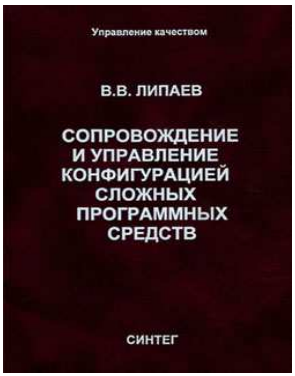

Институт системного
программирования
Российской академии наук

В.В. ЛИПАЕВ

СОПРОВОЖДЕНИЕ И УПРАВЛЕНИЕ КОНФИГУРАЦИЕЙ СЛОЖНЫХ ПРОГРАММНЫХ СРЕДСТВ

СИНТЕГ®
Москва - 2006

УДК 681.324.067
ББК 32.988
Л61

Липаев В.В.
Сопровождение и управление конфигурацией сложных программных средств. - М.: СИНТЕГ, 2006. - 372 с.

Рассматриваются особенности управления проектами, методы и процессы сопровождения программных средств (ПС). Изложены характеристики дефектов, ошибок и модификаций в сложных ПС. Представлены этапы и процедуры при сопровождении, а также ресурсы, необходимые для обеспечения изменений ПС и их характеристики качества. Значительное внимание уделено задачам и особенностям переноса программ и данных в информационных системах, а также факторам, влияющим на эффективность переноса на иные платформы, верификации и тестированию модификаций при сопровождении ПС. Представлены методы и процессы, этапы и процедуры при управлении конфигурацией программных средств. Изложены основы документирования и архивирования, организация специалистов для сопровождения ПС, стандартизации и профилей стандартов жизненного цикла, сопровождения и управления конфигурацией ПС. Отмечены особенности некоторых инструментальных систем для управления конфигурацией. Описаны задачи и направления развития и применения концепции открытых систем, стандарты, регламентирующие структуру и интерфейсы ПС с операционной и внешней средой, и особенности профиля открытых информационных систем, программных средств и баз данных.

Книга предназначена для заказчиков, менеджеров и разработчиков проектов ПС, к которым предъявляются высокие требования к качеству программных продуктов. Она может быть полезна системным аналитикам, исполнителям научных проектов и опытно-конструкторских работ, студентам и аспирантам, связанным с сопровождением и управлением конфигурацией сложных программных продуктов.

ББК 32.988

ISBN 5-89638-095-X

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Программы для вычислительных машин по своей природе значительно отличаются от аппаратурных, технических изделий, поэтому управление программными проектами имеет характерные особенности, по сравнению с другими техническими дисциплинами. Программы и данные в вычислительных машинах являются наиболее ***гибкими компонентами и подвержены изменениям*** в течение всего их жизненного цикла (ЖЦ). По особенностям и свойствам жизненного цикла программ их целесообразно делить на ряд классов и категорий, из которых наиболее различающимися являются два.

Первый класс составляют относительно небольшие программы, создаваемые одиночками или небольшими коллективами (3–5) специалистов, которые не очень думают о их трудоемкости, качестве и документировании, не предполагают их массового тиражирования и распространения как программного продукта на рынке и оценивают их как "художественные произведения". Они создаются преимущественно в вузах для обучения программированию без применения формализованных требований к функциям и допустимым затратам, индустриальных технологий и стандартов на проектирование, не подлежат независимому тестированию, гарантированию качества и/или сертификации. Для таких, а также для многих видов относительно не сложных программ, нет необходимости в длительном, регламентированном сопровождении и управлении конфигурацией множества версий и они не являются предметом анализа в данной книге.

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

- 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 (ПС) и базы данных (БД), необходимо разработчикам получать квалифицированные требования к характеристикам результатов проекта, соответствующие выделенному финансированию и квалификации исполнителей проекта;
- в проектах таких сложных программных средств и баз данных с множеством функциональных компонентов участвуют большие коллективы специалистов разной квалификации и специализации, от которых требуется высокая ответственность за качество результатов деятельности каждого из них;
- для координации этой деятельности специалистов-разработчиков при наличии единой, крупной целевой задачи создания и совершенствования проекта необходимы менеджеры проектов, а также методы, методики и комплексы средства автоматизации регламентированной технологии обеспечения его ЖЦ;
- от разработчиков проекта требуются высокие гарантии качества, надежности функционирования и безопасности применения поставляемых программных продуктов, в которые не допустимо прямое вмешательство заказчика и пользователей, не предусмотренное эксплуатационной документацией разработчиков;
- необходимо применять индустриальные, регламентированные стандартами процессы, этапы и документы, сопровождающие весь жизненный цикл комплексов программ.

Такие крупномасштабные комплексы программ являются компонентами информационных систем, увеличивающими их сложность и создающими предпосылки для последующих изменений, а также наиболее доступны для анализа пользователями и поэтому являются основным объектом их претензий при использовании. Реализация методологии управления и изменения ПС и БД зависит от многих факторов, от персонала, технических, организационных и договорных требований и сложности проекта. Множество модификаций и текущих состояний компонентов, сложных ПС и БД необходимо упорядочивать, контролировать их развитие и применение участниками проекта. Организованное, контролируемое и методичное отслеживание ***динамики изменений*** программ и данных, их слаженная разработка при строгом учете и контроле каждого измене-

ния, является основой эффективного, поступательного развития каждого крупного проекта информационной системы (ИС).

Для решения таких задач разработаны и активно применяются методы, методики и средства автоматизации ***сопровождения и управления конфигурацией***. Они позволяют представить отдельным специалистам и руководителям состояние проекта и его компонентов в любой момент времени и не допускать хаоса при коллективной модификации программ и данных. Дисциплина конфигурационного управления и сопровождения, а также ее соблюдение в значительной степени определяют технико-экономические показатели проекта, его качество, длительность применения и конкурентоспособность ПС и БД на всем их ЖЦ, что является предметом данной книги.

Методы и процессы сопровождения и управления конфигурацией играют ***стабилизирующую и организующую роль*** во всем жизненном цикле ИС. Они обеспечивают:

- расширение и совершенствование функций информационных систем и компонентов с сохранением их целостности и результатов первичных затрат;
- повышение качества функционирования комплексов программ и баз данных, решения задач пользователей в различной внешней среде;
- улучшения технико-экономических характеристик применения информационных систем и программных проектов;
- совершенствование технологий жизненного цикла сложных проектов информационных систем и комплексов программ.

Для этого при создании и сопровождении сложных, распределенных информационных систем, формировании их архитектуры, выборе компонентов и их связей целесообразно учитывать ***ряд современных концептуальных требований***:

- архитектура информационной системы должна соответствовать текущим и перспективным целям и стратегическим, функциональным задачам, создаваемой системы;
- в структуре и компонентах следует предусматривать обеспечение максимально возможной сохранности инвестиций в аппаратные и программные средства, а также в базы данных при длительном развитии, сопровождении и модернизации информационной системы;

- архитектура информационной системы должна 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;
- необходимо обеспечивать эффективное использование ресурсов информационной системы и минимизировать интегральные затраты на обработку данных в типовых режимах ее функционирования с учетом эксплуатационных затрат и капитальных вложений в создание ИС;
- 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 восстановлением функционирования ИС;
- для обеспечения перспективы развития информационной системы может быть полезно 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 комфортный, максимально упрощенный доступ конечных пользователей к управлению и результатам функционирования информационной системы на основе современных графических средств и наглядных пользовательских интерфейсов.

Значительная часть приведенных требований обусловлена реальной гибкостью и естественной динамикой интенсивного развития, совершенствования и расширения функций современных ИС. 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. Гибкость модификации ИС при развитии обеспечивается рядом принципов и правил структурного построения программных средств и баз данных и их компонентов, а также взаимодействия между ними. Эти правила направлены на стандартизацию и унификацию структуры и взаимодействия компонентов разного ранга и на-

значения в пределах проблемной области. Некоторая часть принципов и правил имеет достаточно общий характер и может применяться практически всегда. Другая часть отражает проблемную и машинную ориентированность класса ПС и БД и подлежит отработке для эффективного применения в соответствующей области.

Для обеспечения эффективной разработки и сопровождения необходимо формулировать и соблюдать ряд принципов и правил структурного построения ИС. Их формализация и выполнение обеспечивают значительный эффект в снижении трудоемкости и длительности разработки и сопровождения ПС, БД и их версий. Потеря гибкости архитектуры ИС, некоторое возрастание ресурсов, необходимых для их реализации, обычно полностью компенсируются повышением технико-экономических показателей процессов разработки, развития и сопровождения ИС. Одним из важнейших и эффективных путей решения этой проблемы является применение ***концепции и совокупности стандартов открытых систем***.

В процессе эксплуатации версий ПС и БД 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 ПС и БД. Аналогичные предложения могут поступать от разработчиков комплекса программ. Для общения с пользователями и накопления информации о выявляемых недостатках в широко тиражируемых сложных ПС и БД целесообразно выделение группы специалистов высокой квалификации, овладевших всеми функциональными возмож-ностями данной ИС.

По мере развития применения сложных ИС стало ясно, что интегральные затраты на сопровождение и создание новых версий ПС и БД ***могут значительно превосходить затраты на разработку*** их первой версии. Действительно, если разработка первой версии сложного ПС продолжается 3 года, а последующая его эксплуатация, модификация и создание ряда версий происходит в течение 20 лет, то можно ожидать, что затраты на

этих интервалах времени соизмеримы. Даже при предположении, что сопровождением и разработкой новых версий будут заняты 20% специалистов, осуществивших создание первых версий, суммарные затраты на эти работы превысят первичные в полтора – два раза. Опыт последних лет показал, что во многих случаях для сопровождения и развития версий необходимо практически такое же число специалистов, которое 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, прототипирования 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. Оно зависит от интенсивности создания новых версий, длительности жизненного цикла каждой из них, тиража версий, технологии их разработки, качества программ и требований пользователей. На начальных этапах проектирования первой версии ПС целесообразно прогнозировать последующую организацию и ресурсы, необходимые для регулярного сопровождения и развития всего множества версий. Очень часто разработчики нового ПС не предусматривают этот процесс и требующиеся ресурсы, что значительно снижает эффективность последующего применения созданного ПС. По некоторым оценкам [4, 13, 41] непосредственным программированием ***новых компонентов*** в мире занято около 15 – 20% специалистов, участвующих в создании программных продуктов.

