длительности переноса программного продукта и их сопоставлением с полной разработкой ПС и БД*** при некотором использовании алгоритмического и системного задела. Кроме того, полученный комплекс программ следует оценивать по использованию памяти и производительности новой ЭВМ. Такую оценку необходимо проводить с учетом возможного тиража ПС и перспективы длительной его эксплуатации. Ориентация на снижение затрат при переносе программ зачастую отражается значительными потерями в эффективности эксплуатации системы вследствие увеличения объема программ в объектном коде и снижения их производительности, особенно когда программный продукт длительно используется на многих экземплярах ЭВМ в реальном времени.

15.4. Ресурсы для обеспечения сопровождения и мониторинга программных средств

Прогнозирование необходимых основных ресурсов — труда, времени и числа привлекаемых специалистов для сопровождения и мониторинга сложных комплексов программ осложнено тем, что затраты на изменения состоят из двух принципиально различных частей. Первая, обычно наименьшая, часть изменений характеризуется затратами на обнаружение и устранение дефектов и ошибок в ПС, проявление которых ***непредсказуемо*** и имеет большие флюктуации в зависимости от характеристик проекта, квалификации специалистов, применяемого инструментария и ряда других трудно учитываемых факторов. Априори оценить и прогнозировать такие затраты при сопровождении конкретных ПС вряд ли возможно и

ниже они не рассматриваются. Вторая часть изменений регламентирована целеустремленным совершенствованием и упорядоченными модификациями версий программного продукта, масштаб которых ***может предварительно прогнозироваться*** с некоторой достоверностью. Такие изменения могут служить основой для определения возможных затрат на разработку дополнительных функций и значительных модификаций версий ПС, которые можно обобщать на некотором интервале времени сопровождения или для проекта в целом.

При анализе затрат на сопровождение и мониторинга программных средств целесообразно рассматривать следующие ***сценарии***:

— определение размера отдельных локальных модификаций программ и данных практически без учета взаимодействий с остальной частью версии программного продукта, благодаря его четкой структурированности и возможности выделения размера конкретной изменяемой функции;

— совокупные затраты ресурсов на реализацию каждой модификации, имеющей глубокие взаимосвязи с множеством компонентов всего крупного программного продукта, при которой необходимо учитывать не только размер конкретного изменения, но и величину влияния на весь комплекс программ;

— оценивание интегральных затрат и совокупных размеров изменений при сопровождении и управлении конфигурацией ПС и БД в течение некоторого интервала времени (месяц, год) с учетом всего множества изменений.

Первым этапом прогнозирования необходимых ресурсов при сопровождении является создание ***комплекса требований*** к конкретной модификации функций программного продукта, на которые могут быть разбиты фактические компоненты, определяющие функциональную пригодность ПС. В дальнейшем разбиение может детализироваться, формируя упрощенный или более точный уровень абстракции и взаимодействия изменяемых компонентов. Следует учитывать, что в максимальной степени детализированная структура ПС может принести пользу на стадии предварительного оценивания размера модификации ПС. Один из путей оценки размера изменений ПС, находящегося на этапе концепции проекта, заключается в сравнении его функциональных задач и свойств с уже существующими версиями. При обосновании необходимых ресурсов для сопровождения сложных ПС наибольшее значение имеют три ключевых фактора:

— размер — масштаб подлежащих разработке полностью новых или модификаций программных компонентов;

— размер и относительная доля готовых программных компонентов, которые могут быть заимствованы из предшествовавших проектов и повторно использованы для модификаций в очередной версии программного продукта;

— относительные затраты ресурсов на создание модификаций и новых компонентов ПС с оцененным масштабом изменений: труда специалистов, времени, бюджета на единицу размера (на строку текста программ) или полные затраты на разработку всей новой версии ПС.

Эти факторы могут быть оценены квалифицированными экспертами на основе имеющегося у них опыта мониторинга и реализации предшествовавших подобных модификаций. Достоверность прогнозов требующихся ресурсов зависит, прежде всего, от ***точности оценки исходных требований на совершенствование программного продукта***. Они позволяют использовать опыт прошлых разработок и их отличия от новых методов и функций, предусмотренных в конкретных проектах, а также индивидуальные возможности коллектива разработчиков или другие уникальные особенности конкретного проекта. При наличии перечисленных исходных данных и положительной оценке целесообразности экспертного анализа и прогнозирования ресурсов для сопровождения ПС их следует использовать для:

— оценки размера — масштаба (числа строк) предполагаемого изменения текста разрабатываемых новых программ, с учетом размера готовых повторно используемых компонентов и характеристик возможного языка программирования;

— расчета возможной полной трудоемкости и длительности разработки корректировок версий ПС, а также среднего числа специалистов, необходимых для их реализации;

— обобщения основных технико-экономических показателей и оценки полной стоимости сопровождения ПС, анализа результатов и обоснования, рентабельности продолжения мониторинга, модификаций и сопровождения комплекса программ.

При технико-экономическом обосновании (см. лекцию 5) сопровождения проекта ПС целесообразно применять методы и методики, адекватные целям и этапам его реализации. Приступая к разработке модификаций

комплекса программ, как в любой профессиональной деятельности, необходимо сначала провести реалистическую оценку возможного изменения ***масштаба проекта*** — поставленных целей, ресурсов проекта и выделенного времени. Задача управления масштабом состоит в задании базовых требований, которые включают разбитое на компоненты ограниченное множество дополнительных функций и требований, намеченных для реализации в конкретной версии проекта. ***Базовый уровень изменений масштаба ПС должен обеспечивать***:

— приемлемый для заказчика минимум дополнительных функций и требований к версии программного продукта;

— разумную вероятность успеха с точки зрения возможностей коллектива сопровождения и разработчиков модификаций за требуемое время.

Потребность в ресурсах, объем реализуемых функций и требования спецификаций для модификаций в наибольшей степени зависят от допустимого ***размера — масштаба и сложности модификаций сопровождаемого ПС***. Основная ***цель оценки изменений масштаба ПС*** — подготовить возможность принять обоснованное решение о допустимости дальнейшего мониторинга и сопровождения проекта в область системного анализа, разработки требований и проектирования модификаций и новых версий программного продукта. Если оказывается, что рассчитанные первично масштаб и требуемые ресурсы для изменений ПС не могут быть обеспечены заказчиком для продолжения сопровождения, то возможны кардинальные решения: либо изменение некоторых выделяемых ресурсов, либо прекращение мониторинга и модификации данного программного продукта.

Для ***уменьшения возможных методических ошибок*** оценок ресурсов для сопровождения и модификации ПС следует начинать с прогнозирования размеров изменений или новой версии программного продукта, достоверность и ошибки которых могут быть обусловлены ***следующими факторами***:

— проблема, цель и новые дополнительные функции ПС могут быть недостаточно хорошо поняты разработчиками модификаций и/или заказчиками из-за того, что некоторые существенные факторы были упущены или искажены из-за предвзятого к ним отношения;

— специалисты-оценщики масштаба изменений версии ПС могут допустить значительные ошибки при попытке описания того, насколько боль-

шими могут быть изменения системы или комплекса программ до этапа разработки концепции или предварительного проекта модификации;

— предприятие, сопровождающее ПС, не располагает стандартами и методиками, с помощью которых можно выполнять первичный процесс оценивания масштаба модификации ПС (либо в случае наличия стандартов их никто не придерживается);

— менеджеры и специалисты проекта полагают, что было бы неплохо фиксировать возможные изменения масштаба и спецификаций требований в начале сопровождения, заказчики же часто считают, что не стоит тратить драгоценное время на оценки размеров предстоящих модификаций и на детальную разработку требований к сопровождению.

Затраты на сопровождение ПС можно оценивать потребностью трудовых и временных ресурсов для его обеспечения и для реализации. Эти затраты состоят из двух связанных частей: ***затрат на реализацию*** соответствующих характеристик качества, обеспечивающих эффективное сопровождение программных продуктов, и ***затрат при использовании*** этих характеристик в процессе эксплуатации комплекса программ. Обычно совершенствование качества и повышение затрат на реализацию характеристик способствует снижению затрат при их эксплуатации. Последние трудно оценить априори, так как они зависят от внешней среды и активности применения конкретного ПС, а не от его свойств и требуемого качества. (По некоторым оценкам, количество специалистов, участвующих в сопровождении на расширение функциональности и улучшение качества ПС (≈ 40%) и на устранении дефектов (≈15%), что превышает количество специалистов, занятых созданием новых программ (≈ 45%)).

Стоимость и длительность сопровождения компонентов или комплекса программ может определяться контрактом между заказчиком и поставщиком по прецедентам аналогичных проектов с учетом изменяющейся конъюнктуры. Затраты на обеспечение и реализацию сопровождения программ определяются длительностью цикла жизни комплекса программ; его мобильностью, уровнем автоматизации технологии разработки и тиражом программ. Для их оценивания, прежде всего, необходимо выделять основные виды затрат при сопровождении конкретного комплекса программ и наиболее существенные факторы, которые на них влияют. Такой анализ может дать ***ориентиры для прогнозирования общих затрат на сопровождение*** и для оценивания этой характеристики ***в конкретных***

проектах ПС. При этом важно учитывать соотношение затрат на разработку и на сопровождение в течение цикла жизни ***всего тиража*** ПС.

Уникальное, заказное ПС, основная часть жизненного цикла которого приходится на разработку, может создаваться почти без учета последующих затрат на сопровождение. Однако многократно модернизируемые и широко тиражируемые ПС требуют больших затрат на сопровождение. Вследствие длительного срока сопровождения и эксплуатации (в ряде случаев более 10 лет), а также большого числа версий, содержащих результаты модернизаций, совокупные затраты на сопровождение в некоторых случаях значительно превышают затраты на первичную разработку программ. Эти затраты распределяются по всему интервалу времени сопровождения, вследствие чего при подготовке каждой версии затраты обычно меньше, чем на первичную разработку ПС. Длительное сопровождение иногда вызывает неоднократную смену специалистов, осуществляющих сопровождение. При таких заменах появляются значительные ***затраты на обучение*** новой группы сопровождения, что вызывает рост общих затрат. Сокращение затрат на сопровождение возможно за счет некоторого увеличения затрат при разработке ПС, так что при рациональном проектировании в сумме затраты могут быть уменьшены иногда весьма заметно.

При системном анализе затраты на сопровождение можно считать аддитивными и включающими ***составляющие***:

— затраты на обнаружение и устранение ошибок и дефектов в каждой версии ПС;

— затраты на доработку и совершенствование программ, формирование и испытание новых модернизированных версий ПС;

— затраты на тиражирование каждой новой версии и ее внедрение в эксплуатируемых и новых системах.

Доля каждой составляющей в общих затратах на сопровождение может значительно изменяться в зависимости от особенностей сферы применения и жизненного цикла конкретного ПС. Для долгоживущих (≈10 лет), тиражируемых (1000—100 000 экземпляров) ПС доминирующими обычно являются затраты на модернизацию и доработку версий программ. Затраты на модернизацию зависят от тиража косвенно, вследствие расширения условий применения конкретного ПС и увеличения потока запросов пользователей на развитие программ. Так же косвенно влияет тираж на запросы для устранения выявленных ошибок.

Затраты на обнаружение и устранение дефектов и ошибок в программе определяются двумя факторами: затратами на обнаружение каждой ошибки и затратами на устранение выявленных ошибок при формировании очередной версии. Чем меньше ошибок в программе, тем труднее они обнаруживаются, т.е. тем выше затраты на выявление каждой ошибки. Затраты на устранение ошибок и корректировку программ пропорциональны числу дефектов, выявленных между очередными версиями. При сопровождении непрерывно требуются затраты для контроля состояния версий программ и обеспечения их сохранности. По опыту работ, широко тиражируемый комплекс программ объемом $\approx 10^5$ строк, может требовать непрерывных усилий коллектива в составе десятка и более специалистов для устранения ошибок, корректировок версий и документации.

Затраты на совершенствование и модернизацию программ близки по содержанию (но не по величине) к затратам на их первичную разработку. Модернизация обычно производится поэтапно. Для каждой новой эталонной версии изменяется (разрабатывается) только некоторая часть от всего объема ПС. Эта часть при вводе очередной версии может составлять 10—20% от объема всего комплекса. Сложность связей в ПС приводит к тому, что удельные затраты на изменяемые программы при модернизации каждой версии могут быть в 2—3 раза больше, чем затраты на создание программ такого же объема при первичном проектировании. Эта величина зависит от того, насколько путем стандартизации архитектуры и интерфейсов при системном проектировании предусматривались перспективы совершенствования ПС. Для выполнения этих работ иногда используется коллектив специалистов, осуществивших первичную разработку. Такая организация наиболее характерна для уникальных, заказных ПС. В этих случаях первичную разработку и модернизацию трудно разделить. Для широко тиражируемых ПС на сопровождение часто выделяется специальный коллектив, не проводивший первичную разработку. В этих случаях этапы разработки и сопровождения, а также сопутствующие им затраты можно разделить более четко.

Затраты на тиражирование каждой новой версии включают совокупные затраты на производство экземпляров программного продукта, их инсталляцию в объектных ЭВМ и освоение для нормальной эксплуатации. Затраты на тиражирование версий при сопровождении практически пропорциональны произведению числа версий и их тиража. Вследствие этого

даже относительно малые затраты на каждый экземпляр при внедрении новой версии могут приобретать большое значение в жизненном цикле всей серии версий программного продукта.

Ресурсы инструментальной среды при сопровождении ПС определяют ряд специальных работ, для выполнения которых необходимы отдельные системы и средства. Необходимо наличие отдельных сред разработки модификаций и сред тестирования корректировок. Сопроводитель должен помогать заказчику при создании плана и инструментальной среды сопровождения. Данное требование является критичным при формировании среды сопровождения и должно быть учтено при предварительном планировании выделяемых финансовых средств для сопровождения и мониторинга программного продукта.

Потенциальными средствами, определяющими стоимость сопровождения программных средств, являются инструментальные CASE-средства. Они должны представлять взаимосвязанный набор инструментальных средств, обеспечивающих все аспекты разработки модификаций и сопровождения программных средств (см. стандарт **ISO 14471**). Взаимосвязанный набор CASE-средств должен быть скомпонован в виде инструментальной среды организации и разработки модификаций, представляющей собой методы, политики, руководства и стандарты, обеспечивающие проведение работ по сопровождению, которые обеспечивают инструментарий для разработки и модификации программных продуктов. Сопроводителю также должна быть предоставлена инструментальная среда тестирования модифицированного программного продукта, вне среды его эксплуатации.

На основе анализа и оценивания рассчитанных характеристик ресурсов для сопровождения следует выполнять заключительное ***технико-экономическое обоснование необходимости сопровождения конкретного программного продукта*** и определять:

— целесообразно ли продолжать работы по сопровождению и мониторингу конкретного программного продукта или следует его прекратить вследствие недостаточных ресурсов специалистов, времени или большой трудоемкости разработки модификаций;

— при наличии достаточных ресурсов следует ли провести маркетинговые исследования для определения рентабельности создания очередной версии программного продукта и поставки ее на рынок;

— достаточно ли полно и корректно формализованы концепция и требования к модификациям версий программного продукта, на основе которых проводились экспертные оценки и расчеты затрат, или их следует откорректировать и выполнить повторный анализ с уточненными исходными данными;

— есть ли возможность применить готовые повторно используемые компоненты ПС, в каком объеме относительно размера комплекса программ и рентабельно ли их применять в конкретной версии программного продукта или весь проект целесообразно разрабатывать как полностью новый.

ЛЕКЦИЯ 16
УПРАВЛЕНИЕ КОНФИГУРАЦИЕЙ В ЖИЗНЕННОМ ЦИКЛЕ ПРОГРАММНЫХ СРЕДСТВ

16.1. Процессы управления конфигурацией программных средств

Цель управления конфигурацией при разработке и сопровождении сложных программных средств и систем, состоящих из многих компонентов (единиц конфигурации), каждый из которых может иметь разновидности или версии, обеспечить управляемое и контролируемое развитие их структуры, состава компонентов и функций, а также сокращение дефектов в течение всего жизненного цикла ПС. В процессе организации конфигурационного управления необходимо построить и использовать компактные и наглядные ***схемы однозначной иерархической идентификации и изменения взаимодействия компонентов ПС***:

— объектов — модулей и компонентов ПС разного уровня интеграции, подвергающихся корректировкам (систему идентификации и адресации изменений в комплексе программ и в документах);

— корректировок содержания и взаимодействия проводимых изменений, которая должна обеспечивать возможность однозначного контроля, истории развития модификаций компонентов любого уровня, во времени и в пространстве элементов версий комплекса программ (типы, содержание и взаимосвязь корректировок);

— специалистов, участвующих в конфигурационном управлении и сокращении дефектов, их права на доступ к определенным компонентам ПС и документам на конкретных стадиях сопровождения, реализации и утверждения изменений.

Процесс управления конфигурацией (стандарт **ISO 12207**, п. 6.2), является процессом применения административных и технических процедур на всем протяжении ЖЦ программных средств для: обозначения, определения и установления состояния базовой версии программных продуктов в системе; управления изменениями и выпуском объектов; описания и сообщения о состояниях объектов и заявок на внесение изменений в них; обеспечения полноты, совместимости и правильности объектов; управления хранением, обращением и поставкой объектов. Этот процесс включает: подготовку процесса; определение конфигурации; контроль конфигурации; учет состояний конфигурации; оценку конфигурации; управление выпуском и поставку программного продукта. Все основные и вспомогательные процессы подлежат адаптации и конкретизации применительно к характеристикам определенного проекта

Стандарт **ISO 15846** обобщает, детализирует и развивает основные концептуальные положения, представленные в стандарте **ISO 12207**. Шесть разделов (6-й — 11-й) начинаются с цитирования соответствующих шести базовых требований раздела 6.2 стандарта **ISO 12207.** В каждом из них излагаются подробные рекомендации по реализации его базовых требований по управлению конфигурацией ПС. Существенным достоинством стандарта **ISO 15846** является подробное и систематичное изложение практических рекомендаций по управлению конфигурацией сложных комплексов программ, которые целесообразно использовать в крупных современных реальных проектах систем.

В процессе проектирования ПС должна быть формализована и документально зафиксирована коррелированная с Концепцией сопровождения (см. лекцию 15) ***Концепция организации конфигурационного управления проектами программных средств,*** содержащая в основе:

— ожидаемую длительность поддержки развития и модификации конкретного проекта ПС;

— масштаб и уровень предполагаемых изменений и модификаций;

— возможное число и периодичность выпуска базовых версий программного продукта;

— организационные основы процессов сопровождения и конфигурационного управления программным средством;

— требования к документированию изменений и базовых версий ПС;

— кто будет осуществлять управление конфигурацией — покупатель, разработчик или специальный персонал поддержки ЖЦ ПС.

Если предполагается, что программный продукт будет иметь длительный жизненный цикл или ожидаются значительные изменения, то следует рассмотреть и учесть наиболее детальные требования к методике организации и к коллективу, ответственному за конфигурационное управление и его документирование. В стратегии управления следует учесть характеристики системы: количество компонентов программного средства, типы, размер и критичность создаваемых и применяемых программных компонентов и продуктов. Управление конфигурацией (УК) следует организовать так, чтобы персонал знал свои обязанности и имел достаточно независимости и полномочий для выполнения поставленных задач.

Управление конфигурацией включает действия и средства, позволяющие устанавливать категории, статус и личности руководителей, которые правомочны определять целесообразность и эффективность изменений, а также техническую реализуемость корректируемых версий с учетом ограничений бюджетов и сроков (рис. 16.1). При анализе и селекции изменений важен точный учет степени влияния каждого изменения на все остальные компоненты и на их основные характеристики качества. Поэтому решения о кардинальных изменениях ПС и компонентов должны приниматься на достаточно высоком уровне руководства проектом, способного оценить их влияние на концептуальную целостность и качество всей информационной системы (табл. 16.1).

Концепция конфигурационного управления конкретным проектом должна предусматривать возможность ***анализа изменений иерархической структуры — конфигурации*** программных средств и их компонентов, как сверху вниз, так и снизу вверх. Первая задача состоит в обеспечении пошаговой детализации сверху вниз возможных причин конкретных дефектов (проявлений ***вторичных ошибок***) или неэффективности функционирования программы для обнаружения их первичного источника (***первичной ошибки***) (см. лекцию 10). Вторая задача при движении снизу вверх должна обеспечивать проверку корректности взаимодействия связанных корректировок и сохранения концептуальной целостности и качества комплекса программ и/или его компонентов.

Средства и методы УК должны быть ориентированы на ***координированное развитие множества версий ПС и их компонентов***, каждая из

которых имеет достаточно высокое качество и специфические функции, а также различных, может быть, удаленных разработчиков и пользователей. Корректировки могут проводиться несинхронно по множеству версий пользователей, в результате чего образуется набор версий выполненных изменений, учитывающих текущие состояния версий ПС у конкретных пользователей.

Рис. 16.1

Таблица 16.1

Объекты изменения	Типы изменений объектов	Право на выполнение изменения имеют:	Право на утверждение изменения имеют:
Версии модулей и компонентов программ, данных и документов	Устранение дефектов в программных модулях и компонентах	Программисты — разработчики модулей и компонентов	Руководители разработки функциональных групп программ и данных
Версии функциональных групп программ и документов	Корректировка функций и взаимодействия программных компонентов	Руководители разработки функциональных компонентов	Менеджер-архитектор программного средства
Базовые версии программного продукта и комплекса документации	Модификация и улучшение функций и качества базовых версий программного продукта	Менеджер-архитектор программного продукта	Менеджер и заказчик проекта программного продукта
Версии программного продукта пользователей	Адаптация к характеристикам внешней среды пользователей	Заказчик или сопровождающий версию программного продукта пользователя	Менеджер, сопровождающий версию программного продукта пользователя

Важной целью управления конфигурацией является документальное оформление и обеспечение полной наглядности текущей конфигурации программ и данных и степени выполнения требований к их функциональным характеристикам. Другая задача заключается в том, чтобы все лица, работающие над проектом, в любой момент его жизненного цикла использовали достоверную и точную информацию о всех единицах конфигурации (ЕК) и их взаимодействии. Процессы УК включают работы по идентификации конфигурации, контролю изменений, определению базовой версии разработки и архивированию программного средства, включая соответствующие документы жизненного цикла, по аудиту конфигурации, компоновке и поставке программного продукта в течение всего жизненного цикла системы. Процессы УК, выполняемые совместно с другими процессами жизненного цикла ПС, направлены на достижение ***следующих основных целей***:

— обеспечить возможность оценки соответствия требованиям заказчика результатов жизненного цикла программного средства;

— обеспечить определяемую и управляемую конфигурацию ПС на протяжении всего жизненного цикла;

— обеспечить управление входными и выходными данными процесса в течение всего жизненного цикла, что гарантирует непротиворечивость и повторяемость работ в процессах;

— обеспечить контрольные точки для проверки, оценки состояния и контроля изменений посредством управления элементами конфигурации и определения базовой версии программного продукта;

— обеспечить контроль над тем, чтобы фиксировались дефекты и ошибки, а изменения регистрировались, утверждались и реализовывались;

— гарантировать надежное физическое архивирование, восстановление и сопровождение единиц конфигурации и документов программного продукта.