Общие закономерности развития и изменения комплексов программ выявлены при анализе сопровождения крупной операционной системы OS/360 [18], которые подтверждены в других крупных проектах [20] и практически сохранились до настоящего времени. Были обобщены результаты процессов разработки 21 версии этой системы (за исключением двух самых

первых) на интервале свыше 12 лет. Трудоемкость разработки модификаций можно считать пропорциональной интервалу времени между регистрациями очередных версий, который принят за единицу измерения сложности создания изменений версий.

В качестве меры трудоемкости сопровождения и создания очередной версии использовано число модулей, подвергшихся изменениям и дополненных в версию ПС. Кроме того, оценивалась интенсивность работ по созданию версии, которая измерялась числом измененных модулей на единицу времени. Модули оформлялись стандартизовано и имели ограниченные размеры. Исследованные версии операционной системы развивались, по существу, без ограничений на используемые ресурсы по памяти и производительности реализующих ЭВМ. Исследованные зависимости обобщены в несколько "законов" динамики развития сложных ПС [18].

Первый "закон" состоит в постоянном изменении и увеличении размера ПС до тех пор, пока не сочтено экономически выгодным прекратить сопровождение и разработать полностью новый комплекс программ для аналогичных целей. Для развития программ имеются ***внешние стимулы*** со стороны пользователей, предъявляющих непрерывно новые требования к функциям и характеристикам программ, а также расширение внешних условий и среды функционирования комплекса программ. Кроме того, для изменения программ имеются ***внутренние стимулы***, заключающиеся в необходимости устранения недочетов проектирования и ошибок программирования, выявленных при эксплуатации. В результате образуется цикл взаимного приспособления изменяющегося ПС и расширяющихся внешних условий, в которых оно может эффективно эксплуатироваться.

В процессе развития и изменения ПС его первичная, четкая структура искажается и усложняется, что приводит к возрастанию сложности при модификации программ. Эти особенности отражаются ***вторым "законом" – возрастания энтропии*** (меры неопределенности) структуры и функций ПС при сопровождении, если не предпринимаются специальные усилия для ее уменьшения. По мере развития комплекса программ компоненты и интерфейсы становятся все более хаотичными по

структуре и процессу функционирования и все более отличаются от упорядоченного, первоначально задуманного проекта, характеризующегося наименьшей энтропией. Это происходит до момента возникновения нерентабельности дальнейшего сопровождения данного комплекса и целесообразности его замены на новое ПС.

Третий "закон" состоит в относительно гладком среднестатистическом росте масштаба и сложности ПС. Обобщенные характеристики размера могут иметь случайные колебания на небольших интервалах времени, однако сохраняется регулярная составляющая непрерывного роста масштаба программ. Этому процессу соответствуют постоянные усилия коллектива специалистов, осуществляющих сопровождение ПС. Однако доля продуктивно используемых ресурсов уменьшается по мере старения программного средства. В результате увеличиваются интервалы времени между выпусками очередных базовых версий.

Кроме детерминированной и случайной составляющих, прирост числа модулей и доля измененных модулей на версию, имеют периодическую составляющую, которая может быть обусловлена взаимодействием процессов развития ПС и исправления ошибок. При эксплуатации системы накапливаются замечания и дополнительные требования, а также возрастает опыт разработчиков, что стимулирует проведение доработок программ. Эти доработки подготавливаются для очередной версии и вводятся в нее. В начале эксплуатации новой версии обнаруживается некоторое число вторичных ошибок и недоработок изменений, что вынуждает проектировщиков ограничивать расширение функций системы и сосредоточивать усилия на повышение надежности и других характеристик качества функционирования. В результате может появляться две-три промежуточные версии, близкие по масштабу и функциям и отличающиеся, в основном, уровнем качества. После того как сокращается количество претензий пользователей к качеству функционирования версии, появляется возможность удовлетворить новые требования к развитию функциональных характеристик ПС.

На величину интервала времени между отработанными версиями влияют также административные планы, запаздывания оформления полной документации и инерционность в пе-

редаче базовых версий пользователям. В худшем случае возможно ***явление распада системы***, когда чрезмерный рост доработок в версии системы делает ее развитие неуправляемым и резко ухудшает эксплуатационные характеристики. Период колебаний объема изменений и соответствующих им характеристик качества для крупномасштабных ПС в разных проектах [18] может составлять 1 – 2 года.

В процессе развития OS/360 установлен размер среднего прироста системы на каждую версию, составивший около 200 модулей, который оставался (более 10 лет) для коллектива почти постоянным, несмотря на рост квалификации специалистов, развитие технологии отладки и другие факторы. При этом ПС увеличился по объему от 1 тыс. модулей в первых версиях, почти до 5 тысяч модулей в последних. В результате появилось понятие "критической массы" или "***критической сложности***" 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 доработки и устранения ошибок проведенной модернизации.

Прогрессивному развитию программ способствует пропорциональный рост числа вносимых при этом ошибок в изменениях. Рост сложности ПС способствует увеличению энтропии, которая требует затрат на "антирегрессионную" деятельность – борьбу с ошибками и отставанием корректировок документации, затрат на повышение квалификации специалистов. Пренебрежение "антирегрессионной" деятельностью приводит к накоплению потребностей в этой работе в условиях повышающейся сложности программ и ограниченного бюджета. В результате приходится временно снижать деятельность по совершенствованию программ с тем, чтобы не выйти за допустимые затраты ресурсов [18, 20]. При ограниченных трудовых ресурсах возможна предельная ***критическая сложность сопровождаемых программ***, при которой устанавливается динамическое равновесие между доработками и вносимыми ошибками, вследствие чего качество ПС не улучшается.

При организации сопровождения и конфигурационного управления многочисленными версиями крупномасштабных

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 неодинаковой 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 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 не престижны среди рядовых программистов и недооцениваются руководителями проектов.

Основная цель книги – представить отечественным специалистам современный комплекс задач, методов и стандартов сопровождения и управления конфигурацией сложных, многоверсионных, тиражируемых программных средств и баз данных высокого качества. Изложение ориентировано на коллективную, групповую работу "команд" специалистов над средними и крупными программными проектами. При этом предполагается, что процессы и технология сопровождения комплексов программ и

создания документов опирается на совокупность современных, автоматизированных методов и инструментальных средств.

Для координированного объединения усилий специалистов в крупных проектах программных средств и сокращения непроизводительных затрат вследствие несогласованных действие участников проекта, необходима ***организация коллективов и жесткая дисциплина*** при выполнении частных заданий. Такая дисциплина должна базироваться на стандартах и нормативных документах, которые обобщили и сконцентрировали опыт многих специалистов. Комплекс таких документов должен регламентировать методы и методики при реализации конкретного проекта и обеспечивать возможность контроля результатов технологических процессов жизненного цикла информационных систем, а также применение созданных продуктов по их прямому назначению.

Для решения важнейших проблем развития и применения современных информационных систем, требуется ***подготовка и воспитание квалифицированных специалистов в области индустрии сопровождения, длительной регламентированной модификации и управления конфигурацией программных средств и баз данных***. Необходимо их обучение методам и современной программистской культуре промышленного развития и совершенствования крупных высококачественных проектов ПС и БД. Они должны уметь формализовать требования и изменять конкретные значения характеристик качества функционирования и применения сложных комплексов программ, с учетом тех ограниченных ресурсов, которые им доступны.

В жизненном цикле сложных ПС для обеспечения их сопровождения и управления конфигурацией целесообразно выделять специалистов, ответственных за соблюдение промышленной технологии модификации и совершенствование программ, за измерение и контроль изменения качества ПС в целом и их компонентов. Необходимо научить специалистов анализу и оцениванию конкретных факторов, влияющих на характеристики качества функционирования программных продуктов со стороны реально существующих и потенциально возможных дефектов и ошибок в программах.

В книге рассмотрены цели и концепция методов и процессов сопровождения, планирование, оценка и распределение ре-

сурсов при сопровождении сложных программных средств. Изложены требования к характеристикам и обеспечению качества при сопровождении, этапы и процедуры реализации изменений, поставки и завершения сопровождения версий ПС. Отражены особенности верификации и тестирования модификаций, а также организации коллектива специалистов для сопровождения программных продуктов. Значительное внимание уделено принципам и задачам переноса программ и данных в информационных системах, а также анализу факторов, влияющих на эффективность процессов переноса на иные аппаратные и операционные платформы.

Изложены концепции управления программными проектами, а также управления конфигурацией ПС. Представлены этапы, процедуры и основные стандарты конфигурационного управления в жизненном цикле ПС. Выделены и проанализированы организация баз данных, документирование и архивирование для сопровождения и управления конфигурацией программных средств.

Глава 1

СОПРОВОЖДЕНИЕ И МОДИФИКАЦИИ В ЖИЗНЕННОМ ЦИКЛЕ ПРОГРАММНЫХ СРЕДСТВ

1.1. Особенности управления проектами программных средств

Приступая к разработке крупных проектов, заказчики и исполнители, прежде всего, пытаются понять целесообразно ли их создание и оценить какова будет возможная эффективность применения готового продукта, оправдаются ли затраты на его разработку и использование. Поэтому такие проекты традиционно начинаются с анализа и разработки ***требований и технико-экономического обоснования*** предстоящего жизненного цикла проекта и эксплуатации предполагаемого продукта [26, 34]. Заказчику проекта системы необходимо оценить реальную потребность в его создании и возможную конкурентоспособность, а потенциальному разработчику-поставщику создаваемого продукта, провести оценку реализуемости проекта в условиях и ***ресурсах***, предлагаемых заказчиком (см. стандарт **ISO 15288**).