Изменения конфигурации ПС и его компонентов должны планироваться и предусматривать в ***плане управления проектом действия*** с четкими разделами:

— ***почему*** и с какой целью производится корректировка программ или данных;

— ***кто*** выполняет и кто санкционирует проведение изменений комплекса программ или компонентов;

— ***какие*** действия и процедуры должны быть выполнены для реализации изменений единиц конфигурации;

— ***когда*** по срокам и в координации с какими другими процедурами следует реализовать определенную модификацию компонентов и конфигурацию ПС;

— ***как*** и с использованием каких средств и ресурсов должны быть выполнены запланированные изменения ПС и компонентов.

Четкая ***организация и автоматизация этого процесса*** позволяют избегать множества вторичных ошибок, обусловленных недостаточной координацией проводимых корректировок и формирования новых версий сложных ПС коллективом специалистов. Этому должна способствовать утвержденная иерархия принятия решений на изменения компонентов и ПС в целом должностными лицами проекта (см. табл. 16.1), поддержанная методами и средствами защиты от несанкционированного доступа при выполнении корректировок специалистами различной квалификации и права доступа на разных уровнях проекта.

Управление конфигурацией должно быть организовано таким образом, чтобы можно было гарантировать беспристрастность и независимость

персонала для достижения необходимых целей управления конфигурацией. Для того чтобы управление конфигурацией было эффективным, следует определить его ***организационную структуру коллектива специалистов***. Эта структура обычно ориентируется на конкретный проект и при необходимости адаптируется, чтобы отвечать потребностям различных этапов жизненного цикла программной продукции. Она должна определять связи между различными видами деятельности, непосредственно входящими в процесс управления конфигурацией. Структура должна включать функцию управления конфигурацией; взаимодействующие организации; службы проектирования, закупок и контрактов; управление данными, изготовление, обеспечение качества и другие дисциплины, которые могут быть привлечены, охватывая, если необходимо, субподрядчиков и поставщиков.

Организационная структура специалистов УК должна обеспечивать координацию действий, а также распределение соответствующих полномочий и ответственности за все действия по управлению конфигурацией. В рамках организации проекта ПС следует идентифицировать инстанцию, уполномоченную утверждать конфигурационные базы и любые изменения к ним, обычно это ***совет по конфигурации***. В случае небольших проектов ответственность за управление конфигурацией, руководство проектом может быть возложено на отдельные лица, участвующие в проекте.

Руководитель проекта ***может учредить совет по конфигурации***, который будет иметь полномочия анализировать и утверждать или не утверждать программу и процедуры управления конфигурацией, выбор объектов конфигурации, конфигурационные базы и изменения к этим базам, включая отклонения и разрешения на отклонение. Членов совета по конфигурации обычно назначает руководитель проекта ПС. Совет по конфигурации может быть организован на нескольких уровнях полномочий, если согласно контрактным требованиям необходимо участие заказчика в процессе жизненного цикла ПС, то заказчик тоже может учредить совет по контролю конфигурации.

Конфигурационная идентификация включает методы и средства, с помощью которых можно однозначно устанавливать и различать версии ПС, входящие в них компоненты (единицы конфигурации — ЕК), их варианты и модификации, а также родословное дерево объектов конфигурации, технических условий и идентифицированный комплект документа-

ции. Кроме того, каждый вариант модуля, компонента или ПС и их изменений должен соответствовать правилам нумерации (идентификации).

Цель идентификации конфигурации заключается в однозначной маркировке каждой единицы конфигурации (и ее последующих версий) для того, чтобы:

— установить базис для управления и ссылок на единицы конфигурации;

— выполнить идентификацию для каждой единицы конфигурации, для каждого отдельно управляемого компонента конфигурации и для комбинаций единиц конфигурации, которые составляют ПС;

— провести идентификацию единиц конфигурации до начала реализации контроля изменений и прослеживаемости документов;

— обеспечить идентификацию конфигурации для документов жизненного цикла ПС;

— выполнить идентификацию единицы конфигурации прежде, чем она будет использоваться другими процессами жизненного цикла ПС, для производства и для загрузки;

— исполняемый объектный код содержал идентификацию конфигурации, которая должна быть доступна для других компонентов системы.

Разработчик ПС должен участвовать в выборе ЕК, выполняемому согласно проекту архитектуры системы, должен идентифицировать объекты, которые будут помещены под управление конфигурацией, и назначить уникальный для проекта идентификатор каждой ЕК и каждому дополнительному объекту, находящемуся под управлением конфигурацией. Эти объекты включают программные средства, которые должны разрабатываться или использоваться согласно контракту, и элементы среды разработки ПС. Схема идентификации должна быть составлена на том уровне, на котором объекты будут фактически контролироваться: компьютерные файлы, электронные носители данных, документы, модули ПС, единицы конфигурации. Схема идентификации должна включать статус официальной версии для каждого объекта.

В конкретном проекте следует разработать ***правила нумерации ЕК*** и применять их для идентификации объектов конфигурации, документов по конфигурации и изменений. При этом необходимо учитывать существующие процедуры нумерации, принятые у подрядчика, или общепринятые процедуры. Однако идентификационные номера должны быть уникальны-

ми. В результате с использованием ограниченной и упорядоченной системы символов создается база для однозначного выбора и манипулирования вариантами компонентов или версиями комплексов программ и для процессов прослеживания изменений. Все необходимые функциональные и физические характеристики объектов конфигурации, включая изменения, отклонения и разрешения на них, должны содержаться в четко идентифицированных документах.

Конфигурационный учет составляют методы и средства регистрации и отслеживания состояния объектов — единиц конфигурации, накопления и классификации отчетов о всех реализованных и отвергнутых изменениях вариантов компонентов и ПС в целом. Совокупность отчетов должна обеспечивать идентификацию именования и однозначное отражение текущего состояния ПС и его компонентов, а также накопление историй последовательных модификаций ЕК, приведших к данному состоянию их структуры, функций и характеристик. Правила нумерации или другие системы учета конфигурации должны позволять управлять:

— иерархическими отношениями или отношениями подчинения между объектами конфигурации в рамках структуры ПС;

— иерархическими отношениями или отношениями подчинения между компонентами в каждом объекте конфигурации;

— отношениями между объектами и документами;

— отношениями между документами и изменениями.

В ключевых точках разработки проекта все объекты должны быть подвергнуты ***версионному учету***. В определенный момент нужно фиксировать версии всех объектов и компонентов, составляющих программный продукт или систему. Учет версий ЕК должен проходить, по крайней мере, на каждом из основных этапов проекта. В соответствии с методологией УК в итеративном процессе разработки ПС ***учет версий*** необходим в конце каждой итерации для обеспечения:

— воспроизводимости — возможности вернуться назад во времени и повторить определенный выпуск ранее существовавшего компонента, программной системы или среды разработки;

— контролируемости — объединения требований к системе, проектных планов, результатов тестирования и объектов разработки, для учета версий не только системных компонентов, но и объектов проектирования и планирования;

— отчетов, которые позволяют получать сведения о любой версии системы, сравнивать различные версии, находить ошибки и составлять документацию.

Управление запросами на изменения включает регистрацию, отслеживание и анализ запросов на модификацию ПС, компонентов и данных от субъектов, взаимодействующих с системой (см. рис. 16.1). Оно включает процессы принятия решений для планирования необходимых изменений и процессы их реализации. Управление запросами на изменения представляет собой ***центральную часть Концепции управления конфигурацией***. Незарегистрированные запросы на изменения могут быть потеряны или останутся без реакции.

Запрос на доработку определяет новое свойство ПС или изменение реализации его функций (см. лекцию 10). ***Дефектом*** может быть любая обнаруживаемая проблема, которую следует взять под контроль и разрешить. Хотя с запросами на доработку и дефектами связана достаточно похожая информация, в процессах УК они часто обрабатываются совершенно по-разному. Реализация изменений требует принятия ряда решений. Обычно в определении этого процесса участвуют различные подразделения (руководители проектов, разработчики, специалисты по тестированию). Внешний запрос способен, в свою очередь, породить несколько внутренних технических запросов на доработку.

Следующий шаг после установления типов контролируемых запросов состоит в уточнении объема информации, фиксируемой в ходе жизненного цикла запроса. Наиболее важным и часто наиболее сложным шагом является определение процесса, который используется для контроля каждого запроса на изменение. Типы запросов на изменения, модели переходов и состояний варьируются достаточно широко и обычно включают:

— представление и регистрацию запроса на изменение;

— оценку запроса его категории и приоритета;

— решение о порядке выполнения запроса;

— реализацию корректировок — компоненты системы и программная документация создаются или модифицируются с целью реализации запроса;

— проверку на соответствие требованиям или на отсутствие исправленного дефекта.

Представление запроса на изменение предполагает его регистрацию. Запросы на доработку могут поступать из различных источников. Основные данные, регистрируемые при возникновении таких запросов, — это важность требуемых изменений для пользователей, подробности запроса и личность инициатора. Последнее требуется, чтобы при необходимости можно было выяснить детали и неясные моменты. Иногда запрос на доработку появляется в самой организации в процессе тестирования или внутреннего использования проекта.

Оценка запроса заключается в просмотре вновь поступивших запросов и выводах об их характеристиках. Является ли запрос дефектом или он относится к запросам на доработку; какова значимость этого запроса; каков приоритет в реализации запроса. Необходимо также учесть возможные последствия доработки — изменение доли рынка, увеличение прибыли, влияние на продажи и поддержку пользователей.

Изменения программ и/или данных модулей и небольших программных компонентов подвергается наименее формализованному и защищенному от случайностей конфигурационному управлению. Их изменения в процессе тестирования обычно не требуют санкции руководителей проекта. Версии функциональных групп программ и комплексов программ в целом могут корректироваться только с разрешения руководителей соответствующего ранга. Тем самым должны предотвращаться несогласованные и несанкционированные изменения, способные снизить качество и целостность версий программного продукта. Изменения этих версий допускаются пользователями или поставщиками в пределах, ограниченных эксплуатационными документами по адаптации к характеристикам и особенностям внешней среды и условиям применения конкретной версии ПС.

Разработчик изменений должен составлять сообщения о дефектах и изменениях, чтобы описать каждый дефект, обнаруженный в ПС, находящихся под контролем конфигурации на уровне проекта, и каждую проблему выполнения работ, необходимых в соответствии с контрактом или описанных в Плане разработки ПС. Правила модификации в проекте должны определять, к каким компонентам разработчики различных уровней будут иметь доступ, указывать, доступен определенный компонент в данный момент только для чтения или для чтения и модификации. Если компонент является модифицируемым, специалисты сопровождения вправе получать и изменять составляющие его файлы. Если же компонент ис-

пользуется только для чтения, участники проекта не смогут изменять файлы, а будут лишь ссылаться на них. Компоненты с доступом только для чтения обычно применяются во время тестирования, в процессе сборки групп компонентов или при совместном их использовании в ряде проектов.

Систему корректирующих действий следует реализовать для обработки каждого дефекта, обнаруженного в ПС или модификации, находящихся под контролем конфигурации на уровне проекта (см. рис. 16.1):

— входная информация системы должна состоять из сообщений о дефектах и модификациях;

— необходимо гарантировать, что все обнаруженные дефекты немедленно регистрируются и вводятся в систему УК, необходимые действия инициируются, принятые решения осуществляются, состояние корректирующих действий прослеживается и сообщения о дефектах и модификациях сопровождаются в течение всего срока действия контракта;

— каждый дефект и модификация должны быть классифицированы по категориям и приоритетам;

— должен выполняться анализ для выявления возможных тенденций в зарегистрированных дефектах;

— корректирующие действия должны быть оценены, чтобы определить, были ли дефекты устранены, неблагоприятные тенденции преодолены, а изменения были правильно выполнены без внесения дополнительных дефектов.

Решение о корректировке конфигурации ПС состоит в выборе: выполнить запрос на изменения, отложить реализацию или отклонить запрос. Для запроса на доработку решение обычно выносит руководитель проекта или аналитик. Затем относительно каждого из них принимается решение о реализации в данной версии продукта, отсрочке или отклонении. Процесс принятия решения для дефектов отличается и зависит от двух факторов: текущей фазы цикла разработки и необходимых для реализации усилий. Обычно дефект закрепляется за определенным специалистом; если ошибку удается легко воспроизвести, в новой версии продукта она будет исправлена. Ненужные модификации могут нарушать работоспособность и снижать качество системы, способствуя отставанию от графика и увеличению издержек. На заключительных стадиях проекта целесообразно вводить формальный процесс рассмотрения и внесения исправлений, он должен быть направлен на ограничение изменений и допуск

только самых критических исправлений на стадиях стабилизации кода и квалификационного тестирования.

Реализация корректировок при запросе на доработку может требовать проведения дополнительных проектных работ, поскольку в систему добавляются новые функции. Для исправления дефекта более важно воссоздать среду, где он проявляется и где затем будут тестироваться сделанные модификации. Некоторые представляемые дефекты и запросы на доработку могут относиться не к самому ПС, а к документации. В случае запроса на доработку это означает внесение в документацию описания новых функций; в случае дефектов — исправление документации, если устраненная ранее ошибка изменила поведение пользователя при взаимодействии с системой.

Редакциям ЕК присваиваются определенные ***статусы, определяющие их качество и пригодность для различных операций***. В каждом проекте один из статусов является рекомендуемым, базовым. Последняя редакция ЕК в интеграционном потоке проекта, получившая такой статус, по умолчанию предлагается разработчикам при выполнении операций — ***обновить***. Интегратор может стразу присвоить рекомендуемый статус всем последним редакциям ЕК в интеграционном потоке проекта. Интегратор также вправе присвоить нужный статус нескольким отобранным редакциям. Статус редакций, в которых обнаружены проблемы, допустимо понизить. Если редакция ЕК вызывает ошибки при сборке, ей необходимо присвоить ***статус — отклонена***, что позволит избежать использования данной редакции в дальнейшей работе.

Контроль корректности конфигураций версий компонентов, ПС и данных предназначен для систематической оценки предполагаемых изменений ЕК и координированной их реализации с учетом соответствия спецификациям и требованиям заказчика, эффективности каждого из них и затрат на выполнение изменения. Эти методы должны обеспечивать контроль состояния и развития компонентов ЕК, их вариантов, связей и модификаций, а также адекватность реально изменяющихся объектов и их комплектной документации. Он включает следующие виды деятельности, которые должны быть подробно описаны в документированных процедурах контроля за изменениями ЕК:

— положения, касающиеся организации, состава и срока полномочий совета по конфигурации и его связей с аналогичными советами;

— рассмотрение реализации изменений, начиная с заявки и кончая утверждением модификации после внесения в объект конфигурации, обоснование и оценивание последствий внесения и корректности изменения;

— утверждение или неутверждение изменения;

— обработка и анализ отклонений от требований и подготовка разрешений на отклонения при реализации модификаций;

— сбор, регистрация, обработка и сохранение данных, необходимых для составления отчетов о мониторинге, состоянии и статусе конфигурации ПС.

Разработчик должен установить и выполнять процедуры контроля конфигурации в соответствии с выбранным уровнем контроля для каждого идентифицированного объекта, полномочия людей для санкционирования и выполнения изменений на каждом уровне, последовательность действий, которые необходимо выполнить для того, чтобы запросить разрешение и обработать запрос на изменение, проследить изменение и сопровождать предыдущие версии. Изменения, которые воздействуют на объект, уже находящийся под контролем заказчика, должны быть предоставлены заказчику в соответствии с установленными контрактом формами и процедурами.

Цель контроля изменений — обеспечить регистрацию, оценку, рассмотрение и утверждение корректировок на протяжении всего жизненного цикла ПС:

— контроль изменений должен обеспечить целостность единиц конфигурации и базовых версий программного продукта и защиту их от некорректных модификаций;

— контроль изменений должен гарантировать, что каждое изменение единицы конфигурации учтено в изменении идентификации конфигурации ПС;

— изменения в базовых версиях и единицах конфигурации, находящихся под контролем, должны регистрироваться, утверждаться и прослеживаться, ранняя реализация контроля изменений помогает управлению и организации работ в процессах жизненного цикла ПС;

— изменения ПС должны быть прослежены вплоть до места их источника, а выполнение процессов жизненного цикла следует повторить с того момента, с которого изменения сказываются на выходных данных;

— при внесении изменений должны модифицироваться документы жизненного цикла ПС, на которые эти изменения влияют, а обновление документов должно сопровождаться действиями по контролю изменений.

Работы ***по просмотру и прослеживанию корректности изменений*** должны сопровождать реализацию и контроль изменений ЕК. Целью является оценка дефектов и модификаций, их утверждения, реализации утвержденных изменений и обратной связи к процессам, на которые изменение воздействует, путем использования методов контроля изменений. Последние определяются в процессах планирования проекта ПС и системы и должны включать:

— подтверждение того, что затронутые изменениями единицы конфигурации идентифицированы;

— оценку воздействия изменений на требования безопасности и обеспечение обратной связи к процессу оценки безопасности системы;

— анализ дефектов или изменений и решений о действиях, которые следует предпринять для их коррекции;

— обеспечение обратной связи от сообщений о дефектах, контроля изменений и корректирующих действий к задействованным модификациями процессам.

В интеграционный поток следует отправлять только те модификации, которые предшествуют операции регистрации ЕК. Иначе может получиться, что отправленный набор изменений не был протестирован. Такая ситуация может быть обусловлена отправкой частичных изменений ЕК. Это способно ослабить контроль целостности, действующий в УК. Если в проекте участвуют недостаточно опытные специалисты, и один из них не выполнил операцию регистрации ЕК после внесения окончательных изменений в файл и проигнорировал предупреждения операции отправки компонента на сборку, то в интеграционный поток попадет некорректно функционирующий код ЕК.

Перед сборкой базовой версии ПС рекомендуется ***блокировка интеграционного потока и запрет отправки изменений*** ЕК. В результате интегратор может проводить сборку и создавать базовую редакцию ПС для стабильного множества элементов исходного кода ЕК. Часто во время сборки возникают проблемы, которые можно быстро исправить, внеся изменения в нужный файл ЕК. Такие исправления столь незначительны, что желательно перенести их в очередную редакцию ПС. Если часто про-

водить сборку базовой версии ПС и допускаются на этом этапе мелкие коррективы, то лучше сначала собрать ПС, а потом уточнять и создавать базовые редакции компонентов ЕК.

Сборка версии программного средства — первый уровень тестирования, проводимый интегратором, чтобы убедиться в том, что все модифицированные файлы ЕК могут быть собраны в единый компонент или комплекс программ (см. рис. 16.1). Если сборка прошла успешно, то следует перейти к квалификационному тестированию или повысить статусы редакций компонентов ЕК. В некоторых случаях интегратор может самостоятельно внести исправления и запустить сборку повторно. Сложные проблемы могут возникать, когда модификации ЕК отправлены несколькими сотрудниками. Понять, каким образом их исправить, можно, только обладая достаточными знаниями алгоритмов и внутренней структуры комплекса программ. Чтобы разобраться в ошибочной ситуации, интегратор вправе привлечь разработчиков изменений, они внесут дополнительные изменения в ЕК и снова отправят их в поток интеграции. Чтобы дать им возможность отправить изменения повторно, интегратору необходимо временно исключить их из блокировки интеграционного потока.

Сформированные базовые версии единиц конфигурации должны регистрироваться в контролируемых библиотеках ПС и позволять ссылаться, управлять и прослеживать их изменения. Они должны быть защищены от внесения любых несанкционированных изменений. Конфигурационная база состоит из всех утвержденных документов, которые определяют программную продукцию или компоненты в данный момент. Ее следует устанавливать всегда, когда это необходимо для определения эталонной конфигурации ПС и/или компонентов в течение их жизненного цикла, которая служит отправной точкой для последующей деятельности. Уровень детализации, в соответствии с которым комплекс программ определен в конфигурационной базе, зависит от степени необходимого контроля. Функциональные конфигурационные базы могут состоять из одного документа или из полного комплекта документов, включая документы на инструментальную оснастку и технологические процессы. В программное средство, модифицируемое пользователем, могут вноситься только изменения, которые не будут влиять на идентификацию конфигурации базовой версии комплекса программ.

После проведения работ по реализации и контролю совокупности изменений должна быть разработана и зафиксирована очередная базовая версия ЕК, производная от ранее установленной базовой версии. ***Цель установления базовой версии*** — определить основу для последующих работ процессов жизненного цикла ПС и позволить осуществлять ссылки, управлять и прослеживать единицы конфигурации, для этого требуется:

— установить базовые версии для единиц конфигурации, на которые распространяется сертификационное доверие;

— установить базовую версию для программного средства и определить ее в Указателе конфигурации ПС;

— базовые версии должны храниться в контролируемых библиотеках ПС (физических, электронных), чтобы обеспечить их целостность, они должны быть защищены от внесения несанкционированных изменений;

— после проведения работ по контролю изменений должна быть разработана базовая версия, производная от ранее установленной базовой версии;

— базовая версия должна быть прослежена к той базовой версии, производной от которой она является, если при сертификации новой базовой версии используется сертификационное доверие к работам или документам процессов жизненного цикла, связанных с разработкой предшествующей базовой версии;

— базовая версия или единица конфигурации должны быть прослежены либо к выходным данным, которые они идентифицируют, либо к процессу, с которым они связаны.

Квалификационные базовые тесты предназначены для проверки корректности функционирования комплекса программ. Выполнение базовых тестов позволяет быть уверенным в правильной работе ***программного продукта***. Если тестирование прошло успешно, значит, отправленные и интегрированные изменения достаточно стабильны, статусы новых редакций ЕК можно повысить до рекомендованного базового уровня и сотрудникам разрешено использовать их при обновлении своего рабочего пространства.

После завершения сборки и создания базовой редакции программного продукта интеграционный поток разблокируется. Специалистам снова разрешается отправлять в версии ПС изменения и обновлять рабочие вер-

сии файлов ЕК новыми редакциями. Если есть средства для автоматического выполнения тестов с определением успешного или неуспешного их прохождения, можно автоматизировать этап квалификационного тестирования. Такой вариант подходит для крупных команд и больших проектов. Автоматизация данного процесса особенно важна на заключительных стадиях цикла разработки очередной базовой версии программного продукта, когда требуется жестко придерживаться запланированных сроков.

Составление отчетов о состоянии конфигурации должно начинаться с того момента, когда будут получены первые данные о комплекте ЕК. Отчеты о статусе конфигурации должны предоставлять информацию об идентификации конфигурации и всех отклонениях от зарегистрированных конфигурационных баз. Проверки конфигурации следует проводить до принятия и утверждения конфигурационной базы с целью гарантии того, что программный продукт соответствует контрактным требованиям спецификаций и что он точно отражен в документах по конфигурации. Проверка конфигурации может потребоваться для официальной приемки заказчиком комплекса программ. Существуют, как правило, ***два типа проверок документов конфигурации***:

— проверка функциональной конфигурации: официальная экспертиза с целью установления того, что единицы конфигурации имеют такие эксплуатационные и функциональные характеристики, какие требуются в документах по данной конфигурации ПС;

— проверка физической конфигурации: официальная экспертиза конфигурации непосредственно после сборки и изготовления комплекса программ с целью утверждения того, что он соответствует конфигурационным документам на продукцию.

Цель отчетов о состояния конфигурации заключается в обеспечении информации для управления конфигурацией процессов жизненного цикла ПС в отношении идентификации конфигурации, базовых версий, сообщений о дефектах и контролю изменений. Отчеты о дефектах, модификациях, прослеживаемости и корректирующих действий должны регистрировать несоответствие процесса требованиям спецификаций, планов и стандартов, отсутствие выходных данных процессов жизненного цикла ПС, аномальное поведение программных продуктов, а также гарантировать разрешение этих проблем:

— регистрацию идентификации единиц конфигурации, идентификации базовых версий, состояния сообщений о дефектах, хронологии изменений и состояния выпускаемой версии;

— определение информации, подлежащей сопровождению, и способов регистрации и ведения отчетности о состоянии конфигурации этой информации;

— должны быть подготовлены сообщения о дефектах и модификациях, которые описывают несоответствие процессов требованиям, планам, отсутствие выходных данных или аномальное поведение ПС, а также предпринятые корректирующие действия;

— отчетность о модификациях и дефектах должна предусматривать идентификацию затрагиваемых единиц конфигурации или определение затрагиваемых работ в процессах, утверждение и закрытие сообщений о дефектах;

— сообщения о дефектах, для которых требуются корректирующие действия в ПС или выходных данных процессов жизненного цикла, должны активизировать работы по контролю реализации изменений.

Разработчик УК должен создавать и сопровождать отчеты о состоянии конфигурации всех объектов, которые были помещены под управление конфигурации на уровне программного проекта. Эти отчеты должны поддерживаться в течение срока действия контракта. Они должны включать информацию о текущем состоянии выпускаемой версии каждого объекта, и информацию о состоянии изменений объекта с момента помещения под контроль конфигурации уровня проекта, а также состояние сообщений о дефектах и изменениях, оказывающих воздействие на данный объект.

Архивирование и тиражирование базовых версий программных продуктов и документов должны гарантировать использование исключительно санкционированных версий программного продукта и компонентов, так что официальные версии могут быть получены только из архива. Каждая единица конфигурации должна быть идентифицирована, документирована и выпущена ее официальная версия до того, как осуществляется производство ПС для пользователей. Цель работ по архивированию и применению документов — обеспечить получение документов жизненного цикла ПС, для копирования, повторной генерации, повторного тестирования и модификации программного продукта. Должны быть регламентиро-

ваны ***полномочия специалистов*** по выпуску базовых версий единиц конфигурации:

— документы жизненного цикла, связанные с программным продуктом должны быть получены из утвержденного источника от организации-разработчика;

— установить процедуры, призванные обеспечить целостность хранимых данных независимо от носителей, которые должны:

• гарантировать, что никакое несанкционированное изменение не может быть выполнено;

• выбирать физические носители данных, минимизирующие ошибки регенерирования и износа;

• проверять и/или обновлять архивные данные с частотой, соответствующей сроку службы физического носителя;

• хранить копии в физически отдельных архивах, что минимизирует риск потери данных в случае катастрофы;

— процесс копирования и тиражирования должен быть верифицирован, чтобы гарантировать получение точных копий, и должны существовать процедуры, гарантирующие безошибочное копирование исполняемого объектного кода;

— единицы конфигурации должны быть идентифицированы и выпущена их официальная версия, до того как осуществляется производство программного продукта;

— должны быть установлены полномочия по выпуску базовых версий единиц конфигурации и компонентов программного средства, загружаемого в вычислительную систему или оборудование;

— необходимо установить процедуры хранения, включения, удаления компонентов конфигурации, чтобы удовлетворить требования пригодности к применению и обеспечить возможность модификации ПС.

Финансирование управления конфигурацией программных средств должно определяться специальным договором (или разделом договора на разработку первичной версии ПС) между разработчиком и заказчиком. В техническом задании и в контракте следует четко определить порядок квалификации видов и причин изменений в программах и данных, а также распределение ответственности за их инициализацию, реализацию и финансирование. Выявленные ошибки в программах и данных, которые искажают реализацию функций, согласованных с заказчиком в контракте

и требованиях спецификаций, а также отраженные в документации на версию ПС, должны устраняться за счет разработчика. Модификацию и расширение функций компонентов или создание новых версий комплекса программ, ранее не отраженных в требованиях технического задания и контракте с заказчиком, следует квалифицировать как дополнительную работу с соответствующим финансированием заказчиком.

После передачи версии программного продукта в эксплуатацию затраты ресурсов на обнаружение и первичную квалификацию дефектов несут в основном непосредственные пользователи. На разработчиков (поставщиков) комплекса программ возлагаются затраты на анализ и локализацию причин дефектов и их устранение. Эти затраты зависят от характеристик выявляемых дефектов, от масштаба комплекса программ, организации и технологии его разработки, инструментальной оснащенности сопровождения, квалификации специалистов, а также от тиража и активности применения данного ПС.

16.2. Этапы и процедуры при управлении конфигурацией программных средств

Конфигурационное управление в значительной степени обеспечено, если ***ПС имеет четкую структуру***, а его компоненты — унифицированные интерфейсы по управлению и информации. Для этого правила модульно-иерархического построения ПС должны детализироваться, до уровня конкретных методик создания и оформления модулей, компонентов и межмодульного взаимодействия. При этом унифицированные межмодульные интерфейсы целесообразно, по возможности, упрощать и ослаблять, а также подготавливать условия для их проверок при реальном функционировании программ. Проектирование ПС сверху вниз и последующее сохранение четкой структуры интерфейсов значительно облегчают управление конфигурацией и продлевают срок жизни версий комплексов программ. Регистрация и учет истории этого процесса обеспечивает возможность его контроля и пошагового восстановления выполненных изменений (отката) при ***выявлении вторичных дефектов***, внесенных в процессе разработки модификаций для очередной базовой версии программного продукта. Такие дефекты обычно обусловлены одновременным, нескоординирован-

ным внесением групп изменений несколькими специалистами или потерей некоторых изменений в определенной версии ПС. Это возможно при одновременной разработке или корректировке различных версий ЕК, предназначенных для использования в различных, но определенных версиях сложных комплексов программ.

Модификация, учет и тиражирование версий требует больших затрат. Поэтому при выпуске каждой новой базовой версии разработчики стремятся обеспечить преемственность ее функций и компонентов с предыдущими версиями, а также рассматривается возможность и подготавливается решение для возможного прекращения модификаций некоторой устаревшей версии ПС или ее конкретных компонентов. В результате развития сложного комплекса программ среди всего множества версий для каждого ПС (или компонента) в архиве тиражирования и обеспечения сохранности образуется ***зона сопровождения*** — комплект конфигураций, доступных для изменений базовых версий программного продукта. Число таких сопровождаемых ***базовых версий разработчика конфигураций*** или глубина сопровождения практически всегда не менее двух версий и редко превышает четыре версии. Для крупномасштабных ПС это соответствует рациональному времени жизни и тиражирования каждой очередной базовой версии программного продукта около 1—2 лет.

В стандартах, регламентирующих конфигурационное управление ПС (см. Приложение 1), представлен ***комплекс процедур и состав специалистов***, организующих эти процессы, — рис. 16.2. Конкретное предприятие, в зависимости от стоящих перед ним задач и ЖЦ ПС, может выбрать соответствующую группу процессов и процедур для достижения своей конкретной цели управления конфигурацией. Множество процедур в стандартах сконструировано так, что возможна их ***адаптация в соответствии с характеристиками проектов и внешней среды ПС***. Для реализации на практике приведенных выше концепций и процедур, требований и планов сопровождения и управления конфигурацией программных средств необходимы организационные мероприятия, гарантирующие участникам проектов определенную культуру, дисциплину разработки и выполнения модификаций. Такая ***организационная система*** должна обеспечивать специалистам разной квалификации и роли в проекте возможность взаимодействия при решении требуемых комплексных задач, для накопления, хранения и обмена упорядоченной информацией о состоянии и изме-

нениях компонентов проекта. Формализация обмена информацией должна повысить ответственность специалистов за корректность ее содержания, оперативность формирования и изменения, качество процессов трансформации и реализации данных и документов в жизненном цикле ПС.

Рис. 16.2

В процессе эксплуатации n-й версии ПС у каждого m-го пользователя могут появляться некоторые претензии к ее функционированию, которые квалифицируются им как ошибки или дефекты эталонной или собственной версии (см. рис. 16.2). Для общения с пользователями и накопления информации о выявляемых недостатках в тиражируемых сложных ПС целесообразно выделение группы аналитиков высокой квалификации, овладевших всеми функциональными возможностями данного ПС.

Группа предварительной селекции — аналитиков предполагаемых изменений — должна иметь в своем зарегистрированном и аннотированном арсенале практически весь комплекс тестов, применявшихся при испытаниях опытного образца и предыдущих версий ПС для регрессионного тестирования. Тесты накапливаются, упорядочиваются и каталогизируются в базе данных тестирования. Они используются для контроля сохранности версий и установления достоверности дефектов, сообщенных пользователями. Эти же специалисты осуществляют развитие набора тестов для подтверждения наличия и локализации частных дефектов и ошибок, а также для первичной оценки целесообразности реализации предложений по развитию и модернизации программ. От пользователей или заказчика могут поступать предложения по внесению изменений в $(n + 1)$-ю версию для улучшения эксплуатационных характеристик и совершенствования функциональных возможностей ПС. Аналогичные предложения могут поступать от разработчиков комплекса программ. Для оценки предложений полезно экспериментальное тестирование предварительных компонентов и вариантов возможных изменений версии ПС.

Для ***установления достоверности*** сообщений пользователей о выявленных дефектах и ошибках необходима детальная регистрация сценариев и условий, при которых проявляются аномалии. Установление разработчиками достоверности ошибок начинается с тестирования эталонной базовой версии ПС при исходных данных m-го пользователя, обнаружившего дефект. Если проявляется ошибка, то она регистрируется как подтвержденная при зафиксированных тестовых данных. При отсутствии проявления ошибок на эталонной версии при тех же исходных данных целесообразно проводить последующее тестирование копии версии, адаптированной к условиям применения у m-го пользователя. Если и при этом ошибка не проявляется, то регистрируется ее неподтверждение, о

чем сообщается пользователю, и информация о дефекте снимается с учета. Если ошибка подтверждена на версии пользователя, то возможно, что эта версия была неправильно адаптирована к условиям применения. Для проверки может быть полезным повторение адаптации и повторное тестирование новой адаптированной версии. Тестирование новой адаптированной версии может подтверждать проявление ошибки, о которой информировал пользователь, и тогда ошибка регистрируется для последующего анализа и устранения. Если ошибка не подтверждается, то регистрируется неправильная адаптация версии пользователем и уточняется причина некорректной адаптации.

Многие предприятия для взаимодействия с их пользователями организуют так называемые ***«горячие линии»*** консультаций при затруднениях с некоторыми процедурами применения ПС. Эти консультации позволяют оперативно проводить первичную селекцию претензий пользователей к приобретенным, адаптированным и эксплуатируемым версиям программного продукта, обусловленных их некомпетентностью или недостатками эксплуатационной документации. Некоторые претензии могут иметь причиной попытки применения ПС за пределами условий, допускаемых документацией, и являются некорректными расширениями технических требований и спецификаций, что может разъясняться при консультациях. В результате оперативных консультаций выделяются правомочные претензии, которые являются следствиями дефектов поставляемых программных продуктов или их адаптации к характеристикам среды пользователей. По всем запросам пользователи должны оперативно получать конкретные компетентные рекомендации для преодоления возникших затруднений. Кроме того, по «горячей линии» могут поступать полезные предложения или идеи на совершенствование функций или повышение характеристик качества ПС, которые подлежат анализу и подготовке решений.

Идеи небольших корректировок программ целесообразно накапливать отдельно от предложений по существенному совершенствованию системы. Таким образом, последовательно формируется документ — ***описание выявленных дефектов и предложений по корректировке ПС и компонентов***, содержащий исходные данные для планирования доработок в процессе сопровождения:

— выявленные дефекты и ошибки в программах и данных;

— предложения по совершенствованию функций и улучшению качества эксплуатируемых версий программ;

— идеи и предполагаемая экономическая эффективность коренной модернизации и расширения функций ПС или его компонентов.

Ошибки и предложения по корректировке программ первоначально селектируются специалистами по компонентам ПС и анализируются советом конфигурационного управления по их влиянию на качество функционирования комплекса программ и по затратам на осуществление изменений. ***Приоритетность*** каждого предлагаемого изменения программ целесообразно оценивать по следующим критериям:

— насколько данное изменение может улучшить эксплуатационные характеристики ПС в целом;

— каковы затраты на выполнение корректировок при создании новой базовой версии и их распространение пользователям;

— возможно ли и насколько сильно влияние изменений на функциональные характеристики остальных компонентов версии ПС;

— какова срочность извещения пользователя о разработанной корректировке и целесообразно ли ее распространять до подготовки очередной базовой версии;

— для какого числа пользователей может быть полезно данное изменение;

— как данное изменение отразится на эксплуатации пользователями предыдущих версий;

— насколько подготовка и оперативное внедрение данного изменения может отразиться на сроках создания очередной базовой версии программного продукта.

Для повышения качества новых версий ***руководитель и совет конфигурационного управления*** анализируют все предлагаемые изменения и выделяют из них целесообразные для анализа возможного использования в очередной версии (см. рис. 16.2). При выделении изменений приходится решать оптимизационную задачу по оценке и сопоставлению ущерба от того, что изменение не проведено и не повышается соответственно качество функционирования ПС, с затратами ресурсов на проведение изменений и возможным ущербом, если они содержат дефекты. Селекция проводимых изменений в версиях сложных ПС требует формализации анализа

этого процесса и документирования предполагаемых и утвержденных изменений.

В совете конфигурационного управления сосредоточивается информация для планирования основных операций по доработке и выпуску очередных версий ПС. Специалисты должны оценивать степень срочности исправления ошибок и проведения модернизаций, а также выявлять условия, позволяющие достаточно полноценно эксплуатировать программы до выполнения всех запланированных изменений при выпуске очередной версии. Кроме того, специалистам группы управления конфигурацией необходима «дипломатическая» квалификация для того, чтобы убеждать пользователей до выпуска очередной версии некоторое время использовать ПС без проведения изменений, возможно, с некоторыми ограничениями или видоизменениями функций.

В процессе разработки (n + 1)-й базовой версии ПС используются вновь разработанные компоненты и версии i-х компонентов и модулей, переписываемых из предыдущей n-й версии. После внесения изменений из этих компонентов образуются j-е версии для комплексирования функциональных групп программ, которые после автономного тестирования объединяются в (n + 1)-ю версию ПС для квалификационного тестирования, испытания и документального оформления. При корректировках компонентов, групп программ и версий ПС все процедуры выполняются ***с учетом ограничений права доступа каждого специалиста*** к реализации соответствующих изменений (см. табл. 16.1). Все версии разработчиков должны сопровождаться дубликатами, которые эпизодически тестируются на соответствие основной версии разработчика модификаций на данном уровне. При выявлении отклонения дубликата от основной версии разработчика тестирование продолжается до установления места и причины различия. После этого осуществляется корректировка дубликата или основной версии до абсолютного совпадения.

Корректировку компонентов и сборку очередной базовой версии производят специалисты, ответственные за интеграцию, по возможности, с привлечением разработчиков предыдущих версий. Пользователи в этих работах не участвуют, но могут привлекаться для предварительного испытания и уточнения целей корректировок (Альфа-тестирование). Процесс проведения изменений должен поддерживаться средствами автоматизации конфигурационного управления и обеспечения защиты от несанкцио-

нированного доступа к корректируемым компонентам на последовательных этапах выполнения изменений. Каждое разработанное изменение должно документироваться в ***описании и базе данных предполагаемых корректировок*** с указанием содержания, автора, времени и степени срочности.

Для принятых к внедрению изменений разрабатывается ***план доработок программ*** и определяется конкретный специалист, ответственный за каждую корректировку программы. Изменения программ могут потребовать либо полной замены модуля или группы программ, либо небольшого изменения текста программного модуля, описания данных или констант. При полной замене компонентов ПС они подлежат тестированию, как все вновь созданные компоненты. Изменения, выделенные и утвержденные уполномоченным лицом для реализации, селектируются по приоритетам на группы:

— срочные изменения, которые должны не только быть внесены в очередную (n + 1)-ю базовую версию ПС, но и сообщены пользователям для оперативной корректировки программ до внедрения очередной официальной версии;

— изменения, которые целесообразно внести в (n + 1)-ю базовую версию с учетом затрат на их реализацию и степени улучшения функций и качества ПС;

— изменения, которые требуют дополнительного анализа целесообразности и эффективности их реализации в последующих базовых версиях и могут пока не внедряться в очередную (n + 1)-ю версию ПС;

— изменения, которые не оправдывают затрат на разработку и выполнение корректировок или практически не влияют на качество и эффективность ПС и поэтому пока не подлежат реализации.

Эти группы изменений регистрируются в ***описании и базе данных подготовленных корректировок*** для последующего использования с указанием специалистов, их подготовивших, и лиц, принявших решение о реализации. Подготовленные и утвержденные предложения на изменения ПС поступают к специалистам, осуществляющим планирование их внесения в реальные программы, а также разработку и корректировку конкретных компонентов и документов. После выполнения доработок разработчиками компонентов и корректировки документации следует окончательная проверка корректности изменений, которая осуществляется ***специалистами***

по квалификационному тестированию и испытаниям очередной версии ПС (см. рис. 16.2).

Тестирование необходимо сосредоточивать на компонентах, впервые вводимых или значительно модифицируемых в данной версии. Если изменения в программе или в данных невелики, то тестирование обычно можно ограничить компонентами, непосредственно связанными с выполненной корректировкой. При этом следует учитывать, что корректировки программ в 10—30% случаев сами содержат ошибки и требуют тщательного тестирования не только тех частей, где внесены изменения.

Наличие в сложных программах глубоких межмодульных связей по управлению и по информации вызывают необходимость частичного тестирования тех компонентов и их интерфейсов, где по первому впечатлению корректировки не оказывают влияния. Такие связи зачастую приводят к проявлению дефектов вследствие проведенных изменений и нарушения концептуальной и/или функциональной целостности группы взаимодействующих программных компонентов и данных. Поэтому ***регрессионному тестированию*** необходимо подвергать также некоторые части и интерфейсы комплекса программ, которые не подвергались изменениям. Для этого могут использоваться тесты, ранее применявшиеся при испытаниях предыдущей n-й версии. Проверенные, таким образом, изменения регистрируются в ***описании и базе данных утвержденных и проведенных корректировок для $(n + 1)$-й базовой версии программного продукта или для срочных извещений пользователям***.

Объединение и сборка функциональных компонентов — групп откорректированных программ позволяет создать ***эталон базовой $(n + 1)$-й версии ПС***, подлежащий квалификационному тестированию по полной программе испытаний. При большом количестве выполненных изменений сложность испытаний может приближаться к испытаниям первой базовой версии. Объем тестирования при испытаниях $(n + 1)$-й базовой версии должен согласовываться между разработчиком, заказчиком или с пользователями. Результаты испытаний регистрируются в типовых протоколах и в проекте акта о завершении испытаний $(n + 1)$-й версии. Все проверенные и подтвержденные при испытаниях корректировки программ регистрируются и утверждаются уполномоченным руководителем конфигурационного управления в акте, определяющем завершение подготовки новой базовой версии программного продукта.

Конфигурационная база программного продукта должна быть зарегистрирована официально в определенный момент времени и использоваться в качестве отправной точки для контроля за состоянием конфигурации и утвержденными изменениями (n + 1)-й версии. После утверждения конфигурационной базы уполномоченным лицом (и, возможно, заказчиком) оформляются документация и физические носители подлинника (n + 1)-й версии, которые передаются на тиражирование и внедрение у пользователей, а также в архив базовых версий данного проекта программного продукта. Подлинник снабжается техническими условиями и тестами для проверки сохранности и функциональной работоспособности в базе данных отчетов тиражирования и обеспечения хранения версий ПС. Кроме того, в отчетах должны быть сведения о составе всей документации на версию ПС, основные результаты испытаний и сведения о должностных лицах, утвердивших базовую версию программного продукта.

Пользователям может поставляться копия (n + 1)-й базовой версии для самостоятельной ***адаптации*** к характеристикам внешней среды и особенностям конкретного применения. В этом случае адаптация должна проводиться строго по инструкциям разработчика этой версии и должны быть запрещены любые дополнительные изменения за пределами, предписанными документацией. Подобные изменения автоматически снимают гарантии разработчика, и ответственность за корректность адаптации возлагается на пользователя. Однако возможны ситуации, когда для адаптации требуется особенно высокая квалификация специалистов и ее следует проводить в рамках работ по конфигурационному управлению непосредственными разработчиками базовой версии. Тогда они входят в группу работ коллектива по сопровождению и конфигурационному управлению. В этом случае пользователь должен представить разработчику формализованные исходные данные и характеристики внешней среды, в соответствии с которыми следует проводить адаптацию. Результаты выполненной адаптации и особенности версий пользователей в крупных, критических информационных системах при относительно небольшом тираже целесообразно регистрировать в архиве у разработчиков соответствующей базовой версии. Ответственность за корректность адаптации в этом случае несет разработчик базовой версии программного продукта.