Достижение высоких значений качества комплексов программ существенно зависит от ***качества технологии и инструментальных средств***, используемых разработчиками для обеспечения ЖЦ ПС. Оценивание достоинств технологической базы ЖЦ позволяет прогнозировать возможное качество программного продукта 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 технологий ***крупномасштабных проектов*** сосредоточены в методологии **CMM** (Capability Maturity Model – ***система и модель для оценки зрелости***) – в комплексе технологических процессов жизненного цикла проектов [31, 33]. Модель основана на формализации и использовании пяти уровней зрелости технологий поддержки ЖЦ проектов, 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 ПС. Тем самым при выборе требований к характеристикам 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 реализации функций системой качества технологических процессов и их результатов.

Описание процессов 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 материал, определяющий правила создания и функционирования системы управления жизненным циклом ПС, методы постепенного повышения культуры и качества разработки. 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, фиксироваться в организационных структурах и определять производственную тактику и стратегию корпоративной культуры разработки, производства и системы обеспечения качества ПС.

В 2002 году американский институт программной инженерии (**SEI)** опубликовал новую модель **CMMI**, уточняющую предшествовавшие модели **CMM** и учитывающую требования существующих международных стандартов [31]. Внедрение этой модели акцентировано на улучшении процессов управления проектами ПС, обеспечении их высокого качества и конкурентоспособности, с основной целью – сделать процессы проекта более управляемыми, а результаты – предсказуемыми. Значительное внимание в **CMMI** уделяется процессам разработки и учету итераций требований заказчиков, их прослеживанию к функциям, компонентам, тестам и документам проекта. Концепцию, определяющую функциональную пригодность и качество продукта, рекомендуется сопровождать проверками возможных сценариев событий при его применении.

Модель базируется на сохранении концепции пяти уровней зрелости СММ. Первый и пятый уровни отличаются значительной нестабильностью и неопределенностью процессов в различных проектах, поэтому при уточнении и детализации содержания процессов полезно ограничиться тремя основными, средними уровнями:

- ***второй уровень*** – формализует базовое управление проектами: управление требованиями; планирование; мониторинг и контроль; измерение и анализ; обеспечение качества; управление конфигурацией;
- ***третий уровень*** – содержит стандартизацию процессов: разработка требований; интеграция продукта; верификация; валидация; обеспечение стандартного процесса; обучение; интегрированное управление; управление рисками; анализ и разрешение проблем (устранение дефектов);
- ***четвертый уровень*** – определяет количественное управление: качеством процессов; количественное управление всем проектом.

Рекомендуется на каждом более высоком уровне зрелости применять все процессы предыдущих нижних уровней. Упорядочение и оценка используемых процессов в соответствии с уровнями, позволяет устанавливать производственный потенциал предприятий – разработчиков программных продуктов по прогнозируемому качеству результатов их деятельности и возможности сертификации. Это уменьшает зависимость заказчиков и пользователей от возможных недостатков исполнителей проектов и позволяет их выбирать с учетом прогнозируемого качества продуктов.

Практически все перечисленные процессы на выделенных уровнях **CMMI** соответствуют, регламентированным в стандарте **ISO 12207** и в профиле стандартов жизненного цикла сложных ПС (см. п. 4.1 и п. 4.2). Жизненный цикл программных средств отражается в стандартах набором процессов, этапов, частных работ и операций в последовательности их выполнения и взаимосвязи, регламентирующим ведение разработки, сопровождение и эксплуатацию от 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.

Современные модели ЖЦ ориентированы преимущественно на сложные, критические ПС обработки информации и управления в реальном времени. К таким ПС предъявляются наиболее высокие требования к качеству функционирования, они создаются и развиваются большими коллективами специалистов в течение длительного времени. В стандарте **ISO 12207**

(п.7.1) определены также ***основные процессы и этапы управления проектами и их развития***. Каждый этап управления проектом комплекса программ в этом стандарте подробно детализирован в Руководстве **ISO 16326** с учетом положений стандарта **ISO 10006** и сокращенно изложен ниже. В этих ***стандартах ЖЦ рекомендуется*** руководителям программных проектов постоянно влиять на ход работ, санкционируя определенные работы или приостанавливая их, если они могут повлиять на другие области деятельности, ***определять методологию и технологию реализации проекта*** необходимые для:

- прогнозирования и соответствующего предотвращения или минимизации неблагоприятного воздействия потенциальных проблем проекта;
- принятия временных и/или постоянных решений в жизненном цикле ПС;
- решения возникающих проблем и устранения дефектов проекта ПС;
- принятия ответственности за проект в целом, его процессы, работы, ресурсы, продукты и результаты.

Процессы управления проектом, работы и задачи необходимо выполнять неоднократно, чтобы реализовать требования и цели проекта. Основываясь на выбранной модели жизненного цикла программного средства, некоторые процессы, работы и задачи можно выполнять одновременно, при этом они могут быть взаимосвязаны или скоординированы в зависимости от этапов проекта. Процессы управления проектом состоят из общих работ и задач, которые могут быть использованы любой стороной, управляющей соответствующими процессами. Руководитель должен отвечать за управление проектом, работами и задачами соответствующих процессов, такими как заказ, поставка, разработка, эксплуатация, сопровождение или вспомогательные процессы.

Особенности и проблемы развития крупных программных проектов в значительной степени определяются их сложностью, на которую влияют [5, 19, 49]:

- изменение сложности функций программных продуктов и систем;
- увеличение сложности проектной среды;

- изменение требований заказчика и пользователей, по мере развития этапов жизненного цикла ПС;
- изменение процессов управления в предприятии и состава участников проекта.

Увеличение сложности функций программного комплекса в ходе его эволюции обычно происходит за счет добавления новых возможностей и функций. Возрастающая сложность может превратиться в проблему, если своевременно не предприняты необходимые действия для сохранения архитектурной и функциональной целостности системы. С точки зрения управления сопровождением и конфигурацией ПС усложнение может происходить по направлениям:

- увеличение размеров комплекса программ;
- изменение сложности функций и решаемых проблем;
- увеличение сложности архитектуры ПС;
- добавление готовых программных компонентов сторонних разработчиков;
- увеличение числа поддерживаемых аппаратных и операционных платформ.

Увеличение размеров программы и сложности архитектуры и решение более сложных проблем с помощью существующего ПС происходит для реализации новых функций. Это увеличивает сложность системной архитектуры, ведет к увеличению размеров программы, что выражается как в числе строк кода, так и в количестве файлов компонентов. Совместное и многократное использование компонентов, а также применение компонентов сторонних производителей – это естественно для ускорения разработки программных проектов. Привлечение различных фирм и повторное использование собственных компонентов в ряде проектов увеличивает сложность процессов управления проектами. Такие возможности должны поддерживаться инструментами проектирования. Нужно гарантировать, что применяемые компоненты, независимо от их происхождения, контролируются средствами управления проектом.

Возможность поддержки нескольких аппаратных и операционных платформ требует решения специфических задач управления проектами. По мере увеличения числа платформ возрастают нагрузки на подразделение, тестирующее продукт. Часто выделяется группа, занимающаяся проблемой оптимизации новой версии для основных платформ с сохранением её ра-

ботоспособности на других. В отношении средств управления поддержка новой платформы требуется доступ к программе на данной платформе. Для этого нужно, чтобы подобный инструмент функционировал на ней, либо существовал способ переносить новую программу из первичного хранилища на эту платформу. Второй аспект относится к сборке программы, поскольку она тесно связана с управлением конфигурацией. Если нельзя понять, какие версии файлов использовались в определенной сборке, то невозможно воспроизвести заявленные заказчиком или пользователем ошибки, а в дальнейшем будет затруднительно выпускать пакеты исправлений к программе.

Сложность среды разработки и сопровождения проекта обычно возрастает по следующим причинам:

- возникает необходимость параллельной разработки компонентов и функций;
- увеличивается количество и частота выпуска новых версий проекта;
- растет число участников команды разработчиков;
- проектные группы размещаются в разных, может быть удаленных, местах.

Увеличивающаяся сложность в ходе развития функций ПС обычно требует изменений в составе команды участников проекта. Чаще всего это ведет к увеличению размера команды или количества таких команд разработчиков. Увеличение размера команды повышает интенсивность использования файлов и ресурсов системы. Инструменты проектирования должны обеспечивать блокировку одновременных изменений общих файлов, для модификаций, вносимых двумя или несколькими разработчиками в один файл в одно и то же время. Инструменты должны предоставлять возможность слияния изменений для объединения результатов параллельной работы. Еще одна сложность организации среды разработки возникает при необходимости обеспечения взаимодействия команд, находящихся в географически удаленных регионах, в разных офисах, городах или странах. Географическое удаление усложняет процесс в основном за счет трудностей в общении специалистов.

Количество, поддерживаемых версий программного продукта и частота их появления могут существенно влиять на выбор процессов и инструментов, используемых для управления

проектом. Чем больше версий поддерживается, тем более отлаженными и автоматизированными должны быть процессы обновления, исправления, тестирования и сопровождения. Кроме того, многие ПС выпускаются в разном исполнении в зависимости от требований конкретного заказчика. В этом случае количество поддерживаемых версий зависит от количества заказчиков (рис. 1.1).

Одновременное существование нескольких версий и нескольких вариантов программы – наиболее сложная конфигурация, которая должна поддерживаться инструментом управления. При таких условиях необходимы специальные средства для распространения изменений среди разных вариантов программ пользователей. Частота появления новых версий также способствует увеличению сложности среды разработки. Для уменьшения ошибок и повышения эффективности работы при более частых выпусках версий нужна дополнительная автоматизация.