Для независимого удостоверения достигнутого качества проведенных модификаций в испытанной и утвержденной разработчиком версии

программного продукта, по заявке разработчика, заказчика или пользователей, она может подвергаться обязательной или добровольной ***сертификации*** (см. лекцию 18). Комплект поставки, состоящий из копии физических носителей и эксплуатационной документации, а также применявшиеся разработчиками квалификационные тесты и методики испытаний передаются сертификационной лаборатории. При сертификационных испытаниях допускается расширение набора и параметров тестов, однако в пределах, ограниченных технической документацией на конкретную версию ПС. Успешно проведенные испытания и полученный сертификат качества подтверждают соответствие комплекса программ документации, надежность и безопасность применения (n + 1)-й версии до тех пор, пока в нее не будут внесены какие-либо изменения. Сертификат может быть аннулирован при любых, даже внешне незначительных, корректировках версии программного продукта.

Приведенную схему организации жесткого конфигурационного управления версиями ПС не всегда можно и целесообразно реализовать полностью. Наличие срочных исправлений ошибок и запросов пользователей на частичные изменения в эксплуатируемых версиях, а также ряд других причин могут приводить к появлению между n-й и (n + 1)-й базовыми версиями ПС ряда промежуточных пользовательских версий. Для этого, наряду с выпуском очередных, базовых версий, приходится выпускать ***временные извещения на исправления выявленных ошибок и на корректировки***, а также извещения об основных изменениях функций, которые проведены в (n + 1)-й версии по сравнению с n-й. В результате у пользователя могут создаваться промежуточные версии, каждая из которых кроме адаптации к характеристикам среды эксплуатации содержит специфический набор изменений из состава вводимых в очередную базовую версию ПС.

Появление промежуточных версий у пользователей может сопровождаться ошибками, обусловленными опасным разрушением концептуальной целостности взаимосвязей компонентов ПС, вследствие некоторых противоречивых изменений. Такие ошибки являются следствием того, что корректировки программ для исправления ранее выявленных ошибок и для развития функциональных возможностей (n + 1)-й базовой версии могут быть взаимозависимы и частично альтернативны как непосредствен-

но, так и через некоторые промежуточные программы и данные, что трудно обнаруживать. При подготовке (n + 1)-й версии эти связи между корректировками могут не проявляться, однако они должны проверяться в процессе комплексного тестирования. В то же время корректность каждого отдельного изменения может автономно не полностью испытываться при подготовке версии для конкретного пользователя. В результате предлагаемые пользователями отдельные изменения оказываются между собой не полностью согласованными, что может приводить к ошибкам функционирования версии m-го пользователя. Особое значение синхронизация выполнения изменений ПС приобретает в распределенных системах типа клиент-сервер при наличии центра системы и множества удаленных, взаимодействующих клиентов с различными функциями, которые в одно и то же время эксплуатируют различные версии некоторого ПС.

На совет конфигурационного управления дополнительно возлагаются функции выявления связанных изменений и учета эксплуатации промежуточных версий с частичными изменениями у пользователей. Вследствие этого для сложных ПС может появляться необходимость конфигурационного учета и управления не только базовыми — эталонными версиями, но и рядом промежуточных версий пользователей с частичными изменениями. Сложность сопровождения и управления конфигурацией при этом резко возрастает. Поэтому в большинстве случаев целесообразно принимать организационные меры по ограничению числа частных распространяемых изменений между n-й и (n + 1)-й базовыми версиями в пределах только простейших локальных корректировок ошибок, существенно влияющих на качество конкретных версий ПС. Для предотвращения развала взаимодействия версий клиентских программ в распределенной системе ***следует запрещать несанкционированные изменения пользовательских версий ПС***, минуя совет конфигурационного управления.

Для эффективной реализации перечисленных процессов представленная схема организации конфигурационного управления должна быть поддержана ***службой и базой данных тиражирования, архивирования и гарантированного хранения*** всей необходимой информации об объектах, их корректировках, версиях, авторах и их правах на изменения. Эта служба должна обеспечивать поставку пользователям только утвержденных копий версий ПС и/или их компонентов с полным, адекватным комплектом

эксплуатационных документов, а также извещений на частные изменения конкретных, ранее приобретенных версий. Для гарантированного сохранения состояния и результатов модификаций программ и обеспечения возможности их анализа на любой стадии проекта, а также ***отката по истории выполненных корректировок***, следует организовать четкую систему хранения и копирования подлинников, дубликатов и копий одного и того же программного продукта и документов. Эта система должна иметь соответствующих руководителей и специалистов, ответственных за полную сохранность всех результатов истории сопровождения и изменений конфигурации конкретного программного продукта в течение заданного интервала времени.

Для гарантированного сохранения от случайного или преднамеренного разрушения базовой версии ПС в составе коллектива конфигурационного управления должны быть выделены специалисты, ответственные за сохранность подлинников физических носителей и документации испытанных и утвержденных версий программного продукта. Целесообразно выделять ***специальный архив с резко ограниченным доступом***, в котором накапливать подлинники базовых версий и дополнительные данные, необходимые для гарантии их сохранности и возможности восстановления. Для повышения надежности независимо сохраняются все изменения, внесенные в n-ю версию при подготовке $(n + 1)$-й базовой версии ПС. Благодаря этому в аварийном случае разрушения подлинника, дубликата и всех копий $(n + 1)$-й версии ПС должна обеспечиваться возможность их восстановление на базе предыдущей версии и зарегистрированных изменений. Кроме того, сохранение изменений в некоторых случаях облегчает обнаружение и локализацию ошибок, которые проявились в $(n + 1)$-й версии и отсутствовали в предыдущих версиях.

При переносе базовых версий программного продукта на иные аппаратные и/или операционные платформы процедуры изменения управления конфигурацией должны проводиться в соответствии с планом, который включает: извещение пользователей, обучение персонала, предупреждение о проведении и завершении переноса архива, оценивание влияния новой среды и архивирование соответствующих данных. После планирования переноса пользователям должно быть направлено уведомление о планах и работах по переносу базовых версий ПС и управления конфигу-

рацией на новую платформу. В содержание уведомления должны быть включены: объяснение того, почему прежнюю внешнюю среду УК нельзя больше поддерживать; описание новой аппаратной и операционной среды с указанием даты, с которой она доступна для пользователей; описание других доступных вариантов применения ПС в случае прекращения поддержки прежней среды УК. Для плавного перехода в новую среду параллельно могут выполняться работы пользователями в прежней и новой среде. В течение этого периода необходимо ***обучение пользователей*** для работы с УК и архивом базовых версий программного продукта в новой среде.

Внедрение программного продукта через специализированный архив предполагает широкие масштабы и промышленный подход к тиражированию документации и программных средств, к поставке, обслуживанию, управлению конфигурацией и сопровождению. Передача ПС в архив завершается приемо-сдаточными испытаниями, которым предшествуют опытная эксплуатация и испытания разработчика. Кроме того, разрабатывается учебно-методический план, подготавливаются учебные пособия, необходимые для обучения массового пользователя на курсах, а также проводится обучение выделенной группы специалистов архива, ответственных за последующее обучение коллективов пользователей и сопровождение ПС.

В каждом крупном проекте комплекса программ должен быть организован регламентированный процесс управления конфигурацией и сопровождения, обеспечивающий для коллектива специалистов единую ***среду разработки, хранения, изменения и утверждения модификаций***, адекватных реальному содержанию объектного кода программ и текстовых данных файлов проекта ПС. Процесс организации и технологического обеспечения УК должен быть ориентирован на слаженную, коллективную работу различных профессионалов, объединенных единой целью развития и модификации требуемого заказчиком комплекса программ с заданными функциями и высоким качеством. Каждый участник проекта в соответствии со своими функциональными обязанностями должен иметь доступ к необходимой для него ***корректной информации*** и ограничен возможностями обращения к несанкционированным для него данным.

16.3. Технологическое обеспечение при сопровождении и управлении конфигурацией программных средств

Технологической основой сопровождения и управления конфигурацией ПС являются ***системы управления базами данных*** (СУБД), адекватные целям и функциям проектов, структурированные по целям, назначению и содержанию данных в выделенных подсистемах хранения (рис. 16.3). Они должны обеспечивать возможность управления организационной и проектной деятельностью коллективов специалистов, универсальное хранилище в них необходимых данных, с инструментами наполнения, корректировки, поиска и контроля информации, соответствующей их профессиональной деятельности. Должны быть упорядочены деловые коммуникации между специалистами разных категорий, управление динамическими процессами выполнения изменений и транспортировки корректировок между подсистемами в соответствии с целями их использования специалистами.

Первоначально должен быть разработан ***проект архитектуры системы технологического обеспечения управления конфигурацией, а также Руководство по ее применению***, настроена выбранная СУБД на управление основными взаимодействующими подсистемами базы данных, с учетом класса и масштаба предполагаемого проекта ПС (рис. 16.4). По мере развития жизненного цикла проекта комплекса программ подсистемы базы данных сопровождения и УК должны поэтапно заполняться реальными данными от заказчика и разработчиков соответствующих квалификаций и контролироваться менеджерами проекта. При этом следует управлять динамикой процессов реализации процедур модификации, регистрировать реальное использование ресурсов специалистов, текущее время выполнения процедур развития проекта и оформления изменений в подсистемах БД.

Эта ***информация в подсистемах базы данных сопровождения и УК*** должна быть защищена от случайных и преднамеренных искажений путем организованного санкционирования, дублирования и контроля модификаций, истории их создания и изменения, в процессах жизненного цикла ПС. Необходимо гарантировать сохранность версий изменений, с учетом их важности для результатов всего проекта. Особенно защищен-

ным от искажений и разрушения следует сохранять ***архив базовых версий программных продуктов***, прошедших успешные испытания, утвержденных заказчиком и скрепленных его подписью. Для устранения дефектов, реализации корректировок и ошибок при развитии новых базовых версий целесообразно выделять рабочую копию предшествовавшей базовой версии и архив накопленных изменений, обеспечивающих возможность «***отката***» к предыдущей базовой версии в случае разрушительных некорректных изменений в процессе разработки новой базовой версии.

Рис. 16.3

Рис. 16.4

Такая ***система обеспечения информацией процессов сопровождения и управления конфигурацией*** может быть структурирована в соответствии с адаптированной версией жизненного цикла конкретного ПС. В соответствии с основными задачами специалистов проекта на рис. 16.3 представлены частные подсистемы базы данных информационного обеспечения модификаций, ориентированные на определенные процессы и компоненты ЖЦ комплексов программ. Для каждой подсистемы целесообразно выделять достаточно автономную базу данных компонентов ПС с ограниченным доступом только для определенных категорий специалистов (см. табл. 16.1). Эти фрагменты базы данных могут быть построены на стандартизированной основе СУБД проекта, взаимодействовать с аналогичными по структуре предшествующей и последующей базами данных. Они должны накапливать и содержать основные компоненты и документы проекта на

соответствующем уровне жизненного цикла ПС. Интерфейсы этого взаимодействия баз данных должны быть стандартизированы, по возможности ограничены по объему и доступности обмениваемой текущей и отчетной информации для других категорий специалистов. Для каждого сложного проекта комплекса программ целесообразно оформлять и утверждать ***Руководство и схему базы данных, обеспечивающей документооборот и управление сопровождением и конфигурацией ПС***, а также категории ответственных лиц за их поэтапную реализацию, контроль и сохранность информации.

Выше подчеркивалось, что проводимый анализ сопровождения программных средств в основном ориентирован на жизненный цикл сложных проектов комплексов программ высокого качества. Многие проекты ПС являются более простыми, и их процесс документооборота сопровождения может быть значительно сокращен. Для этого целесообразно проводить адаптацию и формировать ***практическую рабочую версию Руководства сопровождения и управления конфигурацией*** конкретного проекта ПС, которая должна регламентировать работы специалистов. Последовательное сокращение подсистем базы данных и компонентов обеспечения сопровождения начинается с определения масштаба и наличия предыстории проекта (см. рис. 16.4). Известные функции, потенциальные пользователи и концепции существующих версий ПС позволяют прогнозировать направления совершенствования и уменьшения модификаций для нового проекта или базовой версии, имеющих прототип.

В процессе реализации проекта производится наполнение базы данных реальными требованиями и характеристиками результатов разработки модификаций, и архивация откорректированных компонентов и отчетов выполненного проекта. При этом фиксируются корректировки и исправления дефектов и ошибок и оформляются комплекты документов базовых версий программных продуктов, поставляемых заказчику. Эти процедуры целесообразно выделять в отдельную подсистему БД — сопровождения, конфигурационного управления версиями и корректировками программного продукта, для чего формировать группу специалистов, которые могут быть организационно автономными от остальных подсистем сопровождения и даже размещаться на другом предприятии (см. рис. 16.4).

Документирование развития и совершенствования ПС в значительной степени влияет на достигаемое качество версий сложных комплексов

программ, трудоемкость и длительность их создания. Организация документирования должна определять стратегию, стандарты, процедуры, распределение ресурсов и планы создания, изменения и применения документов на программы и данные систем. Для этого должны быть выделены ***руководители и коллективы специалистов, которые должны планировать, утверждать, выпускать, распространять и сопровождать комплекты документов***. Они должны стимулировать разработчиков программных средств, компонентов и их изменений, осуществлять непрерывное, регламентированное документирование процессов и результатов своей деятельности, а также контролировать полноту и качество отчетных документов.

Структура документации и формы отдельных документов, используемых для конфигурационного управления и сопровождения программ, должны позволять точно документально описывать и идентифицировать каждую оформленную версию программных компонентов и ПС в целом в любое время на всем протяжении их жизненного цикла. Стандарт **ISO 9126** регламентирует ***характеристики качества*** программного продукта, его изменений и документов. Один из шести основных показателей качества жизненного цикла ПС (см. лекцию 11), важный для документирования сопровождения, характеризует ***практичность — применимость***: свойства программного средства, его модификаций и документации, отражающие сложность их понимания, изучения и использования, для квалифицированных специалистов при применении в указанных условиях. В жизненном цикле ПС характеристика качества практичности, доступности для понимания и освоения документации должна использоваться и учитываться специалистами с двух позиций:

— ***разработчиками модификаций*** и версий комплексов программ, для которых необходимо адекватное отражение и восприятие в документах состояния и изменений ПС и их компонентов в процессе сопровождения и управления конфигурацией;

— ***пользователями измененных и утвержденных документов***, поставляемых базовых версий программных продуктов в процессе их применения и эксплуатации.

Требования к практичности и ее субхарактеристикам — понятности и простоте использования при сопровождении, зависят от назначения и функций модификаций ПС и могут формализоваться заказчиками набором

свойств, необходимых для обеспечения удобной и комфортной модификации и улучшения версий комплекса программ. Количественно простоту изменений при сопровождении можно характеризовать требованиями допустимой средней длительности разработки и ввода типовых корректировок при устранении дефектов и совершенствовании функций ПС.

Требования к продолжительности изучения и разработки модификаций и документов, достаточной для эффективного сопровождения ПС квалифицированными специалистами, могут составлять часы или недели. Для обеспечения полноценного изучения процессов сопровождения и возможностей применения новых версий ПС этими специалистами необходима документация, объем которой существенно зависит от назначения и дополнительных функций ПС и может быть задан на основе анализа прецедентов подобных успешных проектов.

Понятность: свойства ПС и документации, обеспечивающие специалистам сопровождения понимание, является ли разработка модификации или новой версии комплекса программ пригодной для целей заказчика и как ее можно использовать для конкретных задач и условий применения. Она должна обеспечиваться корректностью и полнотой описания в документах исходной и результирующей информации, а также всех деталей новых функций ПС для заказчика и потенциальных пользователей.

Простота использования: возможность специалистам сопровождения удобно и комфортно изменять и управлять конфигурацией ПС. Аспекты изменяемости, адаптируемости и легкости инсталляции корректировок могут быть предпосылками для простоты использования и сопровождения документов конкретного ПС.

Изучаемость: свойства версий и документов ПС, обеспечивающие удобное освоение его сопровождения достаточно квалифицированными пользователями. Она может определяться трудоемкостью и длительностью подготовки специалистов к полноценному управлению конфигурацией ПС.

План и поддерживающее его ***Руководство по документированию сопровождения и конфигурационного управления*** конкретного крупного проекта ПС (рис. 16.5) должны отражать:

— общую структуру комплекта документов на конфигурацию ПС;

— номенклатуру и структуру содержания каждого документа;

— требования к качеству, оформлению и обозначению документов;

— регламент комплектования, корректировки и хранения документов;

— правила обращения, процессов изменения и сопровождения документов;

— графики подготовки, проверки, редактирования, согласования, утверждения и распространения документов.

Рис. 16.5

Для управления конфигурацией, развитием комплекса программ и испытаний в базе данных документов должны собираться сведения, которые состоят из следующих крупных ***функциональных частей***:

— описания данных об отказах, дефектах и ошибках, условиях их проявления и характеристиках обнаруживающих тестов, а также предложения на изменения программ, подлежащие анализу и селекции для выделения тех из них, для которых будут разрабатываться корректировки программ — ***описания предполагаемых изменений*** ПС;

— разработанные изменения программ, отобранные аналитиками сопровождения и конфигурационного управления для проведения корректировок в очередной версии ПС — ***описания подготовленных и утвержденных корректировок*** компонентов и версий комплексов программ;

— сборки компонентов, откорректированные версии и наборы изменений, выполненных в каждой из них, — ***описания откорректированных и утвержденных базовых версий программного продукта***;

— характеристики и параметры пользователей, которым переданы для использования соответствующие базовые версии и особенности внешней среды эксплуатации у них, — ***описания параметров адаптации пользовательских версий программного продукта***.

Перечисленные документы в базе данных управления конфигурацией проекта реально структурируются на более мелкие фрагменты. Они взаимосвязаны, и сведения об изменениях по мере обработки должны переходить последовательно из одного состава описания в другой и, возможно, из одного сектора базы данных проекта в другой (см. рис. 16.3).

Методика оформления отчетов о выявленных дефектах, ошибках и предложениях по корректировке версий ПС должна содержать рекомендации испытателям и пользователям по выявлению, регистрации и формализации условий проявления и содержания дефектов и желательных модификациях испытываемых и/или эксплуатируемых версий. Эта методика должна включаться в состав эксплуатационной документации и передаваться каждому пользователю версии программного продукта. В методике следует стимулировать специалистов анализировать, подготавливать и представлять рекомендации заказчику и разработчикам по совершенствованию и развитию функций и качества поставляемой версии ПС. Результаты анализа и предложения необходимо передавать в унифицированной форме ***Отчетов специалистов о выявленных дефектах и предложениях по корректировке*** ПС, которые должны содержать:

— подробное описание сценария тестирования и исходных данных, при которых выявлен дефект;

— описание проявления дефекта и документы результатов его регистрации;

— предположение о возможной причине, вызвавшей проявление дефекта;

— предложение о возможной модификации ПС и его компонентов для устранения дефекта или совершенствования качества функционирования комплекса программ.

Изменения, отобранные аналитиками сопровождения и управления конфигурацией, переводятся из описаний предварительных изменений в другую часть базы данных. Здесь изменения конкретизируются вплоть до текстов корректировок программ или созданных новых компонентов. Кроме того, для каждой подготовленной корректировки следует регистрировать результаты ее рассмотрения, проверки и утверждения на введение в новую базовую версию ПС или для частных извещений пользователям. Отвергнутые корректировки возвращаются в состав предлагаемых изменений.

Описание выявленных дефектов и предложений по совершенствованию функций комплекса программ, а также результатов анализа предполагаемых корректировок версий ПС должно содержать отчеты пользователей о выявленных дефектах и предложениях и дополнительно:

— оценки сложности, трудоемкости, эффективности и срочности модификаций программ;

— оценки возможного влияния предлагаемых изменений на эксплуатацию версий ПС, имеющихся у пользователей.

Все корректировки, утвержденные для введения в очередную версию, следует регистрировать в отдельной подсистеме базы данных (см. рис. 16.3). Для каждого изменения должны документироваться содержательная аннотация, а также общие характеристики и достигнутые на предварительных испытаниях показатели качества очередной версии программного продукта. Результаты испытаний версии запоминаются вместе с условиями и параметрами, при которых они формировались, а также с набором тестов или указаниями места их хранения.

Описание подготовленных и утвержденных корректировок, а также реализованных изменений и обобщенных характеристик модифицированной базовой версии программного продукта должно содержать:

— причину изменения программ и базы данных (ошибка, дефект, совершенствование);

— содержание изменений программ и базы данных, а также документации на версию ПС или компонента;

— результаты квалификационного тестирования базовой версии программного продукта с предполагаемыми изменениями;

— результаты испытаний и обобщенные характеристики качества базовой версии программного продукта после внесения изменений;

— решение по распространению пользователям проведенной модификации или версии программного продукта;

— адрес хранения корректировок, документов и квалификационных тестов новой базовой версии программного продукта.

Учет тиражирования, адаптации, переноса на иные платформы и распространения версий программного продукта должны осуществляться с фиксированием документов в базе данных ***описаний пользовательских версий***. В этом документе накапливаются сведения об операционных и аппаратных платформах, а также о параметрах внешней среды применения и адаптации ПС у каждого пользователя и активность его работы с версиями комплекса программ.

Накопленные документы об изменениях и история корректировок подлежат хранению в архиве в течение всего жизненного цикла ПС или значительной его части. Разрушение сведений о выполненных или предполагаемых изменениях программ может приводить к большим затратам на их восстановление. Поэтому база данных архива изменений должна дублироваться и поддерживаться методами и средствами сопровождения, аналогичными применяемым для основной документации, тестов и текстов программ конкретного проекта.

Особое значение при сопровождении и управлении конфигурацией имеет ***документация на реализованные изменения и тесты,*** с помощью которых проверялась корректность версий компонентов и ПС в целом. Эта документация должна позволять восстанавливать ***историю разработки и проверки*** каждого изменения любого компонента. На базе всего комплекса использованных тестов создается и документируется для каждой версии программного продукта эталонная тестовая (контрольная) задача и контрольные результаты ее решения. Эти документы оформляются в соответствии со стандартами, тиражируются и передаются пользователям вместе с программами базовой версии и остальными эксплуатационными документами.

Разработка и тестирование изменений компонентов и ПС всегда несколько опережают их документальное оформление. В течение этого времени возможны отдельные уточнения изменений в версиях. В результате документация должна непрерывно «***догонять***» реальное состояние программного продукта. Для упорядочения этого процесса стандартами установлена возможность оперативного выпуска ***предварительных, официальных извещений на частные изменения***. Эти извещения регистрируются как временные и погашаются при полном оформлении документации на очередную версию программного продукта и все изменения.

ЛЕКЦИЯ 17

ДОКУМЕНТИРОВАНИЕ ПРОГРАММНЫХ СРЕДСТВ

17.1. Организация документирования программных средств

Документация является органической, составной частью программного продукта для ЭВМ, и требуются значительные ресурсы для ее создания и применения. Тексты и объектный код программ для ЭВМ ***могут стать программным продуктом*** только в совокупности с комплексом документов, полностью соответствующих их содержанию и достаточных для его освоения, применения и изменения. Для этого документы должны быть корректными, строго адекватными текстам программ и содержанию баз данных — систематически, структурированно и понятно изложены, для возможности их успешного освоения и использования достаточно квалифицированными специалистами различных рангов и назначения. Качество и полнота отображения в документах процессов и продуктов в жизненном цикле программных средств должны полностью определять достоверность информации для взаимодействия заказчиков, пользователей и разработчиков, а тем самым корректность функций и достигаемое качество программных продуктов и соответствующих систем. Посредством документов (электронных или бумажных) специалисты взаимодействуют с программными средствами и данными в реализующих их вычислительных машинах, а также между собой.

Существует большая разница между тем, чтобы просто написать и запрограммировать некоторую функцию для индивидуального использования ее разработчиком, и тем, чтобы изготовить ее как качественный программный продукт, отчуждаемый от разработчиков, поставляемый за-

казчику и пользователям. Создание программного продукта требует значительных организационных усилий, ибо ее документация — это сложный живой организм, подверженный постоянным изменениям, которые могут вноситься многими специалистами. ***Управление документацией*** должно непрерывно поддерживать ее полноту, корректность и согласованность с программным продуктом. Необходимо обеспечивать возможность достоверного, формально точного общения всех участников проекта ПС между собой, с создаваемым продуктом и с документами для гарантии поступательного развития, совершенствования и применения комплекса программ. Адекватность документации требованиям, состоянию текстов и объектных кодов программ должна инспектироваться и удостоверяться (подписываться) ответственными руководителями и заказчиками проекта. ***Ошибки и дефекты документов не менее опасны для применения ПС***, чем ошибки в структуре, интерфейсах, файлах текстов программ и в содержании данных. Поэтому к разработке, полноте, корректности и качеству документации необходимо столь же тщательное отношение, как к разработке и изменениям текстов программ и данных.

Реализация документов ПС в значительной степени определяет достигаемое качество сложных программных продуктов, трудоемкость и длительность их создания. Для этого должны формироваться и использоваться ***регламентированная стратегия, стандарты, распределение ресурсов и планы создания, изменения и применения документов*** на программы и данные сложных систем. В общем случае должны быть ***выделены руководители и коллектив специалистов***, которые будут планировать, описывать, утверждать, выпускать, распространять и сопровождать комплекты документов. Они должны стимулировать разработчиков ПС осуществлять непрерывное, полноценное документирование процессов и результатов своей деятельности, а также контролировать полноту и качество исходных, результирующих и отчетных документов ЖЦ ПС. Официальная, описанная и утвержденная стратегия документирования должна ***устанавливать дисциплину***, необходимую для эффективного создания высококачественных документов на продукты и процессы в жизненном цикле ПС.

Методы и средства документирования каждой процедуры в стандартах обычно не раскрываются и адресуются к специальным нормативным документам различного уровня. Быстро оснащающиеся различными методами и инструментальными средствами этапы системного анализа, мо-

делирования и проектирования ПС различных классов и назначения затрудняют стандартизацию этих процессов, достаточную для полной формализации структуры и содержания документов на программы и данные на уровне международных стандартов. Поэтому для этих этапов создаются нормативные документы — ***шаблоны*** на уровне стандартов де-факто, использующие, адаптирующие и дополняющие компоненты стандартов де-юре в разумной степени. Такие нормативные документы содержат выделенные фрагменты стандартов ЖЦ ПС и других стандартов, ***регламентирующих шаблоны программных документов на различных этапах проекта***. В результате создаются и применяются проблемно-ориентированные совокупности методических руководств, отражающие наиболее современные методы, формы и фрагменты документов для документальной поддержки этапов и процессов жизненного цикла ПС, определенного класса или функционального назначения.

При создании и применении сложных ПС и обеспечении их жизненного цикла документами целесообразно ***применять выборку из совокупности стандартов***, представленных в Приложении 1, детализирующих частные процессы, работы и документы. В результате на начальном этапе проектирования следует выделять и формировать целесообразный ***комплект шаблонов документов, обеспечивающих*** регламентирование всех этапов, процессов и документов при создании определенных проблемно-ориентированных проектов ПС. Для создания и реализации положений этих документов должны быть выбраны инструментальные средства, совместно образующие взаимосвязанный комплекс технологической поддержки и автоматизации ЖЦ и не противоречащие предварительно скомпонованному набору нормативных документов.

Процессы документирования программ и данных входят в весь жизненный цикл сложных систем и ПС. Поэтому организация и реализация ***работ по созданию документов должны распределяться между специалистами***, ведущими непосредственное и преимущественное создание проектов комплексов программ, и специалистами, осуществляющими в основном разработку, контроль и издание документов. При создании особо сложных систем целесообразно выделение специального коллектива, обеспечивающего организацию и реализацию основных системных работ по документообороту ПС. Совокупные затраты на документирование крупных программных продуктов могут достигать 20—30% от общей трудоем-

кости проекта и необходимого числа (десятки) специалистов в жизненном цикле проекта ПС. В более простых случаях организация работ может быть упрощена, затраты на документирование снижаются приблизительно до 10%, однако всегда целесообразно выделять специалистов, непосредственно ответственных за создание, адекватность и контроль полноценного комплекта документов на программный продукт. Состав и общий объем документов широко варьируются в зависимости от класса и характеристик объекта разработки, а также в зависимости от используемой технологии. Наиболее сложному случаю разработки ***критических ПС реального времени высокого качества соответствует самая широкая номенклатура документов.*** Такой перечень документов может быть использован ***как базовый*** для формирования на его основе состава и шаблонов документов в остальных более простых проектах.

Создание и применение ПС сложных систем сопровождается документированием этих объектов и процессов их жизненного цикла для обеспечения возможности освоения и развития функций программных средств и баз данных на любых этапах проекта ПС, а также для обеспечения интерфейса между разработчиками и с пользователями. По своему назначению и ориентации на определенные задачи и группы пользователей ***документацию ПС можно разделить на***:

— технологическую документацию процессов разработки и обеспечения всего жизненного цикла, включающую подробные технические описания и подготавливаемую для специалистов, ведущих проектирование, разработку и сопровождение комплексов программ, обеспечивающую возможность отчуждения, детального освоения, развития и корректировки ими программ и данных на всем жизненном цикле ПС;

— эксплуатационную документацию программного продукта — объекта и результатов разработки, создаваемую для конечных пользователей ПС и позволяющую им осваивать и квалифицированно применять эти средства для решения конкретных функциональных задач систем.

Технологическая документация непосредственно и в наибольшей степени должна отражать процессы жизненного цикла комплексов программ и данных и требования к этим документам. Стандарты и нормативные документы, входящие в жизненный цикл проекта ПС, должны регламентировать структуру, состав этапов, работ и документов ЖЦ ПС. Они должны: формализовать выполнение и документирование конкретных ра-

бот при проектировании, разработке и сопровождении ПС; обеспечивать адаптацию документов к характеристикам среды разработки, внешней и операционной системы; регламентировать процессы обеспечения качества ПС и его компонентов, методы и средства их достижения, реальные значения достигнутых показателей качества. Для контроля возможных изменений целесообразно предусматривать и согласовывать с заказчиком специальный документ, регламентирующий правила применения и корректировки номенклатуры, а также состава и содержания документации, поддерживающей ЖЦ ПС.

Эксплуатационная документация должна обеспечивать ***отчуждаемость программного продукта*** от первичных поставщиков — разработчиков и возможность освоения и эффективного применения комплексов программ достаточно квалифицированными специалистами — пользователями. Эксплуатационные документы должны исключать возможность некорректного использования ПС за пределами условий эксплуатации, при которых документами гарантируются требуемые показатели качества функционирования ПС. Основная ее задача состоит в фиксировании, полноценном использовании и обобщении результатов функционирования объектов и процессов всего жизненного цикла ПС и системы.

Базой эффективного управления проектом ПС и его документированием должен быть ***План***, в котором задачи исполнителей частных работ согласованы с выделяемыми для них ресурсами, а также между собой по результатам и срокам их достижения. План проекта должен отражать рациональное сочетание целей, стратегий действий, конкретных процедур, доступных ресурсов и других компонентов, необходимых для достижения основной цели с заданным качеством. Планирование реализации проектов и их документирования должно обеспечивать компромисс между характеристиками создаваемой системы и ресурсами, необходимыми на ее разработку и применение. Реализация плана зависит от результатов выполнения частных работ и документов и может требовать оперативной корректировки плана. Контроль обеспечивает исходные данные для координации элементов данной организации в соответствии с планом конкретной задачи. Для этого необходимо следить за ходом проекта и документирования на всем протяжении жизненного цикла и сравнивать запланированные и фактические результаты работ и документы. ***Контроль является органической функцией управления*** и должен иметь ряд средств ре-

гулирования поведения отдельных специалистов и коллектива разработчиков документов в целом.

При подготовке этих планов целесообразно по возможности разделять их цели и функции. План управления разработкой следует ориентировать на организацию специалистов, непосредственно создающих компоненты и ПС в целом, на эффективное распределение и использование ими ресурсов и средств автоматизации. В Плане управления документированием и обеспечением качества ПС внимание специалистов должно акцентироваться на анализе достигнутых результатов разработки, методах и средствах достижения заданных заказчиком характеристик ПС. ***При планировании и разработке комплекс документации должен проверяться и аттестовываться*** на полноту в условиях ограниченных ресурсов, на корректность, адекватность и непротиворечивость отдельных документов.

Различия в организационных службах и процедурах, методах и стратегиях приобретения ПС, масштабах и сложности проектов, требованиях систем и методах их разработки влияют на ***способы разработки, применения и сопровождения документов.*** Во многих предприятиях используются собственные нормативные шаблоны документов на процессы и продукты ЖЦ ПС, адаптированные к конкретным условиям разработки и характеристикам создаваемых ПС. В них выделяются состав и шаблоны документов, наиболее важные для проекта и качества создаваемого ПС, а также для конкретных заказчиков и пользователей. Соответственно определяются технологические и эксплуатационные документы, для которых регламентируются структура и основное содержание шаблонов документов.

Одна из важнейших задач документирования состоит в том, чтобы увязать четкими экономическими категориями взаимодействие разных специалистов в типовой производственной цепочке: заказчик — разработчик — изготовитель — пользователь документации. Для этого объект потребления — программный продукт, его документация и все процессы взаимодействия в цепочке должны быть связаны системой экономических и технических характеристик, в той или иной степени использующих основные экономические показатели — ***реальные затраты ресурсов: финансов, труда и времени специалистов на конечный программный продукт и документы***. Сложность документирования, количество и полнота содержания комплекса документов в первую очередь зависят от ***масштаба — размера проекта ПС***, что целесообразно оценивать в начале его ЖЦ. Для

решения этой задачи необходимо детально учитывать требуемые ресурсы современных процессов создания, документирования и использования программ различных классов и назначения — встроенных, коммерческих, административных, учебных, уникальных.

Особое внимание в последнее время уделяется ***совершенствованию и детализации документов, обеспечивающих высокое качество создаваемых ПС***, а также возможности их эффективного итерационного развития длительное время в многочисленных версиях. Соответственно должны изменяться документы, отражающие состояние процессов и компонентов проектов. Для этого организация процессов документирования должна обеспечивать гибкое и точное изменение документов — ***сопровождение и конфигурационное управление версиями и редакциями каждого документа***. Эти процессы и поддерживающие их средства автоматизации должны быть адекватными тем, которые применяются при сопровождении непосредственных объектов разработки — комплекса программ и данных. Они должны быть поддержаны организацией контроля, регистрации и утверждения версии каждого документа, в той степени и на том уровне, которые необходимы в данном проекте для определенного документа.

Для хранения, тиражирования и распространения документов, сложных ПС высокого качества следует выделять группу специалистов, ответственных за ***контроль, обеспечение и гарантированное сохранение документации***. Для критических, важных систем документация на программы и данные должна храниться и дублироваться на различных типах носителей и эпизодически выводиться на бумажные носители. При определении схемы обеспечения сохранности документации разного содержания, следует учитывать ее важность, трудоемкость хранения и возможность аварийного восстановления. Кроме того, должна быть организована ***служба нормативного контроля***, ответственная за соблюдение стандартов, нормативных и руководящих документов при подготовке документации всеми специалистами, участвующими в крупном проекте. Эта служба обязана обеспечить унификацию и высокое качество содержания, структуры и оформления шаблонов документов.

Для обеспечения достоверных данных об объектах и процессах управления документами ПС необходима автоматизированная ***база данных — информационная система обеспечения и хранения документов проекта*** (см. лекцию 16). Такая информационная система документооборота

представляет собой комплекс формальных и неформальных каналов поэтапного обмена информацией и документами между участниками проекта. Степень формализации документооборота в коллективе специалистов может варьироваться от утверждаемых руководителями планов, технических заданий и документов на компоненты и версии ПС до личных бесед между разработчиками.

17.2. Формирование требований к документации сложных программных средств

Масштаб проектов комплексов и компонентов программ является одним из важнейших факторов, влияющих на ***формирование структуры и содержание документации,*** поддерживающей весь жизненный цикл ПС. Оценки масштаба проекта ПС должны быть проанализированы и скорректированы для установления в договоре между заказчиками и разработчиками исходного компромиссного масштаба, допустимого для разработки первичных требований к комплексу документации. Некоторые требования могут потребовать изменения (обычно увеличения) масштаба и соответственно ресурсов на этапах предварительного и детального проектирования для обеспечения возможности реализации требуемой функциональной пригодности. Таким образом, требования к функциональной пригодности, к конструктивным характеристикам ПС, к ***форматам и структуре документации должны быть согласованы с масштабом проекта*** и доступными ресурсами для реализации на ранних этапах его жизненного цикла. Для этого необходимо использовать адекватные методы и единицы измерения масштаба проектов ПС (см. лекцию 5).

Формированию требований к комплексу программ должно сопутствовать создание требований, отражающих его документооборот, вследствие чего эти процессы во многом подобны. От масштаба ПС непосредственно зависят затраты ресурсов для их документирования и не всегда целесообразно создавать и использовать в реальных проектах весь комплекс документов, отраженных ниже в таблицах 17.1—17.7. Масштаб проекта и спецификация требований к ПС непосредственно отражаются на составе, содержании и объеме документации, необходимой различным участникам проекта. Каждый из разработчиков в той или иной степени должен при-

влекаться к управлению требованиями как к проекту, так и его документации. Разработчикам необходимо выработать профессиональные приемы для ***понимания и изложения в документах потребностей заказчиков и пользователей***, управления масштабом проекта, построением системы и документации, которые включают:

— команду разработчиков ПС — получает представление о сложности и размере создаваемого продукта и составе его документации;

— менеджеров проекта — базу для расчета содержания спецификаций, графиков, затрат и ресурсов;

— группы тестирования — планы тестирования, варианты испытаний и процедуры проверок;

— специалистов по сопровождению и поддержке — получают представление о функциональности каждой составной части продукта;

— клиентов отдела маркетинга и специалистов по продажам — должны иметь представление о конечном программном продукте;

— составителей документации, создающих шаблоны документов, руководства для пользователей и справки на основании спецификации требований к ПС — получат проект пользовательского интерфейса;

— специалистов, ответственных за обучение персонала, — получат спецификации требований к ПС и документацию для пользователей, а также для разработки обучающих материалов;

— персонал, занимающийся юридической стороной проекта — проверяет, соответствуют ли требования к продукту существующим законам и постановлениям.

Размер или масштаб комплексов программ в настоящее время приводится в публикациях в различных единицах, что может изменять их численные значения для одних и тех же программ в несколько раз (см. лекцию 5). Влияние единиц измерения размера программ на оценку рационального объема документации можно значительно изменять, если учесть их принципиальные особенности и, прежде всего, выделить две группы единиц измерения масштаба проектов ПС:

— группу, характеризующую размер исходных текстов программ, которые ***разрабатываются и анализируются специалистом — человеком,*** отражающую сложность, трудоемкость и длительность создания ПС, его компонентов и основных документов;

— группу, отражающую размер программ и данных, ***размещаемых в реализующей (объектной) ЭВМ*** и характеризующую объем памяти и производительность ЭВМ, необходимые для рабочего функционирования и исполнения комплекса программ в соответствии с его назначением.

Рис. 17.1

Эти две группы единиц отражают размер программ и документов с разных позиций и должны использоваться в зависимости от целей анализа и применения значений масштаба проекта (рис. 17.1). Для разработки ***набора документов*** после оценки масштаба комплекса программ необходи-

мо выполнить цикл поэтапного определения и формирования совокупности спецификаций требований к компонентам и документации проекта ПС. Первым этапом является создание ***Концепции проекта ПС и комплекса первичных требований спецификаций*** к иерархическому набору функций, на которые могут быть разбиты предполагаемые фактические компоненты ПС. В дальнейшем разбиение может структурироваться и детализироваться, формируя упрощенный или более точный уровень абстракции и взаимодействия компонентов. Цель документа — Концепция, состоит в сборе, анализе и определении высокоуровневых потребностей пользователей, функций и документов программного продукта. Основное внимание должно уделяться возможностям и функциям документов, в которых нуждаются будущие разработчики и пользователи.

После первичной оценки масштаба проекта ПС итерационно выполняется поэтапная разработка спецификаций требований проекта и документов ПС. Ограничения при прогнозировании требований к документам определяются, прежде всего, имеющимися данными, которые могут быть использованы в качестве исходных аналогов или обобщенных характеристик. Будучи конечным хранилищем комплекса требований к продукту и документов, спецификация требований к ПС должна быть ясной и понятной, дабы у разработчиков и заказчиков не оставалось ни малейших возможностей для разночтения. Не обязательно составлять спецификацию на документацию для всего продукта до начала разработки, но необходимо зафиксировать требования для каждой составляющей перед ее созданием. При работе над любым проектом необходимо достичь согласованности решений по каждому набору требований и документов до начала их реализации разработчиками. Согласованность решений представляет собой процесс изучения и одобрения документов к ПС. В общем случае для оценки, прогнозирования и обоснования ***спецификаций требований нового комплекса документов необходимы исходные данные***:

— обобщенные характеристики использованных ресурсов для документирования и технико-экономические показатели завершенных разработок — прототипов ПС, а также оценки влияния на них функций, различных факторов объектов и среды разработки;

— реализованные планы документирования и обобщенные перечни выполненных работ, реальные графики проведенных ранее оценок и разработок, различных ПС и документов;

— цели и содержание частных работ в процессе создания предыдущих сложных комплексов программ и набора различных документов для обеспечения необходимого качества ПС в целом;

— структура и содержание полного комплекта документов, являвшегося результатом выполнения отдельных работ конкретного проекта.

Составлять спецификацию требований к документации ПС следует таким образом, чтобы все заинтересованные в проекте лица смогли в ней разобраться и применять:

— разделы, подразделы и отдельные требования должны быть названы согласованно;

— полезно использовать средства визуального выделения (такие, как полужирное начертание, подчеркивание, курсив и различные шрифты) последовательно, иерархически и в разумных пределах;

— создавать оглавление, а также алфавитный указатель, чтобы облегчить пользователям поиск необходимой информации;

— нумеровать все рисунки и таблицы, ссылки на них, используя присвоенные номера.

Для устранения или снижения ошибок состава, содержания и рисков документации до допустимых пределов может быть необходимо изменение требований к функциональной пригодности и/или к конструктивным характеристикам и доступным ресурсам проекта ПС. Поэтому на этапах проектирования ***последовательно должны определяться***:

— при проектировании концепции и первичной оценке масштаба проекта — предварительные требования к назначению, функциональной пригодности, к составу и номенклатуре необходимых конструктивных характеристик качества и к первичному ***перечню набора документов ПС***;

— при предварительном проектировании — уточненная оценка масштаба проекта, требования к функциональным и конструктивным характеристикам качества, к ориентировочной структуре и содержанию ***набора шаблонов документов*** в жизненном цикле ПС с учетом общих ограничений ресурсов;

— при детальном проектировании — подробные спецификации требований к функциональным и конструктивным характеристикам качества с детальным учетом и распределением реальных ограниченных ресурсов, а также полный ***состав и содержание шаблонов документов*** в жизненном цикле ПС.

Чем крупнее и сложнее проект ПС и соответственно выше его стоимость, тем тщательнее следует разрабатывать требования к его характеристикам, к составу и содержанию документов, а также распределять ресурсы на их реализацию. Поэтому при средней и относительно невысокой сложности ПС во многих случаях можно удовлетвориться подготовкой требований к комплексу программ с подробностью анализа, соответствующего предварительному проектированию. Для крупных и особо сложных проектов необходим более детальный анализ факторов при разработке набора документов.

В зависимости от сложности проекта окончательным результатом работ должны быть детализированные и утвержденные документы спецификаций к номенклатуре, свойствам, значениям качества и документации проекта ПС, которые достаточны для его полноценного рабочего проектирования и последующей эффективной эксплуатации. Эти ***требования к документам закрепляются в контракте и техническом задании***, по которым разработчик впоследствии должен отчитываться перед заказчиком при завершении проекта (см. рис. 17.1). Однако на последующих этапах жизненного цикла и при конфигурационном управлении документы могут изменяться по согласованию между заказчиком и разработчиком, которые чаще всего приурочиваются к подготовке новой версии программного продукта. Для этого необходим мониторинг масштаба проекта, комплекта документов, спецификаций требований и реализаций характеристик в течение всего ЖЦ ПС.

Требования к функциональным характеристикам, качеству, составу и содержанию документов, утвержденные после предварительного проектирования, могут быть закреплены в техническом задании ***как обязательные для детального и рабочего проектирования***. Эти данные могут использоваться при последующем оценивании качества документации при их сопоставлении с требованиями в процессе квалификационных испытаний или сертификации программного продукта. Однако для крупномасштабных и дорогих проектов может потребоваться уточнение требований к качеству документации при детальном проектировании с позиции улучшения соотношения значений качества и затрат ресурсов, которые необходимы или допустимы для их реализации в ЖЦ ПС.

Для заказчика и пользователей может иметь значение не только определение функциональной пригодности, но и оценка потенциального спро-

са на рынке конкретного программного продукта, а также его ***конкурентоспособности*** с другими аналогичными по функциям ПС с учетом его качества и стоимости. Это обстоятельство может определять необходимость уточнения требований к отдельным характеристикам и документам не только для их реализации разработчиками в ЖЦ ПС, но также для оценивания интегрального качества и документации готового программного продукта, поставляемого на рынок.