По мере продвижения проекта по этапам жизненного цикла, проектные требования к процессам и инструментам управления могут изменяться. Требования к ПС при смене этапов ЖЦ способны как увеличиваться, так и уменьшаться. Хорошие инструменты управления предоставляют максимальные возможности для внесения изменений, сохраняя при этом эффективный контроль над проектом. По мере приближения выпуска базовой версии программного продукта необходимо определять, какие изменения можно сделать, а какие нельзя, обычно на основе оценки рисков. На этом этапе нужно усиливать контроль и четко отслеживать проводимые модификации. В каждом этапе ЖЦ следует найти правильный баланс между аудитом и соблюдением правил. ***Аудит проекта*** – это информация о том, кто сделал изменение, в чем оно заключалось, когда и для чего было сделано. Применяя аудит, можно ослабить контроль, чтобы не препятствовать внесению исправлений, при этом останется возможность проследить любое изменение. Аудит должен быть прозрачен для других пользователей и не вызывать дополнительных накладных расходов.

Изменение процессов и состава участников проекта наиболее непредсказуемо и может иметь тяжелые последствия. Изменения в составе специалистов, работающих над проектом, трудно предсказать и запланировать, поскольку они происходят по множеству причин. Каждый руководитель действует на ос-

нове опыта своих предыдущих проектов, что неизбежно влияет на стиль управления и организацию производства. Смена руководства часто ведет к изменению процесса и инструментов, особенно если новый менеджер назначен для решения возникших с проектом проблем. Другой фактор, влияющий на процесс и инструменты управления проектом – разный опыт работы сотрудников.

Управление процессами должно определять, а менеджер программного проекта контролировать работы, гарантирующие поставку продуктов в соответствии с условиями договора, включение в проект только тех заданий, которые обеспечивают успешную его реализацию и создание соответствующего продукта. Для этого должен быть подготовлен согласованный между заказчиком и разработчиком ***первичный документ***, в котором определены ***цели и концепция проекта***, ***требования*** к характеристикам продукта и необходимые ***ресурсы*** для его реализации. Эти данные должны быть предварительно сбалансированы, и обеспечивать реализацию целей проекта при выделенных ресурсах. Однако масштабы целей и функций сложных проектов имеют устойчивую тенденцию изменяться и увеличиваться по мере развития, а первоначально выделяемые ресурсы не удовлетворять их реализацию. Массовое создание таких проектов промышленными методами и большими коллективами специалистов вызвало необходимость их четкой организации, планирования работ по затратам, этапам и срокам реализации. Технико-экономическое обоснование проектов должно содержать ***оценки рисков*** реализации поставленных целей, обеспечивать возможность планирования и выполнения жизненного цикла продукта или указывать на недопустимо высокий риск его реализации и целесообразность прекращения разработки [28, 54].

Подготовка и определение области управления проектом заключаются в установлении выполнимости процессов при наличии, достаточности и соответствии персонала, средств, инструментальной среды программной инженерии и технологий, необходимых для выполнения проекта и управления им, и определении обоснованных (экономически и технически) сроков его реализации. При этом проводится выбор стратегии разработки ПС, проект может состоять из готовых собственных компонен-

тов, программных средств посторонней разработки или из их комбинации.

Рис. 1.1

Область управления, описание проектируемого продукта, характеристики и методы измерения или оценки характеристик связана с задачами:

- документирования технико-экономического обоснования проекта, его целей и задач;
- перевода спецификаций требований заказчика в результаты проекта и работы, реализуемые при организации и реализации проекта;
- обеспечения персонала разработчиков заданиями и детализированными требованиями в рамках данной области;
- оценки результатов работ, способствующей удовлетворению требований заказчика к результатам, выдвинутых для данной области и продукта.

Наиболее благоприятный сценарий, когда новый программный проект во многом подобен проекту, ранее реализованному предприятием. В этом случае высока вероятность успешного выполнения нового проекта. Работа по подготовке и определению области управления может быть затруднена, если новый проект не имеет аналогов, и отсутствуют предварительные наработки в данном предприятии. Для уникального, не имеющего аналогов проекта должны быть предприняты специальные мероприятия, обеспечивающие определение области управления и соответствующий надзор за проектом. Мероприятия должны сопровождаться соответствующими анализами и оценками опыта аналогичных проектов и экспертными заключениями. Все изменения, вносимые в область управления, и требования должны быть оценены по их влиянию на стоимость, графики работ, риски и эффективность характеристик проекта.

Должны быть предусмотрены специальные мероприятия по определению и документированию ***требований к характеристикам качества*** (**ISO 9126** [1, 25, 45]); в частности, когда ПС встраивается в систему более высокого уровня, некоторые функции распределяются между программными и техническими средствами или между данным программным средством и другими, внешне взаимосвязанными с ним ПС или системами. Согласование с заказчиком требований к проекту достигается в результате итерационного выполнения процессов в жизненном цикле программного проекта. Итерации необходимы вследст-

вие наличия изменений в требованиях, среде проектирования, бюджете и графике работ по проекту, а также в процессах проектирования, что приводит к повторному согласованию и внесению изменений в проект.

Целью планирования является успешное завершение проекта, работ или решение задачи для этого должны быть:

- установлена и отслежена взаимосвязь (трассировка) между требованиями к системе и к программному продукту, требований к проекту и к тестированию;
- установлены и проконтролированы соответствующие интерфейсы как часть спецификаций требований к ПС и соответствующие документы, описывающие интерфейсы;
- детально определены требования по тестированию, распространению, вводу в действие (инсталляции) программного продукта;
- оценены рабочие загрузки специалистов, которые зависят от типа проекта: новая разработка, встраивание или интеграция в систему, изменение (модификация) готового программного продукта, подключение к различным операционным системам.

Планы проектирования должны устанавливать модель жизненного цикла ПС, задачи, их распределение по этапам и соответствующие ресурсы. В программном проекте должен быть определен основной график работ и вспомогательные графики, которые связаны и согласованы с основным графиком. План должен быть применен так, чтобы обеспечить управление программным проектом на соответствующих уровнях его детализации с использованием соответствующих технологий в зависимости от размера, сложности, критичности и риска проекта. Оценки проекта при планировании, должны охватывать [6, 12, 24]:

- инфраструктуру системы и роль в ней программного продукта;
- стоимость реализации соответствующих процессов и этапов;
- потребности в ресурсах, обеспечивающих жизненный цикл ПС, включая соответствующее управление и контроль;
- оценку и контроль характеристик качества программного продукта;
- управление риском в жизненном цикле ПС;

- обеспечение ЖЦ инструментальными средствами;
- задания, выполняемые в каждом процессе и (или) работе ЖЦ ПС.

Планирование является постоянной и повторяющейся работой, при выполнении которой оценивают, уточняют и, при необходимости, корректируют ход выполнения проекта. Руководители программного проекта должны использовать соответствующие процессы, способствующие проведению перепланирования и уточненных оценок в жизненном цикле ПС. В каждом программном проекте имеется масса взаимосвязей, поэтому для уточнения исходного плана обычно требуется несколько итераций. Должны быть разработаны ***планы вспомогательных процессов*** – управления конфигурацией, документирования, обеспечения качества, так как эти процессы обычно являются важной частью проекта. Они должны быть привязаны к основному плану управления программным проектом, и обес-печивать его реализацию, могут быть оформлены в виде отдельных планов или включены в основной план управления проектом.

Изменение процессов разработки и сопровождения ПС обычно проводится с целью разрешения проблем в существующем процессе. Такие идеи нередко оказываются правильными и дают нужный эффект. Иногда корректировка процессов вызывается пересмотром планов и правил функционирования и документооборота предприятия. Для этого необходимо выработать рекомендации по элементам, основам и практическому применению ***системы качества*** (см. рис. 1.1) [25]. Обязанностью руководителя проекта является общее управление взаимосвязями между проектными процессами для обеспечения требуемого качества программного продукта:

- полный план тестирования должен быть определен в начале жизненного цикла программного проекта;
- необходимо учесть, что следствием любого нарушения графика работ, предшествующих тестированию, без соответствующей корректировки даты поставки может быть недостаточно полное тестирование продукта;
- должны быть установлены строгие правила регистрации, хранения, модернизации, резервирования и сопровождения программ, контрольных, тестовых данных и среды тестирования;

- должна быть разработана стратегия возможности возврата при тестировании к исходному состоянию изменений продукта;
- необходим интегрированный план сборки системы, компонентов и модулей ПС в соответствии со стратегией выпуска версий комплекса программ и системы;
- должно быть представлено подтверждение функциональных возможностей программного продукта и системы, показывающее явное соответствие их возможностей текущим и запланированным требованиям заказчика;
- для обеспечения процесса управления следует ориентироваться на контрольные тесты их результаты, используемые в процессе разработки.

Выполнение и контроль работ обычно начинают после получения руководителем программного проекта полномочий от заказчика по расходованию основных финансовых средств (см. рис. 1.1). При реализации контрольной части этой работы руководитель проекта должен отвечать за надзор, обеспечивающий обнаружение и анализ любых отклонений от запланированного выполнения контролируемого процесса, и внесение соответствующих корректировок в ход данного процесса. Он должен использовать контрольную часть данной работы в целях выполнения корректирующих действий для решения возникших проблем и проведения превентивных мероприятий с целью предотвратить возникновение прогнозируемых проблем. Инструментарием надзора и контроля за выполнением процесса являются измерения характеристик компонентов и ПС в целом, процесс контроля вносимого изменения, а также оценки и аудиторские проверки программных процессов и продуктов. Измерения ПС могут быть использованы для проверки соответствия между требуемыми функциями программного продукта, его функциями и сценариями при эксплуатации. Для обеспечения взаимодействий между персоналом полезно иметь централизованную, обновляемую базу данных результатов управления проектом.