Обычно заказчики и разработчики первоначально устанавливают требования к каждой характеристике и к номенклатуре документов ПС без учета относительных затрат на их достижение, а также ***без детального анализа их совместного влияния на полную функциональную пригодность у потребителей***. Это может приводить к значительным перекосам и к несбалансированным значениям требований к отдельным взаимосвязанным характеристикам и документам, на которые нерационально используются ограниченные ресурсы ЖЦ ПС, или к неадекватно низким их значениям. В проектах крупных ПС это может угрожать значительным повышением стоимости и/или снижением конкурентоспособности создаваемого программного продукта из-за недостаточного уровня отдельных показателей качества и документов.

Атрибуты качества ПС и полезность документов имеют различные меры, вследствие чего они в большинстве своем непосредственно ***несопоставимы между собой***. Они предварительно выбираются и согласовываются с заказчиком при последовательном, почти независимом анализе каждого документа, для использования в контракте и техническом задании. Для обобщенного оценивания качества ПС необходим учет относительного влияния каждой конструктивной характеристики и документа на функциональную пригодность. При этом не всегда учитываются ресурсы для их реализации в конкретном ПС. Это часто приводит к нерациональным требованиям и документам, которые значительно отличаются: либо по степени влияния на функциональную пригодность, либо по величине ресурсов, необходимых для их реализации как полноценных документов.

Для эффективного управления документацией сложного ПС при детальном проектировании целесообразно учитывать и обобщать степень полезности разнородных характеристик конкретных документов в некоторый интегральный показатель, отражающий их совокупное влияние на его

функциональную пригодность. Таким образом, при разработке требований к характеристикам документов проекта выявилась проблема анализа системной эффективности документации и обобщения их характеристик, а также оценивания совместного влияния различных документов на функциональную пригодность ПС с учетом затрат на их реализацию.

Для управления и сопоставительного оценивания выбранных характеристик качества документов целесообразно каждому из них присваивать ***коэффициент или приоритет*** влияния на функциональную пригодность. Аналогично, экспертам целесообразно оценивать относительные ресурсы, которые следует затрачивать на реализацию каждого документа. Для каждого вида документов отношение коэффициента влияния на функциональную пригодность к относительным затратам на его достижение можно рассматривать как ***обобщенный уровень приоритета реализации этого документа*** (см. рис. 17.1). Для конкретного проекта ПС состав и значения приоритетов документов следует поэтапно адаптировать и уточнять с учетом их назначения и функций. Если затраты на разработку документации ПС можно оценивать и прогнозировать с некоторой достоверностью, то эффективность применения и особенно будущий спрос на конкретные документы комплекса программ со стороны различных пользователей априори оценить трудно. Такие оценки могут проводиться на основе специальных маркетинговых исследований и опыта эксплуатации аналогичных комплексов программ или достаточно близких их прототипов.

17.3. Планирование документирования проектов сложных программных средств

Общее руководство процессом документирования комплексов программ можно разделить на ***два уровня***:

— адаптация состава и содержания документов к данной деловой, проблемно-ориентированной области, например, авиационной, медицинской, военной, финансовой или административной;

— адаптация номенклатуры, структуры и содержания документов для каждого специфического проекта, контракта или предприятия.

В соответствии со стандартами план документирования в виде совокупности руководящих, промежуточных и отчетных документов должен

разрабатываться системными аналитиками и утверждаться менеджером проекта вместе со спецификацией требований к ПС. В спецификации формализуются требования к результатам документирования, а в плане — методы и средства их достижения. Тем самым характеристики ПС не только декларируются в виде требований, но и сопровождаются совокупностью рекомендуемых мероприятий и документов по их обеспечению и реализации.

Достоверность прогнозов требующихся ресурсов для документирования зависит, прежде всего, от ***точности оценки исходных данных***. Такие оценки зависят от компетенции и объективности экспертов, их ***оптимистичности, пессимистичности*** и знания существенных особенностей проекта. Оценивая масштаб, функции и требования к документации, заказчик и разработчики должны представлять тот объем затрат и физический размер комплекса документации, который придется создать в процессе всего жизненного цикла ПС, а также для обеспечения его эффективного применения.

Может представлять интерес оценка ***ориентировочного физического объема документации*** (например, в стандартных страницах А4 или эквивалентных объемов файлов) для проектов комплексов программ. В качестве ***гипотетического примера*** выделим ***два масштаба проектов***: малый — 50 тысяч строк и крупный — один миллион строк, и выделим оценки на ***технологическую и на эксплуатационную документацию***. В эксплуатационной документации обычно не оформляются и не приводятся спецификации компонентов, тексты программ с комментариями, тесты и результаты тестирования, что резко сокращает номенклатуру документов до трех — семи видов (см. таблицу 17.7). Каждое описание, руководство или инструкция может содержать до 100 страниц текста, что в совокупности дает до тысячи страниц эксплуатационных документов. Для крупного программного продукта несколько возрастает номенклатура документов, но главное, пропорционально увеличению сложности и масштаба комплекса программ до 10^6 строк объем эксплуатационной документации может увеличиться до 10 тысяч страниц.

Номенклатура технологических документов в жизненном цикле крупного ПС может доходить до 50 видов (см. таблицы 17.1—17.6) , среди которых наибольшее влияние на объем документации оказывают: специ-

фикации программ и данных, тексты программ с комментариями, тестовые сценарии и результаты тестирования компонентов и модулей. Для отражения совокупности этих документов в среднем на каждую строку текста программы может требоваться от 10% до полной страницы документации. Остальные документы в основном являются интегральными для проекта и вряд ли займут более 10% общего объема от перечисленных категорий документов. Таким образом, технологическая документация для жизненного цикла ПС размером 10^6 строк может составить около ста тысяч страниц или ста томов по тысяче страниц.

Вряд ли целесообразно изготавливать такой объем твердых копий документов на бумаге. Большая часть этих документов ***может оставаться в файлах*** (сотни мегабайт), однако каждый документ должен быть оформлен в соответствии со стандартами и шаблонами и скреплен подписями разработчиков и, где нужно, заказчика. Изменение этих документов должно санкционироваться, так же как твердых копий. Для относительно малого проекта ПС (50 тысяч строк) с минимальной технологической документацией может потребоваться около 5 тысяч страниц или эквивалентных файлов.

Приведенные оценки следует рассматривать ***только как ориентиры***, которые в реальных проектах могут изменяться на порядок в ту или иную сторону, в зависимости от требований заказчика и характеристик проекта. Оценки объема предстоящей разработки технологической и эксплуатационной документации целесообразно проводить на этапе детального проектирования с учетом реальных характеристик проекта ПС, что позволит избежать неприятных сюрпризов, вызванных превышением затрат на реализацию документов.

Менеджер проекта для оценок объема и содержания документации должен подготовить ***план выполнения документирования*** в жизненном цикле ПС (рис. 17.2). Этот план должен содержать описания соответствующих работ и задач и обозначения создаваемых программных продуктов и документов. Он должен охватывать следующие ***задачи***:

— установление графиков и сроков своевременного решения задач документирования;

— оценку необходимых трудозатрат на создание каждого документа и всего комплекса;

— определение времени, необходимого для выполнения конкретных задач документирования;

— распределение задач документирования по исполнителям;

— определение обязанностей исполнителей по созданию содержания документов;

— выделение критических ситуаций, связанных с задачами или самим процессом документирования;

— установление критериев управления и обеспечение качества документов;

— обеспечение внешних условий и определение инфраструктуры проекта системы для выполнения процесса документирования.

Рис. 17.2

В проекте ПС должен быть определен ***базовый график*** выполнения работ в жизненном цикле, а графики создания отдельных документов долж-

ны быть связаны и согласованы с этим базовым графиком. Менеджер каждого программного проекта должен стремиться по возможности использовать существующую организационную инфраструктуру документирования на предприятии. Должен быть определен механизм для разрешения или преодоления конфликтных ситуаций между менеджером всего проекта ПС и администратором процесса документирования на соответствующем уровне их полномочий по организационному управлению. Это положение является общим для всех контрольных этапов, связанных с выполнением договорных обязательств по вспомогательным процессам, поэтому необходимы синхронизация соответствующих планов и своевременное уведомление менеджера ПС о всех затруднениях, возникающих при выполнении соответствующих задач процессов документирования.

В интересах сокращения стоимости и улучшения качества стандарты и регламентируемый ЖЦ ПС рекомендуется адаптировать к характеристикам конкретного проекта. Соответственно сформированному жизненному циклу следует адаптировать состав документов конкретного проекта ПС. Для адаптации состава и содержания документации должны быть определены характеристики окружения проекта, которые могут воздействовать на адаптацию документирования: процессы жизненного цикла создаваемой системы; требования к системе и программному средству; организационные процедуры и стратегии документирования; размер, критичность и функции основных компонентов системы; количество задействованного в проекте персонала и сторон.

В ***плане управления документированием*** каждого этапа жизненного цикла ПС целесообразно фиксировать и документально оформлять:

— исходные данные (шаблоны), требующиеся для успешного выполнения данного этапа документирования проекта или компонента ПС;

— контролируемые и документируемые данные о состоянии объекта и процесса разработки, регистрируемые после завершения этапа;

— содержание процедур контроля состояния проекта и документов в процессе выполнения работ этапа;

— критерии оценки результатов выполненных работ и качества отчетных документов при завершении этапа;

— состав и содержание отчетных документов (шаблонов), представляемых для оценки состояния проекта, результатов завершенного этапа и

работ и для использования на следующем этапе или при завершении проекта ПС.

Менеджер программного проекта должен отвечать за проведение оценок программных продуктов, документов и планов: на соответствие принципам, методологии и технологии управления проектом и документированием; в части удовлетворения их установленным требованиям договора с заказчиком. Необходимыми компонентами успешной реализации программных проектов являются периодические анализы и оценки хода выполнения и завершения этапов и заданий на документирование.

Планирование качества документов в ряде стандартов принято отделять от планов непосредственного управления процессом создания комплекса программ. Для реализации планов качественного документирования должны быть созданы регламентирующие документы, охватывающие:

— процессы создания документов, отражающих качество программного продукта;

— обязанности и ответственность специалистов за качество конкретных документов;

— используемые ресурсы, обеспечивающие создание документов высокого качества;

— требования к качеству конкретных документов и способы его контроля.

Реальные ограничения ресурсов, используемых в процессе разработки, квалификация специалистов, изменения внешней среды и требований заказчика объективно приводят к отклонениям реализации плана документирования от предполагавшегося. Величина таких отклонений в значительной степени зависит от принятой технологии разработки, от уровня и характеристик средств разработки, а также от средств автоматизации создания программ. Для своевременного обнаружения отклонений от плана документирования необходимо регулярно регистрировать результаты выполненных работ и их характеристики качества. Для реализации таких измерений целесообразно предусмотреть и согласовать с заказчиком специальный документ, регламентирующий правила корректировки плана обеспечения качества ПС, а также состава и содержания поддерживающей его документации.

Адаптация номенклатуры и содержания документов ПС к особенностям системы и пользователей может базироваться на выборе

подходящих шаблонов из набора документов. В процессе адаптации состава и содержания документов должны быть учтены особенности пользователей, поддерживающего персонала, руководителей контракта, потенциальных покупателей. Процессы, работы, шаблоны и задачи ЖЦ должны селектироваться и включать в себя перечень документов, которые ***обязательно нужно разработать, и информацию о персональной ответственности за них***. Выбранные процессы, работы и задачи, не обеспечиваемые конкретными документами, следует оговаривать в самом контракте и утвержденном ЖЦ ПС. При адаптации документирования ПС необходимо также учитывать особенности проекта: стоимость, планирование, производительность специалистов, размеры проекта и интерфейс с человеком-пользователем. Все исходные данные и решения по адаптации номенклатуры, структуры и содержания документов должны быть документированы и утверждены руководством проекта вместе с обоснованием их целесообразности.

Для реализации на практике требований и планов документирования в жизненном цикле программных средств необходимы ***организационные мероприятия***, гарантирующие участникам проектов ***определенную культуру, дисциплину разработки и применения документов***. Такая организационная система должна обеспечивать специалистам разной квалификации и специализации возможность взаимодействия при решении требуемых комплексных задач для накопления, хранения и обмена упорядоченной информацией о состоянии и изменениях компонентов проекта. Формализованная в документах информация должна повышать ответственность специалистов за корректность ее содержания, оперативность формирования и изменения, качество процессов трансформации и реализации в жизненном цикле ПС.

Для этого в каждом проекте комплекса программ должен быть организован ***регламентированный процесс документооборота,*** обеспечивающий для коллектива специалистов единую среду разработки, изменения и утверждения документов, адекватных реальному содержанию объектного кода программ и текстовых данных файлов ПС. Процесс организации и технологического обеспечения документооборота должен быть ориентирован на слаженную, коллективную работу различных профессионалов, объединенных единой целью создания требуемого заказчиком комплекса

программ с заданными функциями и высоким качеством документации. Каждый участник проекта в соответствии со своими функциональными обязанностями должен иметь доступ к необходимой для него корректной информации и ограничен возможностями обращения к несанкционированным для него данным (см. табл. 16.1). Должны быть упорядочены деловые коммуникации между специалистами разных категорий, управление динамическими процессами изменений и транспортировки документов между подсистемами документооборота.

Первоначально должен быть разработан ***проект архитектуры системы документооборота и руководство по ее применению***, настроена выбранная СУБД на управление взаимодействующими подсистемами документооборота, с соответствующими комплектами выбранных шаблонов документов, с учетом класса и масштаба предполагаемого ЖЦ ПС (см. лекцию 16, п. 16.3). Шаблоны подсистем документооборота должны ***поэтапно заполняться реальными данными проекта*** от заказчика и разработчиков соответствующих квалификаций и контролироваться менеджерами проекта. При этом следует управлять динамикой процессов реализации процедур документооборота, регистрировать реальное использование ресурсов специалистов, текущее время выполнения процедур процессов ЖЦ ПС и оформления документов в подсистемах. В реальных проектах ПС возможны ***отклонения от такой линейной схемы*** двух типов:

— прерывание процессов документооборота и прекращения всей разработки после промежуточных этапов системного, предварительного или детального проектирования вследствие недостаточности ресурсов для его полной реализации или вследствие отказа заказчика продолжать данный проект при появлении альтернативного варианта программного продукта;

— итерационный возврат на предшествующие подсистемы документооборота, а также на их компоненты и шаблоны для корректировки и уточнения, обусловленные изменениями компонентов ПС.

При наличии адекватных прототипов и ряда детальных спецификаций требований, для создания новой базовой версии программного продукта может отсутствовать необходимость ее системного, предварительного или детального проектирования, а разработка и тестирование новых программных компонентов выполняться в процессе расширения функций предшествовавшей базовой версии. Тем самым процессы и руководства

документооборота при развитии и совершенствовании версий ПС могут формализоваться, начиная с некоторых промежуточных этапов жизненного цикла комплекса программ.

В состав базовых версий программного продукта входит ***седьмая подсистема документооборота и комплект документов пользователей*** (см. таблицу 17.7). Этот комплект и содержание документов также следует адаптировать в соответствии с требованиями и характеристиками проекта. В системах реального времени в ряде случаев могут отсутствовать документы административного управления, а также сокращены в шаблонах требования и функции оперативных пользователей. В комплексах программ административных систем может доминировать документооборот администраторов, действующих совместно с оперативными пользователями при реализации основных функций программного продукта. В некоторых автоматизированных системах реального времени программный продукт является органическим компонентом системы управления и внешней среды, и эксплуатационная документация должна отражать в основном контрольные функции, инсталляцию и оценки целостности функционирования программного продукта в системе.

На базе стандартов, представленных в Приложении 1, с учетом ряда публикаций и нормативных документов, созданы и представлены ниже семь таблиц, отражающих состав ***комплекта технологических и эксплуатационных документов жизненного цикла базовой версии сложного программного средства реального времени, создаваемого «с нуля»***. По ряду работ (например, программирование компонентов) в ЖЦ ПС рекомендуется использовать специфические документы, детально регламентирующие содержание и применение определенных технологических процедур в соответствии с инструкциями использования инструментальных средств. Детальную структуру каждого документа целесообразно уточнять менеджерам предприятия или проекта, в соответствии с их традициями, используемой технологией и особенностями проектируемого программного продукта. Формализованная структура и типовое содержание каждого документа должны позволять контролировать результаты и качество выполненных работ.

Представленный ***в таблицах 17.1—17.7 перечень документов*** в реальных проектах может варьироваться в зависимости от класса, масштаба

и характеристик ЖЦ программного средства. Наиболее сложному случаю разработки крупномасштабных критических ПС реального времени высокого качества соответствует номенклатура и детализация всего комплекта представленных документов. Для каждого этапа жизненного цикла выделены три уровня сложности проектов ПС: особо сложные (свыше 200 тысяч строк — крупные), средние по масштабу (свыше 50 тысяч строк — средние) и небольшие проекты программных средств (малые). ***Рекомендуемые для таких проектов документы*** в таблицах отмечены знаком +. Некоторые документы целесообразно применять факультативно, или с сокращением и упрощением содержания, что отмечено знаками + –. Выделенные три перечня документов целесообразно использовать как типовые, базовые, при адаптации и формировании состава и содержания документов в конкретных проектах, с учетом их особенностей, масштаба и характеристик.

Таблица 17.1

Документы предварительных требований, спецификаций и ресурсов для разработки программного средства

№	Документы	Масштаб проекта		
		Крупный	Средний	Малый
1.1	Интервью заказчиков и пользователей о проблемах и целях создания программного продукта	+	+	
1.2	Результаты обследования и описание системы и целей разработки комплекса программ	+	+	+ –
1.3	Технико-экономическое обоснование проекта программного средства	+	+	+ –
1.4	Концепция и основные предложения по созданию программного средства	+	+	
1.5	Предварительный укрупненный план проектирования и разработки версии программного средства	+	+ –	
1.6	Системный проект, общее описание программного средства и среды разработки для согласования между заказчиком и разработчиком	+	+	
1.7	Техническое задание на предварительное (детальное) проектирование программного средства	+	+	+

Таблица 17.2

Документы проектирования и выбора характеристик качества программного средства

№	Документы	Масштаб проекта		
		Крупный	Средний	Малый
2.1	Стандарты и ограничения на процессы проектирования программного средства	+	+	
2.2	Спецификация требований к системе и к комплексу программ	+	+	
2.3	Предварительное описание и контроль согласованности требований компонентов программного средства	+	+–	+–
2.4	Описание функционирования программного средства, взаимодействия с объектами внешней среды и человеко-машинного диалога	+	+–	+–
2.5	Описания алгоритмов компонентов (модулей) программного средства	+	+–	
2.6	Описание информационного обеспечения программного средства и системы управления базами данных	+	+	+–
2.7	Требования к характеристикам качества программного средства	+	+–	+–
2.8	Пояснительная записка к предварительному или детальному проекту программного средства	+	+	+
2.9	Описание концепции технологии автоматизированного проектирования программного средства	+	+–	
2.10	План и поддерживающее его Руководство по документированию жизненного цикла программного средства	+	+–	+–
2.11	Ведомость предварительного или детального проекта программного средства	+	+–	

Таблица 17.3

Документы процессов разработки и программирования компонентов программных средств

№	Документы	Масштаб проекта		
		Крупный	Средний	Малый
3.1	План разработки компонентов программного средства	+	+	+
3.2	План обеспечения качества компонентов программного средства	+	+–	
3.3	Стандарты кодирования компонентов программного средства	+	+	+–
3.4	Руководство по программированию компонентов комплекса программ	+	+	+
3.5	Документация на разработанный функциональный программный компонент или модуль программного средства	+	+	+

Таблица 17.4

Документы верификации и тестирования компонентов программных средств

№	Документы	Масштаб проекта		
		Крупный	Средний	Малый
4.1	Состав базовых документов, регламентирующих верификацию и тестирование программных компонентов	+	+–	
4.2	Исходные данные для верификации программных компонентов	+	+–	
4.3	Результаты верификации корректности взаимодействия компонентов в составе программного средства	+	+–	+–
4.4	Исходные данные для тестирования компонентов	+	+	+
4.5	Организация, подготовка тестирования и обеспечение качества компонентов	+	+–	+–
4.6	Сценарии тестирования и спецификации тестов для каждого компонента	+	+	+
4.7	План тестирования программного компонента	+	+	+
4.8	Отчет о результатах верификации и тестирования компонентов	+	+	+

Окончание табл. 17.4

№	Документы	Масштаб проекта		
		Крупный	Средний	Малый
4.9	Методика комплексирования функциональных компонентов	+	+–	
4.10	Оценка реализации комплексирования функциональных компонентов комплексов программ	+	+–	

Таблица 17.5

Документы квалификационного тестирования, испытаний и оценивания качества программных продуктов

№	Документы	Масштаб проекта		
		Крупный	Средний	Малый
5.1	Методика генерации тестов, имитирующих внешнюю среду и обработку результатов квалификационного тестирования	+	+–	+–
5.2	Методика применения проблемно-ориентированной системы квалификационного тестирования и испытаний комплексов программ	+–	+–	
5.3	Методика, содержание и сценарии квалификационного тестирования и испытаний программных средств	+	+	+–
5.4	Программа испытаний комплекса программ	+	+	+
5.5	Методики проведения испытаний комплекса программ по отдельным характеристикам качества	+	+	+
5.6	Протоколы по результатам испытаний функциональных компонентов и/или комплекса программ	+	+	+–
5.7	Итоговый отчет результатов разработки программного продукта	+	+	+
5.8	Акт завершения работ по проекту программного продукта	+	+	+–
5.9	Акт приемки программного продукта в промышленную эксплуатацию	+	+–	+–

Таблица 17.6

Документы сопровождения и конфигурационного управления версиями программного продукта

№	Документы	Масштаб проекта		
		Крупный	Средний	Малый
6.1	Описание среды жизненного цикла и конфигурации программного средства	+	+	+–
6.2	План сопровождения версий программного продукта	+	+	+–
6.3	План управления конфигурацией программного продукта	+	+	+–
6.4	Программа управления конфигурацией программного продукта	+	+	+–
6.5	Отчеты пользователей о выявленных дефектах и предложениях по корректировке комплекса программ	+	+	+
6.6	Описание выявленных дефектов и предложений по совершенствованию функций версии программного продукта	+	+	+
6.7	Описание подготовленных и утвержденных корректировок и обобщенных характеристик новой базовой версии программного продукта	+	+	+
6.8	Извещение пользователям о выпуске новой версии программного продукта и/или о прекращении сопровождения предшествующей версии	+	+–	+–
6.9	Описание новой базовой версии программного продукта	+	+	+
6.10	План передачи и внедрения новой базовой версии программного продукта пользователям	+	+–	
6.11	Отчет о результатах эксплуатации снятой с сопровождения версии программного продукта и ее архивации	+	+–	
6.12	Отчет о результатах тиражирования базовых версий, конфигурациях и параметрах пользовательских версий программного продукта	+	+–	

Таблица 17.7

Документы процессов эксплуатации программных продуктов

№	Документы	Масштаб проекта		
		Крупный	Средний	Малый
7.1	Общее описание системы, в которой используется программный продукт	+	+	+ –
7.2	Общие требования к формированию пользовательской документации программных продуктов	+	+	+ –
7.3	Описание административного управления программными продуктами системы	+	+ –	+ –
7.4	Руководство системного администратора программного продукта	+	+ –	+ –
7.5	Общее описание руководства пользователей программного продукта	+	+ –	+ –
7.6	Руководство оперативного пользователя программного продукта	+	+	+
7.7	Инструкция по формированию и ведению информации базы данных	+	+ –	+ –
7.8	Паспорт на программный продукт	+	+	+
7.9	Пользовательская документация на коммерческие пакеты — закрытые коробки программных средств			
7.10	Руководство по подготовке документов и обучению специалистов применению программного продукта	+	+ –	+ –

ЛЕКЦИЯ 18

УДОСТОВЕРЕНИЕ КАЧЕСТВА И СЕРТИФИКАЦИЯ ПРОГРАММНЫХ ПРОДУКТОВ

18.1. Процессы сертификации в жизненном цикле программных средств

Основной целью сертификации программных средств и систем качества, обеспечивающих их жизненный цикл, является контроль и удостоверение качества технологий и продукции, гарантирование их высоких потребительских свойств. Задача состоит в повышении эффективности затрат в сфере создания и применения конечного программного продукта, а также улучшение объективности оценок его характеристик и конкурентоспособности. Формальной целью сертификации является подготовка и принятие решения о целесообразности выдачи заявителю сертификата соответствия с учетом следующих факторов:

— полноты, точности и достоверности исходного технического задания и спецификаций требований, представленных в документации на ПС, а также на технологию поддержки его жизненного цикла;

— достоверности и точности измерения и обобщения результатов сертификационных испытаний и получения адекватных показателей качества конечных программных продуктов и/или технологических процессов их создания;

— методологии и качества интерпретации данных об объекте испытаний и/или технологии с учетом достоверности оценок, квалификации и объективности испытателей, заказчиков и пользователей.

В международных стандартах ***сертификация соответствия*** определена как действие третьей — ***независимой стороны***, доказывающее,

что обеспечивается необходимая уверенность в том, что должным образом идентифицированная продукция, процесс или услуга соответствует конкретным стандартам и/или другим нормативным документам. В понятие «нормативные документы» включены документы, содержащие правила, общие принципы или характеристики, касающиеся различных видов деятельности или их результатов, стандарты, технические условия, инструкции и регламенты по применению конкретной продукции или технологии.

Результатом положительных испытаний является ***сертификат соответствия*** — документ, изданный по правилам Системы сертификации, удостоверяющий, что обеспечивается необходимая уверенность в том, что должным образом идентифицированная продукция, процесс или услуга соответствует конкретным стандартам и/или другим нормативным документам. Срок действия сертификата обычно ограничен либо по времени (например, 3 года), либо до проведения достаточно значительной модификации продукта или процесса. Сертификат вступает в действие с момента его регистрации в государственном реестре.

Специалисты третьей стороны имеют право на расширение условий испытаний в пределах требований нормативной документации, при которых должно обеспечиваться заданное качество и безопасность результатов применения ПС. При этом в качестве первой стороны в процессе сертификации выступают разработчики или поставщики ПС и их компонентов, а второй стороной являются заказчики, потребители или пользователи. Одна из этих двух сторон может выступать инициатором — ***заявителем*** на сертификационные испытания.

Для удостоверения качества конечного продукта — программных средств и их компонентов следует сертифицировать технологические процессы, обеспечивающие их жизненный цикл. Поэтому далее рассматриваются ***совместно*** задачи сертификации ***конечных объектов — программных продуктов***, а также ***технологий и систем качества***, обеспечивающих их создание и совершенствование. В ряде случаев сертификат на технологию и систему качества предприятия может удовлетворить потребителя и заменить в контракте его требования наличия сертификата на продукцию. При анализе и организации процессов сертификационных испытаний технологий и/или объектов ПС следует учитывать ***ряд базовых компонентов методологии сертификации***, подлежащих рассмотрению и утверждению перед испытаниями конкретного проекта:

— цели сертификации — правовые, экономические, формальные;

— исходные данные и документы, необходимые для проведения сертификации — стандарты, нормативные и эксплуатационные документы, их структура и содержание;

— характеристики и классификация объектов и/или процессов испытаний и сертификации, а также требуемые значения характеристик и атрибутов качества;

— ресурсы, необходимые для проведения испытаний, — финансовые, специалисты, аппаратурная оснащенность, нормативные и программно-инструментальные средства.

В зависимости от области применения системы, от назначения и класса ПС их сертификация может быть обязательной или добровольной. Первоначальные затраты на их проведение должны нести инициаторы испытаний: либо заказчик и конкретные потребители системы и программного продукта, либо ее разработчики и поставщики. Соответственно меняются экономические и юридические механизмы их взаимодействия, распределения ответственности за дефекты и дополнительная прибыль за повышение качества сертифицированной продукции или технологии. Распределение ответственности за ущерб у пользователей при использовании дефектной продукции, имеющей сертификат, рекомендуется устанавливать в договорах на ее поставку и на сертификационные испытания.

В исходных нормативных документах и требованиях должны быть сосредоточены ***все функциональные и эксплуатационные характеристики***, обеспечивающие заказчику и пользователям возможность корректного применения сертифицированного объекта и/или технологического процесса во всем многообразии его функций и характеристик качества. Для особенно важной продукции, например, программных продуктов по государственным заказам для оборонной техники, результаты положительной сертификации системы качества могут использоваться заказчиком как основание для выдачи ***лицензии на производство и поставку*** этой продукции. Такая лицензия дает преимущество соответствующему поставщику ПС при конкурсах на производство определенной продукции и на заключение контракта на ее поставку.

Сертификация систем качества предприятия или проекта проводится для оценки достоинств потенциального поставщика, при наличии

предложения от него об установлении договорных отношений с заказчиком, на проектирование или производство ПС. Кроме того, в рамках договорных отношений она проводится, чтобы установить, что система качества поставщика соответствует установленным требованиям и применяется полностью, а также для внутренней оценки поставщиком собственной системы качества предприятия по отношению к стандартам. Испытания для сертификации проводятся в проблемно-ориентированных, технически компетентных испытательных центрах или лабораториях, аккредитованных на право проведения тех испытаний, которые предусмотрены в ее нормативных документах. Такие проверки могут проводиться по графику или вследствие важных изменений системы качества предприятия, процессов ЖЦ и качества продукции, а также после проведения корректирующих действий ПС. Проведение сертификации систем качества предприятий обычно ***планируется и осуществляется для целей***:

— определения соответствия или несоответствия технологии и элементов системы качества установленным требованиям стандартов;

— определения эффективности применяемой системы качества предприятия с точки зрения соответствия поставленным целям по обеспечению качества продукции;

— выявления слабых мест в технологии и системе качества предприятия, в наибольшей степени отрицательно влияющих на качество продукции;

— обеспечения возможности проверяемому предприятию улучшить свою систему качества;

— предотвращения и сокращения рекламаций за недостаточное качество и/или дефектную продукцию.

Сертификационные испытания являются наиболее формализованным и регламентированным этапом тестирования, как ***объектов*** — программных продуктов, так и ***процессов*** их создания, поддерживаемым значительным числом стандартов и документов. При сертификации обычно ***руководствуются следующими основными исходными документами:***

— действующими международными, государственными и ведомственными стандартами на проектирование и испытания комплексов программ, на жизненный цикл ПС, системы обеспечения и характеристики их качества, а также на технологическую документацию;

— утвержденным заказчиком и согласованным с разработчиком техническим заданием и/или спецификацией требований, утвержденным комплектом эксплуатационной документации на ПС и его компоненты, а также на систему обеспечения их качества;

— Программой сертификационных испытаний по всем требованиям технического задания и положениям эксплуатационной документации;

— методиками испытаний по каждому разделу требований технического задания и документации.

Подготовка регламентированной документации и такие испытания оправданы, когда необходимо длительное развитие и модификация крупного комплексов программ с гарантией малой вероятности проявления дефектов и ошибок. Таким образом, обычный процесс ЖЦ ПС дополняется соответствующей системой последовательных официальных проверок. При изменениях программ необходимо подтверждение имеющегося сертификата и проведение некоторого минимума проверок, удостоверяющих корректность выполненных модификаций. При этом используется система официальных уведомлений пользователей о проведенных изменениях, извещений об изменениях и дополнительных контрольных испытаниях, удостоверяющих их корректность.

Ресурсы для сертификации программных средств и систем качества предприятия должны выделяться в зависимости от характеристик испытываемого объекта или процесса. Определяющими ресурсами сертификации обычно являются: возможная трудоемкость и длительность испытаний, совокупная численность и структура коллектива специалистов-сертификаторов, а также их квалификация и подготовленность к коллективной проверке конкретного типа ПС и его компонентов или системы качества предприятия. Аппаратурная оснащенность испытателей конкретного программного продукта определяется, прежде всего, ресурсами и другими характеристиками ЭВМ, доступных для использования коллективу специалистов при сертификации.

18.2. Организация сертификации программных продуктов

Сертификация состоит из ряда организационных процессов, составляющих ***Систему сертификации***, которые поддерживаются регламентиро-

ванными процедурами и документами и должны выполняться квалифицированными, аттестованными экспертами — инспекторами. Для сертификации предприятия-разработчика и результатов его деятельности — программных продуктов ***моделями*** **CMMI** или ***профиля стандартов*** **ISO** рекомендуется определенная дисциплина, которая должна быть адаптирована к конкретным характеристикам объектов и внешней среды жизненного цикла ПС. Перечисленные ниже процессы и документы ориентированы на крупные проекты, и их состав может сокращаться по согласованию между разработчиками, заказчиками и сертификаторами в более простых случаях.

Работы по сертификации начинаются с аккредитации органа или испытательной лаборатории, формирования и представления в Центральный орган по сертификации заявки и комплекта документов для принятия решения о целесообразности аккредитации. При положительных результатах проверки оформляется и выдается аттестат аккредитации.

Положение об органе сертификации или лаборатории является основным документом, устанавливающим тематическую область аккредитации, юридический статус, функции, структуру, права и обязанности, методы, средства и организацию испытаний. Паспорт сертификационной лаборатории (центра) должен содержать сведения об оснащенности средствами вычислительной техники, необходимыми для проведения испытаний, о персонале и кадровом составе, оснащенности инструментальными средствами проведения испытаний, обеспечении нормативными, техническими и методическими документами, а также другими ресурсами, необходимыми для испытаний.

Руководство по качеству содержит изложение принципов, описание методов и процедур, связанных с выполнением основных функций и задач органа по сертификации или лаборатории, обеспечивающих качество проводимых испытаний и доверие к результатам оценок, испытаний и экспертиз. Руководство по качеству, как правило, включает разделы:

— политика в области обеспечения качества проведения испытаний и экспертиз;

— оснащение центра актуальными методологическими материалами и программно-инструментальными средствами испытаний;

— формализация требований к объектам испытаний;

— политика в области технической оснащенности центра и повышения квалификации персонала;

— архивация и контроль сохранности документации результатов сертификации.

Заявитель для оценивания продукции или процесса, подлежащих сертификации, направляет в орган по сертификации заявку по форме, принятой в Системе сертификации. Орган по сертификации проводит ***работу по подготовке и организации сертификации*** продукции по заявке. Эта работа включает в себя:

— выбор схемы сертификации с учетом специфики продукции (объем, технология, требования нормативных документов и др.) и предложений разработчика;

— определение количества и порядка отбора образцов и компонентов, подлежащих испытаниям, если это не указано в стандартах;

— выбор и определение аккредитованной испытательной лаборатории, которая должна проводить испытания;

— подготовку проекта договора на выполнение работ.

Подготовительная часть работы по сертификации заканчивается выпуском решения по форме, принятой в Системе сертификации. ***Решение вместе с проектами договора*** на выполнение работ направляется заявителю. При организации сертификационных испытаний осуществляется подбор и изучение действующих нормативных документов на продукцию, заявленную к сертификации, методов ее испытаний и оценки результатов.

Заявитель принимает окончательные решения, какие элементы системы качества, участки и виды организационной и технической деятельности подлежат проверке при сертификации в заданный интервал времени. Заявитель должен создать ***условия и представить документы для обеспечения процессов проверок***. Он может представить в орган по сертификации протоколы испытаний, проведенных при разработке и постановке продукции на производство, документы об испытаниях, выполненных сторонними испытательными лабораториями и другие документы, свидетельствующие о соответствии технологии или продукции установленным требованиям. На основе анализа представленных с заявкой документально подтвержденных доказательств соответствия его продукции установленным требованиям орган по сертификации может принять решение о сокращении объема испытаний или о выдаче сертификата.

Испытания проводятся испытательными лабораториями, аккредитованными на проведение ***только тех испытаний, которые предусмотрены в их нормативных, аккредитационных документах***. При невозможности проведения испытаний на испытательной базе аккредитованной испытательной лаборатории испытания могут проводиться персоналом аккредитованной испытательной лаборатории у изготовителя или потребителя данной продукции с использованием собственных средств испытательной лаборатории или имеющихся у поставщика средств испытаний.

Процесс сертификации программных продуктов и систем качества предприятия включает:

— анализ и выбор разработчиком или заказчиком (заявителем) компетентных в данной области органа и аттестованной лаборатории для выполнения сертификационных испытаний;

— подачу заявителем заявки на испытания в орган сертификации и принятие сертификаторами решения по заявке, выбор схемы сертификации, заключение договора на сертификацию;

— идентификацию требований к системе качества предприятия и/или к версии программного продукта, подлежащих испытаниям;

— выполнение сертификационных испытаний системы качества предприятия или версии программного продукта сертификационной лабораторией;

— анализ полученных результатов и принятие решения лабораторией и/или органом сертификации о возможности выдачи заявителю сертификата соответствия;

— выдачу органом сертификации заявителю — сертификата и лицензии на применение знака соответствия и на выпуск сертифицированной продукции — версий программного продукта;

— осуществление инспекционного контроля органом сертификации сертифицированной системы качества предприятия и/или продукции;

— проведение заявителем корректирующих мероприятий при нарушении соответствия процессов системы качества и/или продукции установленным требованиям и при неправильном применении знака соответствия.

При ***проверке ответственности руководства разработчика за качество продукции*** должно быть определено наличие у предприятия или проекта документально оформленных политики, целей и обязательств в

области качества, а также степень понимания этой политики, ее практическое осуществление и поддержание в рабочем состоянии на всех уровнях организации. Должно быть установлено наличие на предприятии представителя руководства, который независимо от других обязанностей имеет полномочия и несет ответственность за постоянное ***выполнение требований стандартов и нормативных документов системы качества***. Следует проверять наличие требований, процедур, средств и обученного персонала для практической реализации процессов системы качества, а также актуальность и систематичность оформления документации на все компоненты, требования и положения системы качества, представляющей собой интегрированный процесс на протяжении всего жизненного цикла ПС. ***Проверки системы качества должны включать определение***:

— наличия и полноты технологической документации и соблюдения ее требований на практике;

— состояния средств технологического оснащения и наличия системы их технического обслуживания;

— наличия и эффективности системы контроля и испытаний;

— состояния средств измерений и испытаний;

— наличие системы выявления и устранения выявленных недостатков продукции или технологии.

На основании испытаний оцениваются полученные результаты и обосновываются выводы о соответствии или несоответствии продукции или процессов требованиям нормативных документов. Протоколы испытаний представляются в орган по сертификации, а также заявителю по его требованию. Протоколы испытаний подлежат хранению в течение сроков, установленных в правилах систем сертификации продукции и в документах испытательной лаборатории, но не менее трех лет.

После получения и проверки комплектности и качества документации специалистами испытательной лаборатории следует провести ***экспертизу степени реального применения системы качества на предприятии***. Испытания начинаются с составления Программы проверки системы качества, которая должна служить рабочим планом проведения последующих работ. Программа является внутренним рабочим документом испытательной лаборатории и должна содержать перечень работ, детализируемый в соответствии со спецификой предприятия-разработчика и включающий в себя анализ полноты и качества, представленных исходных документов

и степени их практического применения при проектировании, разработке и поставке ПС. Экспертиза применения процедур системы качества осуществляется испытательной лабораторией на рабочих местах предприятия, обеспечивающего ЖЦ ПС. Проверки проводятся по наличию на рабочих местах специалистов — разработчиков соответствующих документов и по полноте использования их положений и рекомендаций. Анализы состояния проекта и внутренние проверки системы качества, процессов и/или продукции должны проводиться персоналом, независимым от лиц, непосредственно ответственных за выполнение этих работ.

Методики проверок качества разработки должны быть обеспечены необходимыми ресурсами для выполнения Программы испытаний, методиками планирования и разработки частных процедур проверок. Методики должны содержать: объекты и цели испытаний; оцениваемые показатели качества; условия и порядок испытаний; методы обработки, анализа и оценки результатов испытаний; техническое обеспечение испытаний и отчетность. Следует указывать технические и программные средства, используемые во время проведения испытаний, и порядок проведения испытаний, а также ожидаемые результаты проверок. Должны быть разработаны методики контроля за корректировками, действиями по исправлению дефектов, если в службу управления проверок поступит такой запрос. Служба управления Программами испытаний должна разработать методики сохранения конфиденциальности любой информации об испытаниях, а также данных, имеющихся у экспертов.

Протоколы испытаний представляются заявителю и в орган по сертификации. Заявитель может представить в орган по сертификации протоколы испытаний с учетом сроков их действия, проведенных при разработке и постановке продукции на производство, или документы об испытаниях, выполненных отечественными или зарубежными испытательными лабораториями, аккредитованными или признанными в Системе сертификации. На основании протоколов сертификационных испытаний оцениваются полученные результаты и обосновываются сделанные выводы о соответствии или несоответствии продукции требованиям нормативных документов.

Заключение по результатам сертификационных испытаний разрабатывается сертификаторами и содержит обобщенные сведения о результатах испытаний и обоснование целесообразности выдачи сертифика-

та. В случае получения отрицательных результатов сертификационных испытаний принимается решение об отказе в выдаче сертификата соответствия. После доработки сертифицируемой продукции или системы качества испытания могут быть повторены. Результаты анализа состояния технологии или качества продукции ***оформляются актом***, в котором даются оценки по всем позициям Программы испытаний и содержатся выводы, включающие общую оценку состояния производства и продукции, необходимость корректирующих мероприятий. Акт используется органом по сертификации наряду с протоколами испытаний, заявкой для выдачи и определения срока действия сертификата на программный продукт, периодичности инспекционного контроля, а также для составления корректирующих мероприятий.

По результатам сертификационных испытаний и экспертизы документации принимается решение о выдаче сертификата. В случае получения отрицательных результатов сертификационных испытаний принимается решение об ***отказе в выдаче сертификата*** соответствия. Кроме того, предприятию-заявителю могут быть направлены предложения по устранению предполагаемых причин отрицательных результатов испытаний, после доработки сертифицируемой продукции испытания могут быть повторены.

Орган по сертификации после анализа протоколов испытаний, оценки производства, сертификации системы качества, анализа документации, указанной в решении по заявке, осуществляет оценку соответствия продукции установленным требованиям, ***оформляет сертификат*** на основании заключения экспертов и регистрирует его. При внесении изменений в конструкторскую или эксплуатационную документацию, которые могут повлиять на качество системы или программный продукт, удостоверяемые при сертификации, заявитель должен известить об этом орган по сертификации для принятия решения о необходимости проведения дополнительных испытаний. После регистрации сертификат вступает в силу и направляется предприятию-заявителю. Одновременно с выдачей сертификата предприятию-заявителю может выдаваться ***лицензия*** на право применения знака соответствия.

За сертифицированными программными продуктами в процессе их эксплуатации в течение всего срока действия сертификата соответствия должен осуществляться ***инспекционный контроль***. Инспекционный кон-

троль проводится в форме периодических и внеплановых проверок соблюдения требований к качеству технологии и сертифицированной продукции. Объектами контроля, в зависимости от схемы сертификации, является сертифицированная продукция, система качества или стабильность производства предприятия-разработчика. При определении периодичности и объема инспекционной проверки учитываются следующие факторы: степень потенциальной опасности программного продукта, стабильность производства, объем выпуска, наличие и применение системы качества при разработке, информация о результатах испытаний продукта и его производства, проведенных изготовителем, органами государственного контроля и надзора.

Результаты инспекционного контроля ***оформляются актом***, в котором дается оценка результатов испытаний образцов и других проверок, делается общее заключение о состоянии производства сертифицированной продукции и возможности сохранения действия выданного сертификата. Акт хранится в органе по сертификации, а его копии направляются разработчику и в организации, принимавшие участие в инспекционном контроле. По результатам инспекционного контроля ***орган по сертификации может приостановить или отменить действие сертификата*** и аннулировать лицензию на право применения знака соответствия в случае несоответствия продукции требованиям нормативных документов, контролируемых при сертификации, а также в случаях:

— принципиальных изменений модели зрелости, профиля стандартов, нормативных документов на продукцию или метода испытаний;

— изменения конструкции (состава), комплектности продукции;

— изменения организации или технологии разработки и производства;

— невыполнения требований технологии, методов контроля и испытаний, системы качества, если перечисленные изменения могут вызвать несоответствие продукции требованиям, контролируемым при сертификации.

Решение о приостановлении действия сертификата и лицензии на право применения знака соответствия не принимается в том случае, если путем корректирующих мероприятий, согласованных с органом по сертификации, его выдавшим, заявитель может устранить обнаруженные причины несоответствия и подтвердить без повторных испытаний в аккредитованной лаборатории соответствие продукта или процессов нормативным

документам. Если этого сделать нельзя, то действие сертификата отменяется, и лицензия на право применения знака соответствия аннулируется. Информация о приостановлении или отмене действия сертификата доводится органом по сертификации, его выдавшим, до сведения заявителя, потребителей и других заинтересованных организаций. Действие сертификата и право маркирования продукции знаком соответствия могут быть возобновлены при выполнении предприятием-разработчиком следующих условий:

— выявления причин несоответствия и их устранения;

— представления в орган по сертификации отчета о проделанной работе по улучшению и обеспечению качества продукции;

— проведения по методикам и под контролем органа по сертификации дополнительных испытаний продукции и получения положительных результатов.