Основная цель проекта может быть не достигнута при выборе варианта его выполнения в минимальные сроки или с наименьшими затратами, (например, для систем, критичных по безопасности, требуется достаточно полное тестирование). Процессы, связанные с риском и ***управление проектным риском***, охватывают неопределенности при планировании проекта

и требуют структурированного подхода [28, 53, 54]. Целями данных процессов являются минимизация воздействия потенциально неблагоприятных событий и выработка предложений по совершенствованию проекта. Риск связан либо с проектными процессами или инструментарием, либо с соответствием продукта целям и ресурсам проекта:

- исправление дефектов графика работ необходимо тщательно проанализировать, и оно не должно оказывать негативного влияния на эффективность, стоимость или риск проекта;
- для уменьшения риска целесообразно проводить демонстрации, позволяющие заказчикам и покупателям оценивать функциональные возможности программного продукта до выбора его поставщика;
- целесообразно использовать макетирование для разработки части функций ПС с целью демонстрировать реализуемость его функциональных возможностей в целом;
- работы по проектированию критических систем не следует проводить без наличия достаточных экспертных знаний в области соответствующей тематики проблемной области;
- при реализации проекта следует совместно с заказчиком поэтапно проводить анализы базовой версии требований к ПС, обеспечивающие соответствие целям проекта или их корректировку по стоимости, срокам и эффективности;
- необходимы постоянные контакты с заказчиком во избежание неожиданных изменений в требованиях, стоимости, графике работ и эффективности проекта.

Проверка и оценка проекта должны проводиться по результатам анализа программных продуктов, работ и задач. Необходимыми элементами успешной реализации программных проектов являются периодические анализы и оценки хода выполнения и завершения заданий. Руководитель проекта должен подтверждать соответствие прогнозируемых и реальных функций программного продукта, определять набор измерений характеристик ПС, обеспечивающий понимание развития жизненного цикла программного средства. Ход работ должен быть определен по фактическим измерениям: размера продукта, объема работ, их стоимости, графика выполнения этих работ, наличия или разработки модулей и компонентов, объема заданий, подлежащих выполнению. Руководитель должен непосредственно

проводить анализ деятельности групп специалистов и анализ хода работ для доклада заказчику о их состоянии.

Оценку процессов выполнения заданий рекомендуется [5, 19] проводить специалистами знающими: проектные требования, используемые технологии, создаваемые продукты, технические требования к ним, соответствующие процессы и инфраструктуру системы. Проверка выполнения графика работ, особенно в части создания программного продукта, должна быть структурирована и проводиться с целью получить реальную оценку состояния дел. Для реальных оценок ПС, обеспечивающих точную оценку проекта, необходимо использовать измерения программ, анализ заработной платы и результаты инспекционных проверок, сквозного контроля, а также совместных и односторонних проверок. Необходимо документально оформлять наиболее существенные и оперативные проблемы, рассмотренные в ходе проверок и оценок, и решения, принятые по их результатам. Инструментами управления выполнением программного проекта и контроля за его выполнением являются измерения программного средства и контроль за вносимыми изменениями. Руководитель проекта должен разработать программу измерений характеристик ПС для обозначения, количественного определения и оценки функционального риска, для определения областей технического и финансового риска.

Руководителем проекта ПС должна быть определена ***отчетность по процессам проекта***: 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 заказчиком, руководителем проекта и менеджером вспомогательного процесса на соответствующем уровне их полномочий по организационному управлению. Следует определить контрольные точки проекта и результаты, установленные для них, чтобы отчетные материалы были представлены точно и своевременно в соответствии с установленными планами. Это требование является общим для всех контрольных точек, связанных с выполнением договорных обязательств по процессам, поэтому необходимы синхронизация соответствующих планов и своевременное уведомление руководства о всех затруднениях, возникающих при выполнении процессов. Син-

хронизация всех планов может быть затруднена при наличии ряда субподрядных соглашений и заданий, но упрощена при наличии единого основного плана.

Архивные данные по проекту являются основным источником для анализа и оценивания возможностей и эффективности процессов проекта, рабочих характеристиках ПС и обобщении полученного опыта. Архивные данные общей базы данных предприятия, следует использовать для усовершенствования процессов жизненного цикла. В каждом предприятии должна быть создана официальная система для сбора, анализа, обобщения, архивирования и поиска архивных данных по проектам. Для этого может быть использована ***стратегия эффективного управления конфигурацией***, которая является ключевой в проектах ПС для управления изменениями.

Завершение управления программным проектом должно обеспечивать его оценку и гарантировать, что он удовлетворяет проектным требованиям, критериям и процедурам. Подобные оценки могут быть выполнены по мере успешного создания программных продуктов, реализации процессов, работ и задач. Результаты и отчетные материалы по программным продуктам, процессам, работам, тестам или задачам должны быть проверены на полноту и, если они признаны полными, сохранены в архиве в соответствии с договорными требованиями и требованиями, принятыми в предприятии. Иногда предприятия проходят сертификацию, чтобы показать своим заказчикам, что их процесс разработки и сопровождения соответствует промышленным стандартам, а выпускаемая продукция имеет высокое качество. В результате может производиться корректировка процессов, которая нередко ведет к смене инструментов управления проектом, к выбору другого способа применения этих инструментов или к созданию еще одного уровня автоматизации в дополнение к существующим. Необходимо четко представлять, каким образом оно сможет повысить производительность и качество работ. В процессах завершения проекта следует использовать полученные ранее результаты с соответствующими данными и отчетными материалами.

По окончании основного цикла работ по управлению созданием базовой версии проекта, должны быть выданы ***предложения по сопровождению и конфигурационному управлению***

последующими версиями и документами программного продукта, по их возможной модификации, размещению, переносу на иные платформы, носителям и длительности хранения информации. Этой части жизненного цикла крупномасштабных комплексов программ посвящено основное содержание данной книги.

1.2. Особенности дефектов, ошибок и модификаций в сложных программных средствах

Статистика ошибок и модификаций в комплексах программ и их характеристики в конкретных предприятиях и типах проектов ПС могут служить ***ориентирами для разработчиков при распределении усилий на сопровождение*** 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 сопровождении ПС может 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 систематизации и выявлении. Этому могут помогать оценки типовых дефектов и корректировок, путем их накопления и обобщения по опыту сопровождения определенных классов ПС в конкретных предприятиях.

Характеристики изменений ПС непосредственно связаны с достигаемой корректностью, безопасностью и надежностью функционирования сопровождаемых программ и ***помогают***:

- оценивать реальное состояние проекта и планировать необходимую трудоемкость и длительность для его завершения;
- выбирать методы и средства автоматизации тестирования и отладки программ, адекватные текущему состоянию раз-

работки и сопровождения ПС, наиболее эффективные для устранения определенных видов дефектов;

- рассчитывать необходимую эффективность контрмер и дополнительных средств оперативной защиты от потенциальных дефектов и не выявленных ошибок;
- оценивать требующиеся ресурсы ЭВМ по расширению памяти и производительности, с учетом затрат на реализацию контрмер при модификации и устранения ошибок.

Понятие ошибки в программе – в общем случае под ошибкой подразумевается неправильность, погрешность или неумышленное искажение объекта или процесса, что может быть ***причиной ущерба*** 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, предъявляемые к сопровождаемым 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. Различие 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. Тем самым проявляется объективная реальность, заключающаяся в невозможности абсолютной корректности и полноты исходных спецификаций и эталонов для проектов ПС.

На практике в процессе сопровождения 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 не корректность таких эталонов

еще труднее, чем обнаружить дефекты в сопровождаемых программах, так как принципиально отсутствуют формализованные данные, которые можно использовать как исходные.

В процессе сопровождения ПС его исходная спецификация подвергается декомпозиции на спецификации программных и информационных компонентов. Эти спецификации рассматриваются как частные эталоны для составных частей ПС, однако они редко бывают абсолютно полными и корректными. 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 создания гаммы методов и средств тестирования для выявления некорректностей в спецификациях на компоненты разных уровней (см. п. 3.4).

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, т.е. последствия и результаты проявления некоторых внутренних дефектов или некорректностей программ. Эти внутренние ***дефекты*** следует квалифицировать как ***первичные*** ошибки или причины обнаруженных аномалий результатов [21]. Последующая локализация и корректировка таких первичных ошибок должна приводить к устранению ошибок, первоначально обнаруживаемых в результатах функционирования программ (рис. 1.2).

Потери эффективности 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 последствия применения комплекса программ.

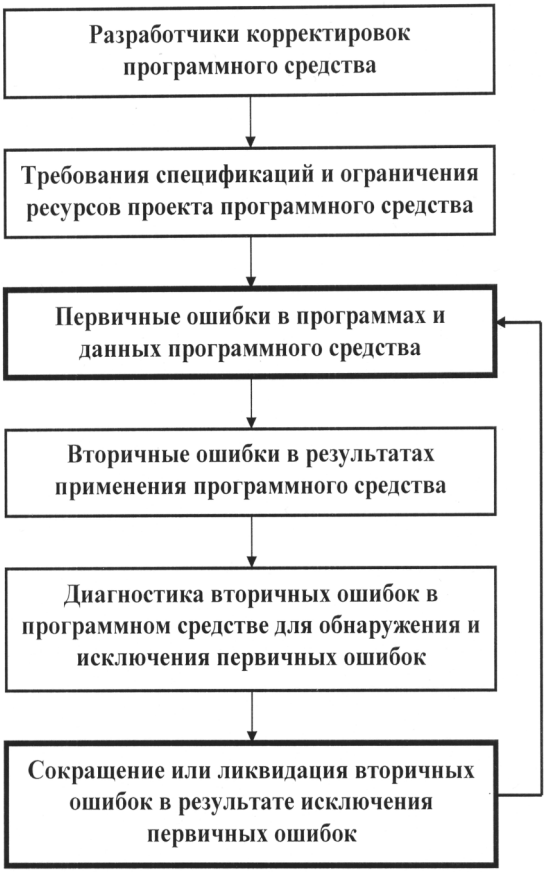

Рис. 1.2

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 вторичные ошибки и функционирование программ, может, ***в принципе***, производиться по значениям ущерба вследствие не устраненных 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 не обнаруженным 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 не велика, и ее трудно выделить из процесса проектирования изменений ПС. Интенсивность проявления и обнаружения вторичных ошибок наиболее велика на этапе активного тестирования и автономной отладки корректировок 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 модификаций версий ПС. Уровень серьезности последствий ошибок варьирует от классов проектов и от предприятия, но, в общем, можно разделить ошибки при модификации ПС 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 обнаружатся. Небольшие ошибки могут включать орфографические ошибки на экране, пропущенные разделы в справочнике и другие мелкие проблемы. Такие ошибки никогда не помешают выпуску и применению модифицированной версии системы. По десятибалльной шкале рисков небольшие ошибки находятся в пределах от 1 до 3 приоритета (см. п. 3.3).