18.3. Документирование процессов и результатов сертификации программных продуктов

Состав и содержание документации для сертификации системы качества предприятия зависят от характеристик проектирования, разработки и модификации программных средств, а также от требований к их качеству и особенностей технологической среды. Поэтому необходимый комплект документов для каждого предприятия или проекта следует выбирать и адаптировать применительно к этим характеристикам. Оцениваемыми при сертификации показателями системы качества являются наличие соответствующих документов и практическое ***выполнение требований определенного уровня модели зрелости*** **CMMI** или ***адаптированного профиля стандартов на базе*** **ISO 9000:2000**, а также созданных на их основе должностных инструкций специалистами предприятия-разработчика. Заявитель должен подготовить и предъявить испытательной лаборатории согласованный между заказчиком и разработчиком и утвержденный комплект документов для проверки их достоверности, достаточности состава и качества изготовления в соответствии с нормативными документами.

Ориентировочный комплект основных документов при сертификации состоит из трех групп:

— ***базовые нормативные документы*** систем качества в соответствии с номенклатурой и содержанием ***профиля стандартов*** на базе **ISO 9000:2000** или ***модели зрелости*** **CMMI**, а также подготовленные разработчиками на их основе Программа, Руководство и инструкции, предъявляемые испытателям (экспертам) системы качества или продукции проверяемого предприятия;

— ***исходные документы***, характеризующие конкретное предприятие или проект, а также жизненный цикл программного средства, подготавливаемые руководством проекта для сертификации его качества;

— ***отчетные документы*** испытателей, отражающие результаты проверки (сертификации) системы качества предприятия и/или программного продукта, представляемые органу сертификации, заявителю и руководству проверяемого предприятия.

Предъявляемые на сертификацию программный продукт или система качества предприятия должны представляться в комплекте с соответствующей документацией. Перечень и приблизительное содержание групп этих документов ниже ***ориентированы на общий случай проверки систем качества предприятий***, обеспечивающих ***жизненный цикл крупных программных продуктов***. Комплект документов может сокращаться и адаптироваться по согласованию между заявителем, сертификатором и руководством проверяемого предприятия в соответствии с характеристиками проектов программных средств. Некоторые документы могут объединяться в интегрированные отчеты с четкой ответственностью определенных специалистов за их выполнение.

Базовые документы системы качества предприятия и жизненного цикла программного средства

1. Концепция, терминология, требования и Руководство по улучшению деятельности — Системы менеджмента качества — **ISO 9000:2000** или версия модели зрелости **CMMI**.

2. Адаптированные версии или перечень разделов и рекомендаций стандартов **ISO 12207**, **ISO 15504**, их **Изменений** и **Руководств по применению**, выделенных при адаптации и обязательных для использования в системе качества конкретного предприятия или проекта программного продукта.

3. Адаптированная версия или перечень разделов и рекомендаций стандарта **ISO 90003**, выделенных при адаптации и обязательных для применения в системе качества предприятия, выпускающего программный продукт.

4. Базовые характеристики и атрибуты качества проекта ПС, выделенные, адаптированные и конкретизированные на основе стандартов **ISO 12182**, **ISO 9126**, **ISO 14598**, **ISO 25000**.

5. Адаптированная версия и утвержденная редакция Руководства по сопровождению и конфигурационному управлению на основе рекомендаций стандартов **ISO 14764**, **ISO 10007**, **ISO 15846**.

6. Комплект должностных инструкций, определяющих ответственность, полномочия и порядок взаимодействия всего руководящего, выполняющего и проверяющего работу персонала, участвующего в процедурах системы качества предприятия для конкретного проекта ПС.

Исходные документы, отражающие особенности жизненного цикла конкретного программного средства

1. Описание характеристик программных продуктов, создаваемых на предприятии, системы и внешней среды их жизненного цикла, необходимых для адаптации и подготовки рабочих версий стандартов и требований проекта ПС и системы качества предприятия в соответствии с рекомендациями стандартов **ISO 12207, ISO 15504, ISO 90003** и **ISO 9126**.

2. Описание целей, требований и обязательств предприятия-разработчика в области системы качества, критериев качества процессов и продуктов разработки, поставки и поддержки всего жизненного цикла ПС.

3. Комплект эксплуатационных документов, поставляемых заказчику и пользователям для обеспечения ЖЦ и применения конкретной версии программного продукта на основе адаптированных стандартов **ISO 9294, ISO 15910, ISO 18019**.

4. Документация и средства автоматизации проектирования, разработки, модификации, контроля и испытаний, используемых для обеспечения жизненного цикла программного продукта.

5. Планы и методики испытаний применения и оценки эффективности процессов системы качества предприятия и программного продукта.

6. Методики сопровождения, идентификации компонентов программного продукта и документации, анализа и утверждения версий комплексов программ и данных.

7. Методика конфигурационного управления, утверждения, хранения, защиты, копирования версий программного продукта и сопровождающих документов, а также накопления и хранения зарегистрированных в архиве предприятия данных о характеристиках качества в течение жизненного цикла версий программного продукта.

Результирующие документы испытаний — сертификации системы качества предприятия и/или программного продукта

1. Отчет о наличии, актуальности и систематичности оформления документации, адаптированной к требованиям и положениям системы качества предприятия, обеспечивающей интегрированный процесс гарантии качества на протяжении всего жизненного цикла программного продукта.

2. Результаты контроля и испытаний состояния и применения системы качества, проводимых периодически для определения ее пригодности и эффективности.

3. Отчет о наличии и поддержании в рабочем состоянии методик проведения проверок и документально оформленных отчетов о результатах достигнутого качества выполнения требований договора на сертификацию с заказчиком.

4. Результаты регистрации достигнутых характеристик качества комплекса программ: идентификация, накопление, хранение зарегистрированных данных о характеристиках и атрибутах качества программного продукта и его компонентов.

5. Результаты реализация плана разработки, документально оформленных входных и выходных данных этапов разработки и протоколов проверки реализации жизненного цикла ПС.

6. Результаты практического выполнения Программы качества и осуществления регламентированной деятельности в области качества на всех этапах жизненного цикла ПС.

7. Результаты аттестации имитаторов внешней среды и генераторов тестов, а также оценка их достаточности для выполнения сертификационных испытаний программного продукта.

8. Результаты анализа выполнения планов и методик проведения испытаний, протоколы испытаний, оценки соответствия результатов испытаний предъявляемым требованиям, а также результаты испытаний, утвержденные представителями заявителя, заказчика и поставщика.

9. Акт результатов проверок реальных характеристик жизненного цикла ПС и системы качества предприятия, выводы об их соответствии требованиям к сертификации производства программного продукта.

10. Сертификат системы качества предприятия и/или программного продукта и обеспечения его жизненного цикла, лицензия на применение знаков соответствия.

ПРИЛОЖЕНИЯ

Приложение 1

ПЕРЕЧЕНЬ ОСНОВНЫХ СТАНДАРТОВ ПРОГРАММНОЙ ИНЖЕНЕРИИ

1. **CMMI — Capability Maturity Model Integration for Product and Process Development** — Интегрированная модель оценивания зрелости продуктов и процессов разработки программных средств.

2. **ISO 15288:2002.** Системная инженерия. Процессы жизненного цикла систем.

3. **ISO 19760:2003.** Системная инженерия. Руководство по применению стандарта ISO 15288.

4. **ISO 12207:1995.** (ГОСТ Р—1999). ИТ. Процессы жизненного цикла программных средств.

5. **ISO 12207:1995.** ИТ. Процессы жизненного цикла программных средств. **Изменения 1 и 2:2002-2004**.

6. **ISO 15271:1998.** (ГОСТ Р—2002). ИТ. Руководство по применению ISO 12207.

7. **ISO 16326:1999.** (ГОСТ Р—2002). ИТ. Руководство по применению ISO 12207 при административном управлении проектами.

8. **ISO 15504:1-9:1998.** ТО. Оценка и аттестация зрелости процессов жизненного цикла программных средств. Ч. 1. Основные понятия и вводное руководство. Ч. 2. Эталонная модель процессов и их зрелости. Ч. 3. Проведение аттестации. Ч. 4. Руководство по проведению аттестации. Ч. 5. Модель аттестации и руководство по показателям. Ч. 6. Руководство по компетентности аттестаторов. Ч. 7. Руководство по применению при усовершенствовании процессов. Ч. 8. Руководство по применению при определении зрелости процессов поставщика. Ч. 9. Словарь.

9. **ISO 15504:1-5:2003-2006**. ИТ. Процесс аттестации. Ч. 1. Концепция и словарь. Ч. 2. Выполнение аттестации. Ч. 3. Руководство по произ-

водству аттестации. Ч. 4. Руководство пользователей для процессов усовершенствования и определения зрелости процессов. Ч. 5. Образец модели процессов аттестации.

10. **ГОСТ Р 51904—2002**. Программное обеспечение встроенных систем. Общие требования к разработке и документированию.

11. **ISO 9000:2000.** (ГОСТ Р—2001). Система менеджмента (административного управления) качества. Основы и словарь.

12. **ISO 9001:2000.** (ГОСТ Р—2001). Система менеджмента (административного управления) качества. Требования.

13. **ISO 9004:2000.** (ГОСТ Р—2001). Система менеджмента (административного управления) качества. Руководство по улучшению деятельности.

14. **ISO 90003:2004.** Руководство по организации применения стандарта **ISO 9001:2000** для программных средств.

15. **ISO 10005:1995.** (ГОСТ). Административное управление качеством. Руководящие указания по программам качества.

16. **ISO 10006: 1997.** (ГОСТ). Руководство по качеству при управлении проектом.

17. **ISO 10007:1995.** (ГОСТ). Административное управление качеством. Руководящие указания при управлении конфигурацией.

18. **ISO 10011-1-3:1990.** (ГОСТ). Руководящие положения по проверке систем качества. Ч. 1. Проверка. Ч. 2. Квалификационные критерии для инспекторов-аудиторов систем качества. Ч. 3. Управление программами проверок.

19. **ISO 12182:1998.** (ГОСТ Р—2002). ИТ. Классификация программных средств.

20. **ISO 9126:1991.** (ГОСТ—1993). ИТ. Оценка программного продукта. Характеристики качества и руководство по их применению.

21. **ISO 14598-1-6:1998-2000.** Оценивание программного продукта. Ч. 1. Общий обзор. Ч. 2. Планирование и управление. Ч. 3. Процессы для разработчиков. Ч. 4. Процессы для покупателей. Ч. 5. Процессы для оценщиков. Ч. 6. Документирование и оценивание модулей.

22. **ISO 9126-1-4:2002**. ИТ. ТО. Качество программных средств: Ч. 1. Модель качества. Ч. 2. Внешние метрики. Ч. 3. Внутренние метрики. Ч. 4. Метрики качества в использовании.

23. **ISO 25000:2005.** ТО. Руководство для применения новой серии стандартов по качеству программных средств на базе обобщения стандартов ISO 9126:1-4:2002 и ISO 14598:1-6:1998-2000.

24. **ISO 15939:2002.** Процесс измерения программных средств.

25. **IEC 61508:1-6:1998-2000.** Функциональная безопасность электрических/электронных и программируемых электронных систем. Ч. 3. Требования к программному обеспечению. Ч. 6. Руководство по применению стандартов IEC 61508-2 и IEC 61508-3.

26. **ISO 15408-1-3.1999.** (ГОСТ Р—2002). Методы и средства обеспечения безопасности. Критерии оценки безопасности информационных технологий. Ч. 1. Введение и общая модель. Ч. 2. Защита функциональных требований. Ч. 3. Защита требований к качеству.

27. **ISO 13335-1-5.1996-1998.** ИТ. ТО. Руководство по управлению безопасностью. Ч. 1. Концепция и модели обеспечения безопасности информационных технологий. Ч. 2. Планирование и управление безопасностью информационных технологий. Ч. 3. Техника управления безопасностью ИТ. Ч. 4. Селекция (выбор) средств обеспечения безопасности. Ч. 5. Безопасность внешних связей.

28. **ISO 10181:1-7.BOC.1996-1998.** Структура работ по безопасности в открытых системах. Ч. 1. Обзор. Ч. 2. Структура работ по аутентификации. Ч. 3. Структура работ по управлению доступом. Ч. 4. Структура работ по безотказности. Ч. 5. Структура работ по конфиденциальности. Ч. 6. Структура работ по обеспечению целостности. Ч. 7. Структура работ по проведению аудита на безопасность.

29. **ISO 14252:1996** (IEEE 1003.0). Руководство по функциональной среде открытых систем POSIX.

30. **ISO 9945:1-4:2003.** ИТ. Интерфейсы переносимых операционных систем. Ч. 1. Базовые определения. Ч. 2. Системные интерфейсы. Ч. 3. Команды управления и сервисные программы. Ч. 4. Обоснование.

31. **ISO 13210:1994.** ИТ. Методы тестирования для измерения соответствия стандартам POSIX.

32. **ISO 14756:1999.** ИТ. Измерение и оценивание производительности программных средств компьютерных вычислительных систем.

33. **ISO 12119:1994.** (ГОСТ Р—2000). ИТ. Требования к качеству и тестирование.

34. **ANSI/IEEE 829-1983**. Документация при тестировании программ.

35. **ANSI/IEEE 1008-1986**. Тестирование программных модулей и компонентов ПС.

36. **ANSI/IEEE 1012-1986**. Планирование верификации и подтверждения достоверности качества (валидации) программных средств.

37. **ISO 14764:1999.** (ГОСТ Р—2002). ИТ. Сопровождение программных средств.

38. **ISO 15846:1998.** ТО. Процессы жизненного цикла программных средств. Конфигурационное управление программными средствами.

39. **ISO 16085:2004.** Характеристики процессов управление рисками при разработке, применении и сопровождении программных средств.

40. **ISO 6592:2000.** ОИ. Руководство по документации для вычислительных систем.

41. **ISO 9294:1990.** (ГОСТ—1993). ТО. ИТ. Руководство по управлению документированием программного обеспечения.

42. **ISO 9127:1990.** (ГОСТ—1993). ТО. ИТ. Руководство по управлению документированием программного обеспечения.

43. **ISO 15910:1999.** (ГОСТ Р—2002). ИТ. Пользовательская документация программных средств.

44. **ISO 18019:2004.** ИТ. Руководство по разработке пользовательской документации на прикладные программные средства для офисов, бизнеса и профессиональных применений.

45. **ISO 6592:2000.** ОИ. Руководство по документации для вычислительных систем.

46. **ISO 9294:1990.** (ГОСТ—1993). ТО. ИТ. Руководство по управлению документированием программного обеспечения.

47. **РД 50-34.698-90.** Методические указания. Информационная технология. Автоматизированные системы. Требования к содержанию документов.

48. **ГОСТ Р 51901-2002.** Управление надежностью. Анализ риска технологических систем.

49. **DO-178 B -1995.** Соглашение по сертификации бортовых систем и оборудования в части программного обеспечения.

50. **ISO 14102:1995.** Оценка и выбор CASE-средств.

51. **ISO 14471:1995.** Руководство по адаптации CASE-средств.

52. **ISO 14143: 1-5: 1998-2004.** ИТ. Измерение программных средств. Измерение функционального размера. Ч. 1. 1998. Определение концепции. Ч. 2. 2002. Оценивание соответствия методов измерения размера программных средств стандарту ISO 14143:1:1998. Ч. 3. 2003. Верификация методов измерения функционального размера. Ч. 4. 2002. Эталонная модель. Ч. 5. 2004. Определение функциональных доменов для использования при измерении функционального размера.

53. **ISO 20926:2003.** Руководство по практическому методу измерения функционального размера программных средств.

54. **ISO 20968:2002.** Руководство по расчетам на основе анализа функциональных точек — Марк II.

Приложение 2

ТЕМЫ СЕМИНАРСКИХ ЗАНЯТИЙ ПО КУРСУ «ПРОГРАММНАЯ ИНЖЕНЕРИЯ»

Семинар 1. Разработка состава и содержания документов системного проекта сложного комплекса программ.

Семинар 2. Разработка спецификации требований к проекту сложного комплекса программ.

Семинар 3. Разработка требований к характеристикам качества проекта сложного комплекса программ.

Семинар 4. Анализ и сравнение требований к характеристикам качества трех типов программных средств.

Семинар 5. Разработка документов технико-экономического обоснования проекта сложного программного средства на базе экспертных оценок модели COCOMO.

Семинар 6. Разработка проекта контракта с заказчиком на обеспечение жизненного цикла сложного комплекса программ.

Семинар 7. Разработка группы планов обеспечения жизненного цикла и распределения ресурсов проекта сложного комплекса программ.

Семинар 8. Анализ и оценка рисков при разработке сложного комплекса программ.

Семинар 9. Анализ и оценивание корректности программ по покрытию тестами их структуры.

Семинар 10. Разработка комплекта документов и структуры базы данных для управления конфигурацией проекта комплекса программ.

Семинар 11. Подготовка проекта комплекта эксплуатационных документов для конкретного, сложного программного продукта на основе стандартизированных шаблонов.

Семинар 12. Анализ и выбор инструментальных средств для обеспечения жизненного цикла сложного комплекса программ.

Примечание. Руководителю семинаров рекомендуется выбирать тип и основные характеристики анализируемого комплекса программ на базе личных профессиональных интересов, сферы деятельности и опыта соответствующих разработок.

ЛИТЕРАТУРА

1. Боэм Б.У. Инженерное проектирование программного обеспечения. Пер. с англ. / Под ред. А.А. Красилова. — М.: Радио и связь, 1985.

2. Брауде Э. Технология разработки программного обеспечения. Пер. с англ. — СПб.: Питер, 2004.

3. Вигерс К.И. Разработка требований к программному обеспечению. Пер. с англ. — М.: Русская редакция, 2004.

4. Гецци К., Джазайери М., Мандриоли Д. Основы инженерии программного обеспечения. Пер. с англ. — СПб.: БХВ-Петербург, 2005.

5. Леффингуэлл Д., Уидриг Д. Принципы работы с требованиями к программному обеспечению. Унифицированный подход. Пер. с англ. — М.: Вильямс, 2002.

6. Липаев В.В. Отладка сложных программ. — М.: Энергоатомиздат, 1993.

7. Липаев В.В. Системное проектирование сложных программных средств для информационных систем. Изд. второе, переработанное и дополненное. — М.: СИНТЕГ, 2002.

8. Липаев В.В. Методы обеспечение качества крупномасштабных программных средств. — М.: РФФИ; СИНТЕГ, 2003.

9. Липаев В.В. Технико-экономическое обоснование проектов сложных программных средств. — М.: СИНТЕГ, 2004.

10. Липаев В.В. Функциональная безопасность программных средств. — М.: СИНТЕГ, 2004.

11. Липаев В.В. Анализ и сокращение рисков проектов сложных программных средств. — М.: СИНТЕГ, 2004.

12. Липаев В.В. Документирование сложных программных средств. — М.: СИНТЕГ, 2005.

13. Соммервилл И. Инженерия программного обеспечения. Пер. с англ. — М.: Вильямс, 2002.

14. Тэллес М., Хсих Ю. Наука отладки. — М.: Кудиц-образ, 2003.

15. Уайт Б.А. Управление конфигурацией программных средств. Практическое руководство по Rational ClearCase. Пер. с англ. — М.: ДМК Пресс, 2002.

16. Фаулер М., Скотт К. UML в кратком изложении. Пер. с англ. — М.: Мир, 1999.

17. Фатрелл Р. Т., Шафер Д. Ф., Шафер Л. И. Управление программными проектами: достижение оптимального качества при минимальных затратах. Пер. с англ. — М.: Вильямс, 2003.

18. Якобсон А., Буч Г., Рамбо Дж. Унифицированный процесс разработки программного обеспечения. Пер. с англ. — СПб.: Питер, 2002.

19. Boehm B.W. et al. Software cost estimation with COCOMO II. Prentice Hall PTR. New Jersey, 2000.

20. Encyclopedia of Software Engineering. Vol.1 A-N; Vol.2 O-Z. Editor — In — Chief John J. Marciniak. John Wiley & Sons. Inc. 2001.

Липаев, В.В. Программная инженерия. Методологические основы [Текст] : Учеб. / В. В. Липаев ; Гос. ун-т — Высшая школа экономики. — М. : ТЕИС, 2006. — 608 с. — 1000 экз. — ISBN 5-7598-0424-3 (в пер.).

Учебник содержит курс лекций, отражающий методологические основы современной программной инженерии, обеспечивающей жизненный цикл (ЖЦ) сложных программных средств (ПС). Представлены профили международных стандартов ЖЦ систем и комплексов программ, регламентирующие в программной инженерии модели и процессы управления проектами ПС. Значительное внимание уделено системному анализу и технико-экономическому обоснованию проектов крупных ПС. Ряд лекций посвящен разработке требований, планированию, структурному и объектно-ориентированному проектированию ПС. Рассмотрены процессы управления ресурсами проектов ПС, дефекты, ошибки и риски в ЖЦ сложных комплексов программ. Представлены стандартизированные характеристики качества программных средств и проанализированы методы их выбора в проектах ПС. Детально изложены методы и процессы верификации, тестирования и оценивания корректности программных компонентов, а также их интеграции, квалификационного тестирования и испытаний крупных комплексов программ. В лекциях подробно отражены процессы сопровождения, мониторинга и управления конфигурацией в жизненном цикле ПС. Завершают курс лекций методы и процессы документирования, удостоверения качества и сертификации программных продуктов.

Учебник целесообразно использовать при обучении студентов старших курсов, аспирантов и менеджеров проектов и создании сложных комплексов программ на всем их жизненном цикле (64 часа лекций и 32 часа семинарских занятий). Курс лекций ориентирован также на заказчиков, менеджеров крупных проектов, аналитиков и ведущих специалистов, обеспечивающих все этапы ЖЦ сложных программных средств и систем, к которым предъявляются высокие требования к качеству функционирования и ограничены доступные ресурсы разработки.

УДК 004.41(075.8)
ББК 32.973.26–018я73

Учебник

В.В. Липаев

ПРОГРАММНАЯ ИНЖЕНЕРИЯ

Методологические основы

Редактор Т.П. Рындак
Корректор Т.В. Кочемасова
Художественный редактор И.В. Смирнова
Компьютерная верстка А.В. Плотников

Подписано в печать 1.12.2006. Формат 60 x 88 1/16.
Усл. печ. л. 24,03. Печ. л. 38
Тираж 1 000 экз. Заказ 4873

Издательство «ТЕИС»
115407, Москва, Судостроительная ул., 59

ППП типография «Наука»
121099, Москва, Шубинский пер., 6