Умеренными ошибками называют те, которые влияют на конечного пользователя, но имеются слабые последствия или обходные пути, позволяющие сохранить достаточную функцио-

нальность 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 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 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. 1.3. 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 корректировок программ и данных полезно использовать ***как ориентир*** для учета необходимых ресурсов при 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 1). Такую корреляцию целесообразно рассматривать и учитывать как общую качественную тенденцию при анализе и поиске их причин. Кроме того, от величины и важности корректировок зависят права специалистов на разработку и реализацию контрмер для их реализации (таблица 2).

Одной из основных причин изменений при сопровождении 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. 1.3).

Таблица 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
Управление сопровождением и конфигурацией, инструкторы интерфейсов	Ошибки проектирования и реализации версий программного продукта
Документаторы	Дефекты и ошибки обобщающих документов

Ошибки и дефекты данного типа появляются из-за недостаточного понимания коллективом специалистов технологии про-

цесса сопровождения ПС, а также вследствие отсутствия четкой его организации и поэтапного контроля качества продуктов и изменений.

Таблица 2

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

Это порождается пренебрежением руководителей к организации всего технологического процесса сопровождения 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 не выявленным 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 модификаций 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 её 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 дефектов модификаций.

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 отладки и текущий этап сопровождения и модификации комплекса программ;
- класс ПС, масштаб (размер) и типы компонентов, в которых обнаруживаются ошибки;
- методы, виды и уровень автоматизации тестирования, их адекватность характеристикам отлаживаемых компонентов и потенциально имеющимся в программах ошибкам;
- виды и достоверность эталонов, которые используются для обнаружения ошибок.

Первичные ошибки в ПС в порядке уменьшения их влия-

ния на сложность обнаружения и масштабы корректировок можно разделить на следующие группы [21, 39]:

- ошибки, обусловленные сложностью сопровождаемых компонентов и ПС в целом и наиболее сильно влияющие на модификации;
- ошибки вследствие большого масштаба – размера комплекса программ, а также высоких требований к его качеству (см. п. 3.1);
- ошибки планирования и корректности требований модификаций часто могут быть наиболее критичным для общего успеха сопровождения 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 модификаций функциональных задач;
- ошибки реализации спецификаций изменений – программные дефекты, возможно, ошибки нарушения требований или структуры 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 конкретизируются и становятся измеримой, когда устанавливается связь этого понятия с конкретными ресурсами, необходимыми для решения соответствующей задачи и возможными проявлениями дефектов. При 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 сопровождения можно разделить на ***две большие группы***:

- ***сложность ошибок при создании корректировок*** комплекса программ для применения – статическая сложность, когда реализуются его требуемые новые 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 и качестве применения версии ПС.

К группе факторов, влияющих на ***сложность ошибок при сопровождении*** комплексов программ, относятся:

- величина – размер модифицируемой программы, выраженная числом строк текста, функциональных точек или количеством программных модулей в комплексе (см. п. 3.1);
- 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 сопровождении комплекса программ.

Некоторые из перечисленных параметров коррелированны 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 при сопровождении*** комплексов программ целесообразно анализировать на базе трех наиболее специфических компонентов:

- ***сложность ошибок изменяемых программных компонентов и модулей*** определяется конструктивной сложно-

стью модификации оформленного 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*** при сопровождении ПС. Качество откорректированного ПС характеризуется многими показателями, состав которых зависит от класса и конкретного назначения комплекса программ (см. п. 3.3) [4, 24, 41]. Ниже в этом разделе 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 при сопровождении сложных и сверхсложных ПС всегда велики проявления не устраненных ошибок и недостаточна достоверность оценок достигнутого качества.

Ошибки корректности формирования и планирования выполнения требований к изменениям ПС в процессе сопровождения проекта часто считаются наиболее критичным для общего успеха совершенствования версий ПС и системы. Ошибки требований являются наиболее трудными для обнару-

жения и наиболее сложными для исправления. Вот почему исправление ошибок требований на модификации может быть в 15 – 70 раз дороже, чем ошибок их программирования [39]. Требование к изменению может быть пропущено в спецификации к системе и ПС. Это ведет к неудовлетворенности пользователя, и программа считается заказчиком и пользователем ошибочной. Пропуск требования части изменения – 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 корректировок ПС определяются процессами перевода неопределенных и общих положений, сделанных на стадии спецификаций требований модификаций, 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 [39].

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, конст-

руктивных 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 изменений структурного уровня. Они проявляются, когда требования интерпретируются неправильно, по крайней мере, с точки зрения конечного пользователя. Если разработчик структуры либо неверно прочитает требования, либо не увидит содержание требования также как конечный пользователь, появится ошибка разработки структуры данного компонента или ПС.

Системные ошибки изменений в ПС определяются, прежде всего, неполной информацией о реальных процессах, происходящих в источниках и потребителях информации [21]. 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 сопровождения не всегда удается точно и полно сформулировать целевую задачу модификации всей системы, а также целевые задачи основных групп программ, и эти задачи уточняются в процессе проектирования изменений. 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 абонентов, принятые в качестве исходных данных в процессе разработки модификаций алгоритмов, могут являться результатом аналитических расчетов, моделирования или исследования аналогичных систем. Во всех случаях отсутствует полная адекватность условий получения предполагаемых и реальных характеристик внешней среды, что может являться причиной сложных и трудно обнаруживаемых ошибок. Это усугубляется тем, что очень часто невозможно заранее предусмотреть все разнообразие возможных внешних ус-

ловий и реальных вариантов сценариев функционирования и применения модифицированных версий ПС.

При автономной и в начале комплексной отладки модифицированных версий, 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 ошибок). Следует также отметить большое количество команд, корректируемых при исправлении каждой такой ошибки (около 20 – 50 команд на одну ошибку) [20].

Алгоритмические ошибки корректировок 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 измененных программ. К алгоритмическим ошибкам следует отнести, прежде всего, ошибки, обусловленные некорректной постановкой требований к модификациям функциональных задач, 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 не полным 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 передается информация и управление между измененными, взаимодействующими модулями. Этот вид ошибок составляет около 10% от общего количества, и 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, 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 [21]. 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 на изменения для решения каждой из них. В результате возникает либо не достаточное 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 изменений 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. табл. 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. Однако некоторые виды ошибок

реализации характерны для среды, в которой они возникли [20, 39].

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 спецификации.

Программные ошибки модифицированных компонентов по количеству и типам в первую очередь определяются степенью автоматизации программирования и глубиной статического контроля текстов изменений 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 изменений 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 измененных 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 модификации 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 неудовлетворен 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 [20]. 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 опубликованном примере анализа ошибок конкретного крупного проекта [41], было принято, что завершилась инспекция начального запрограммированного кода крупного ПС на предмет его соответствия рабочей проектной спецификации, в ходе которой было обнаружено 3,48 ошибок на тысячу строк кода. Наибольшее совпадение аппроксимации рэлеевской кривой распределения ошибок [4] с фактическими данными установлено для момента получения этих данных, ему соответствует значение, равное также 3,48. Значения числа ошибок 3,3 и 7,8 на тысячу строк получены, при расчетах на более ранние этапы соответственно эскизного и рабочего 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 ошибок на тысячу строк исходного кода. В случае сохранения той же закономерности, в момент поставки клиенту на испытания, программный продукт мог содержать менее 0,07 ошибок на тысячу строк кода. Отмечается также, что частость проявления 0,1 – 0,05 ошибок на тысячу строк кода, можно считать допустимыми для ответственных систем реального времени.

В [41] приведены результаты исследования 20 крупных поставляемых программных продуктов, созданных в 13 различных организациях. В этих продуктах 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 ошибок 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 Шатла [39], созданный NASA. По оценке авторов в нем содержится менее одной ошибки на 10000 строк кода. Однако стоимость программного продукта достигает 1000 $ за строку кода, что в среднем в сто раз больше, чем для адми-

нистративных систем и в десять раз больше, чем – для ряда критических систем реального времени.

Приведенные характеристики типов дефектов и количественные данные могут служить ***ориентирами при прогнозировании возможного наличия не выявленных ошибок*** при сопровождении 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. Развитие технологии сопровождения сложных ПС и средств автоматизации проектирования модификаций приводит к изменению ***уровня и интенсивности устранения ошибок корректировок ПС***. Подобные оценки гарантируют ***от избыточного оптимизма*** при определении сроков и при разработке графиков сопровождения и реализации модификаций программ с заданным качеством. Непредсказуемость конкретных ошибок в программах приводит к целесообразности последовательного, методичного фиксирования и анализа возможности проявления любого типа дефектов и необходимости их исключения на наиболее ранних этапах корректировок при минимальных затратах.

1.3. Организация и методы сопровождения программных средств

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.

Целью сопровождения является 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. 1.4.


Цели и состав процессов сопровождения программных средств
Причины и виды изменения программных средств при сопровождении
Организация процессов и передача на сопровождение программного средства
Заключение договора между заказчиком и исполнителем на сопровождение программного средства
Разработка спецификаций требований на модификации при сопровождении программного средства
Разработка концепции методов и процессов сопровождения программного продукта
Оценка и распределение ресурсов, финансов и специалистов для сопровождения программного продукта
Разработка требований к обеспечению качества при сопровождении программного продукта
Утверждение заказчиком концепции, договора и технического задания на сопровождение программного продукта
Организация контроля реализации концепции и договора на сопровождение программного продукта

Рис. 1.3

Основная задача – изменить и улучшить существующий программный продукт, сохраняя его целостность и функциональную пригодность [8, 40, 57].

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. Сопровождение составляют процессы регламентированного изменения программных средств и их компонентов, которые реализуются методами конфигурационного управления.

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 работу ***подготовка*** 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**.

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 определить 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 со-

провождения 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.

Реализация процесса сопровождения в жизненном цикле ПС начинается с планирования сопровождения и завершается снятием данного программного продукта с эксплуатации. Данный процесс заключается в модификации (изменении) текста программы и соответствующих документов, вследствие обнаруженных дефектов или необходимости их усовершенствования. Если отсутствует исходный текст (код) комплекса программ, то есть программный продукт состоит только из готовых к использованию компонентов, все равно может потребоваться его сопровождение. Сопровождение совокупности готовых программных продуктов заказчиком или поставщиком обычно связано с модификацией соответствующих внешних интерфейсов объединенного программного продукта, включая данные и режимы функционирования.

Стоимость процесса сопровождения может составлять значительную (даже наибольшую) часть стоимости жизненного цикла программного продукта (см. Введение). 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 [18].

Характеристики, описывающие качественные и количественные требования к сопровождаемости программного средства, устанавливает заказчик. Качественные требования применяются для описания методик, позволяющих сократить стоимость сопровождения и количество используемых при этом ресурсов. Количественные требования применяют для описания размеров изменяемых функций или критериев качества и характеристик, с помощью которых определяются соответствующие значения или признаки реализации сопровождения в течение жизненного цикла ПС. Эффективность таких усилий, предпринятых при разработке, становится очевидной после начала работ по сопровождению.

При реализации процессов разработки, эксплуатации и сопровождения 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. В ходе работы по контролю конфигурации из процесса УК должен быть решен вопрос о принятии конкретного предложения на изменение.

Корректирующее сопровождение связано с изменениями, вызванными необходимостью устранения фактических ошибок и дефектов в программном продукте. Корректирующее сопровождение проводят в случае выявления несоответствия программного продукта установленным требованиям. Этот вид изменений по причинам и источникам является непредсказуемым и его трудно планировать и регламентировать. При групповой работе специалистов, кроме личных ошибок программистов в модулях, они могут быть обусловлены некорректной интеграцией компонентов программ при наличии нескольких версий модулей, применением дублирующих или конфликтующих друг

с другом копий модулей и различными дефектами взаимодействия программных компонентов между собой и с данными. Остальные виды корректировок носят упорядоченный характер и проводятся в соответствии с заранее подготовленными планами и документами. Поэтому изменения, обусловленные ошибками и выявленными дефектами, в большинстве случаев целесообразно накапливать и реализовывать, приурочивая к изменениям, регламентированным модернизациями. Однако некоторые ошибки могут приводить к необходимости срочного исправления программ.

Изменения, при модернизирующем сопровождении – адаптивном и полном, вносятся сверх технических и функциональных требований спецификаций, установленных при первичном проектировании или выпуске предыдущей, базовой версии программного продукта. Изменения, вносимые при адаптивном сопровождении, связаны с необходимостью приспособления программного продукта к изменившейся внешней среде или требований пользователей. Данные изменения могут быть связаны с реализацией новых требований к системному интерфейсу, самой системе или техническим средствам. Изменения, вносимые при полном – модернизирующем сопровождении, улучшают функциональные характеристики ПС и его сопровождаемость. Такие изменения могут приводить к предоставлению пользователям новых функциональных возможностей, пересмотру технологии применения сопровождаемых документов или к изменению самих документов. Сопровождение программного средства, необходимое для изменения структуры или системы, то есть модификации ПС, вносятся в существующую архитектуру в рамках ограничений, установленных первичной структурой проекта. Таким образом, модернизации продукта при адаптивном и полном сопровождении зачастую наиболее дорогостоящие и требуют больших временных затрат.

Заказчик может заключить соглашение с разработчиком базовой версии ПС об ***организации сопровождения*** или выбрать в качестве сопроводителя третью сторону (помимо разработчика) (см. рис. 1.4). Сопровождение может также быть проведено по соглашению между двумя сторонами внутри одного предприятия. В стандарте **ISO 14476** описаны задачи, рекомендуемые при заключении соглашения между заказчиком и поставщиком.

Эти положения должны быть использованы независимо от того, принадлежит ли заказчик или поставщик к одному или к разным предприятиям.

Если заказчику необходимо вести сопровождение ПС силами разработчика после поставки данного средства или по окончании гарантийного периода, это должно быть указано в соответствующем соглашении. Поставка модернизированных документов должна быть предусмотрена в соответствующем соглашении. Также должно быть оговорено обучение соответствующего персонала пользователей. Поставщик должен подготовить процедуры выполнения каждой задачи сопровождения, выполнять эти процедуры во время сопровождения и проверять соответствие конкретных работ договорным требованиям и установленным процедурам. Использование опытных данных по конкретным процедурам повышает эффективность их применения. В плане сопровождения должны быть указаны объекты сопровождения, процедуры сопровождения и период сопровождения каждого объекта.

Передача программного средства для сопровождения является контролируемой и координируемой последовательностью действий, при выполнении которых разработанное ПС 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 средства: комплектование персонала, его обучение, ввод в действие версий программного продукта, распространение опыта по сопровождению.

Сопроводители иногда сталкиваются 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. Имеется ряд факторов, влияющих на создание или обновление документов, некоторыми из них являются доступ к исходным программам, наличие инструментальных средств анализа программ и наличие среды тестирования программного средства.

Сопроводитель и заказчик должны ***заключить договор на сопровождение*** и указать в нем возможные процедуры внесения изменений в сопровождаемые программные продукты (см. рис.1.3). Процедуры могут быть использованы как разработчиком оригинала, базовой версии ПС, так и независимым сопроводителем и ***охватывать*** [41, 63]:

- 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 (стандарт **ISO 12207**), требования к которому ***дополнены***:

- установленными и документально оформленными критериями проведения испытаний, оценки их результатов, а также оценки измененных и неизмененных объектов (программных модулей, компонентов и элементов конфигурации) ПС;
- полнотой и правильностью реализации новых и измененных требований, чтобы исходные, неизмененные требования к ПС, 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 **ISO 12207** и **ISO 9126**. При этом должны быть учтены следующие ***факторы, влияющие на сопровождаемость***:

- определение и описание новых функций;
- точность и логическая организация данных;
- интерфейсы (системные, машинные и пользователей), особенно новые и перспективные интерфейсы;
- требования к функциям и рабочим характеристикам, включая влияния корректировок и дополнений;
- требования, налагаемые запланированной внешней средой;

- неоднородность требований, определяющая простоту или сложность их трассировки и прослеживания;
- план обеспечения качества модифицированного программного средства, в котором особое внимание должно быть уделено документам на изменения и их согласованность.

Разработку концепции сопровождения в соответствии со стандартом **ISO 14764,** целесообразно начинать с формализации и обоснования набора исходных данных, отражающих общие особенности класса, назначения и функции ПС, потребителей и этапов жизненного цикла проекта ПС, каждый из которых влияет на выбор определенных характеристик изменения комплекса программ (см. рис.1.4). 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 ПС, 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 ПС 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*** ПС 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:

- корректировка, модернизация, профилактика или адаптация к новым условиям;
- размер изменения, стоимость, время на реализацию изменения;
- критичность, влияние на 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 программного средства. Она должна быть разработана, при первом выпуске исходного ***программного продукта и отражать***:

- область сопровождения и изменений программного средства;

- практическое применение (адаптацию) данного процесса;
- определение предприятия и лиц, ответственных за сопровождение;
- оценку стоимости и длительности сопровождения.

Область сопровождения должна представлять 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 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 юридических), 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 (аутсорсинг) 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. В качестве исходных данных при оценке стоимости сопровождения могут быть использованы соответствующие архивные данные по аналогичным проектам.

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 её назначения, функций и возможностей реализации при доступных ресурсах. Чем крупнее и сложней 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 завышение) требований к некоторым характеристикам модифицированного ПС. Кроме того, возможно, что некоторые характеристики или их изменения противоречивы или принципиально нереализуемы в данном проекте. В результате ***не сбалансированные 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 про-

граммного 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.

Требования к функциональной пригодности должны выполняться для любых классов и назначения ПС. Номенклатура учитываемых требований к конструктивным характеристикам качества существенно зависит от назначения и функций комплексов программ. После определения назначения и функций ПС подготовка исходных данных и концепции сопровождения должны завершаться выделением номенклатуры изменений приоритетных конструктивных характеристик, имеющих достаточно сильное влияние на функциональную пригодность ПС [5, 19].

На этапе проектирования концепции модификаций, после фиксирования исходных данных конкретного изменения, начинаются выбор требований к его свойствам, а также установление и утверждение конкретных характеристик качества модификаций. Такой анализ должны проводить заказчик и некоторые потенциальные пользователи совместно со специалистами, обеспечивающими ЖЦ комплекса программ и реализацию установленных требований к показателям качества при сопровождении. 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 программном продукте.

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. Требования к значениям изменений характеристик ПС должны быть предварительно проверены разработчиками на их реализуемость с учетом доступных ресурсов конкретного проекта и при необходимости откорректированы по составу и значениям. 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 проектирования модификаций (см. рис. 1.3). 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. 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 реализации ПС.

При детальном проектировании изменений может быть целесообразен ***контроль реализации концепции и договора*** и уточнение совокупности функциональных характеристик и качества модифицированных версий сложных ПС, а также соотношения качества и затрат ресурсов на изменения. 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 кон-

троля 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.

1.4. Этапы и процедуры при сопровождении программных средств

В соответствии с требованиями по развитию и модификации программного продукта в жизненном цикле ***должен быть организован процесс его сопровождения*** (см. п. 5.5 стандарта **ISO 12207**). 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. Процессы и этапы сопровождения охватывают: подготовку процесса; анализ дефектов и модификаций; внесение изменений; поставку, проверку и приемку ПС при сопровождении; перенос на иные платформы. Они завершаются окончательным снятием программного продукта с эксплуатации. Исходные данные преобразуют или используют в работах по сопровождению для получения выходных результатов – ***модифицированных версий программного продукта***. Рекомендуется проводить регулярный контроль с целью проверки корректности выходных результатов конкретных работ по сопровождению. Для обеспечения работ по сопровождению рекомендуется использовать вспомогательные и организационные процессы по стандартам **ISO 12207** и **ISO 15504**.

При ***подготовке процесса*** сопроводитель должен создать планы и определить процедуры, выполняемые при реализации сопровождения (рис. 1.5). План сопровождения целесообразно создавать параллельно с планом разработки первой, базовой версии ПС. При выполнении данной работы 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 про-

дукта; 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 многих видов 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.

Рис. 1.4

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.

Стратегии сопровождения должны быть ориентированы на людские и материальные ресурсы, необходимые и доступные для обеспечения развития и модификаций программного продукта. В качестве целей при планировании сопровождения следует использовать результаты анализа сопровождаемости проекта. Этот анализ является исходным при разработке стратегии сопровождения. Политика сопровождения ПС должна охватывать следующие основные компоненты: концепцию сопровождения; план сопровождения; анализ ресурсов (см. п. 1.3). Процесс разработки изменений включает в себя ряд работ, связанных с планированием сопровождения 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;
- использование пакетов изменений, облегчающее их внесение в ПС с минимальной корректировкой графика работ и обеспечивающее максимальную целостность и стабильность данного программного продукта;
- соглашения по интерфейсам для всего проекта в части постоянных проблем, связанных с неясностью, неточностью,

изменчивостью или не проверяемостью 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 (см. п. 1.3). Этот план должен быть подготовлен сопроводителем во время первичной разработки базовой версии ПС и включать в себя процессы анализа предложений пользователя по внесению изменений в программный продукт. ***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;
- место проведения сопровождения;
- время начала и длительность сопровождения.

Стандартом **ISO 14764** рекомендуется сопроводителям формализовать ***конкретный план сопровождения ПС*** из представленного общего состава процессов ЖЦ, который уточнить и

адаптировать с учетом объема и особенностей проекта и содержащим разделы (подробнее см. п.2.3):

- 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 в плане трансформируется в его общую структуру и 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 планировании и выборе ***языка программирования*** для разработки модификаций целесообразно учитывать следующие факторы: стабильность языка; возможности структурирования программ; простоту создания новых редакций (версий); возможности структурирования данных; наличие и стабильность компилятора и других инструментальных средств разработки измене-

ний ПС; долговечность различных инструментальных средств разработки.

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** (см. п. 3.3). При программировании и тестировании ПС создаются, документируются и тестируются программные модули и базы данных. Сопровождаемость ПС может быть повышена, благодаря совершенствованию качества документов. Документы компонентов должны содержать информацию, способную существенно помочь при выполнении процесса сопровождения. Для улучшения сопровождаемости целесообразно:

- обеспечивать удобочитаемость документов;
- избегать неструктурированных программных кодов в компонентах;
- выявлять и предупреждать ошибки в спецификациях компонента;
- использовать методы, облегчающие выявление ошибок в документах.

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, должна быть документально оформлена и введена 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.

Основой для проведения работы по анализу дефектов и изменений являются: официальное предложение о модификации или отчет о дефекте, системные и (или) проектные документы и нормативные документы. Исходные данные для проведения работы включают: предложения о модификациях или отчеты о дефектах; последняя базовая версия программного продукта; информационный архив проекта программного средства; информацию о состояниях конфигурации ПС, БД и системы; функциональные требования; требования к интерфейсам.

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;
- масштабу: размеру изменения, стоимости, времени на реализацию изменения;

- 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 реализуемые 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 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;
- реализовать управление конфигурацией представленной версией программного продукта;
- ввести в действие (инсталлировать) представленную версию программного продукта;
- провести тестирование для проверки проблемы – дефекта, предпочтительно с использованием копий представленных тестовых данных;

- документально оформить результаты квалификационного тестирования.

Если конкретная проблема не может быть повторена сопроводителем, он должен проверить правила, политики и документы 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 (см. п. 3.3).

Сопроводитель должен реализовать ***процесс управления конфигурацией*** для управления изменениями существующей системы или определить организационный интерфейс с данным процессом. При выполнении работы по подготовке процесса сопровождения используют следующие ***вспомогательные и организационные процессы*** жизненного цикла ПС (см. стандарт **ISO 12207** п.5 и п.6): документирование; управление конфигурацией; обеспечение качества; совместный анализа; управление проектом; создание инфраструктуры; обучение. 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. Все результаты изменений должны быть охвачены управлением их конфигурацией (подробнее см. главу 2 и стандарт **ISO 15846**).

До внесения изменений в систему и программный продукт в соответствии с договором с заказчиком, сопроводитель должен ***согласовать выбранный вариант корректировки***, выполнив следующее:

- представить результаты анализов на согласование в соответствующие группы Совета по управлению конфигурацией;

- участвовать в обсуждениях с заказчиком рассматриваемой корректировки;
- обновить, после согласования, состояние, содержание и статус предложения о модификации;
- обновить, после согласования, конкретные требования к программному продукту, если соответствующая заявка на корректировку носит характер модернизации и совершенствования объекта.

Контроль за рассматриваемыми работами следует проводить посредством процесса ***совместного анализа*** (см. стандарт **ISO 14764**). 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. В анализе влияния корректировки должны быть отражены:

- формулировка и оценка проблемы – дефекта или нового требования;
- классификация типа требуемых процедур сопровождения;
- начальный приоритет реализации рассматриваемой задачи;
- дата проверки для вносимых изменений;
- начальная оценка ресурсов, необходимых для модификации существующей системы;
- обновленные документы по тестированию, включая план, процедуры и отчеты о тестировании;
- комплект документов результатов разработки модифицированного программного продукта;
- обновленные спецификации требования к ПС.

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 (см. стандарт **ISO 12207)**. После согласования корректировки сопроводителю следует провести анализ и определить, какие документы, программные модули и их версии требуют изменения. Результаты такого дополнительного анализа должны быть документально 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 результатов разработки базовой версии программного продукта.

Требования к стандартизированным процессам разработки ПС для реализации изменений целесообразно дополнить процедурами:

- 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, неизменные 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 полноты 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 изменениях версий ПС. 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. Трудности, как правило, усугублялись тем, что новая технология сопровождения обычно внедряется параллельно с использованием старых, а также тем, что степень отработка новой версии ПС не позволяет гарантировать отсутствие дефектов и ошибок. 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 (подробнее см. п. 2.3):

- 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 эксплуатационного тестирования;
- результаты аудиторских проверок версии программного про-дукта;
- 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;
- правильность оценок необходимых и использованных ресурсов;
- официальные рекомендации с указаниями о целесообразных последующих модификациях и создании новых версий ПС.

Внедрение новой версий программного продукта для массового применения (см. рис. 1.4) 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 по новой технологии. Тем не менее, известны примеры попыток подключения новой технологии к проектированию модификаций ПС с определенным ***унаследованным заделом***. Обычно такая

ситуация характерна для сложных, эксплуатируемых ПС, подвергающихся серьезной модернизации или развитию, либо когда сроки разработки ПС 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 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. Разработка нового ПС может проводиться для: усовершенствования модульности; упрощения сопровождения; обеспечения стандартизации или независимости от поставщика-продавца компонентов. 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 пользователей об этом, провести соответствующее обучение персонала, уведомить всех заинтересованных субъектов о завершении сопровождения и архивировать соответствующие данные. Должен быть разработан, документально оформлен и реализован план снятия с эксплуатации и прекра-

щения активной поддержки ПС сопровождающим предприятием. К запланированным работам целесообразно привлечь пользователей. 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.

Кроме того, сопроводитель должен определить все объекты, затрагиваемые при данной работе, определить специфику каждого абонента и отработать обратную связь с абонентами – пользователями.

Перед прекращением сопровождения следует определить влияние снятия программного продукта с сопровождения на пользователей, установить программный продукт, заменяющий снимаемый (при его наличии) и определить обязанности по любым оставшимся вопросам последующей поддержки применения ПС. Пользователи должны получить уведомление о планах и работах по снятию с сопровождения и эксплуатации. В содержание уведомления необходимо включить: описание заменяющего или модернизированного ПС с указанием даты его доступности для пользователей; объяснение того, почему прежний программный продукт нельзя больше поддерживать и применять; описание других доступных вариантов поддержки в случае прекращения сопровождения прежнего ПС.

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. Всё, 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. Результатами данной работы должны быть: план снятия с эксплуатации; уведомление о намерениях по снятию с эксплуатации; результаты, полученные при выполнении снятия программного средства с эксплуатации; обученный персонал пользователей; архивированная базовая версия ПС.

1.5. 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 необходима 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 аналогичных ПС и БД на новой платформе. Использование методического, техно-

логического, алгоритмического и программного задела из предшествующих проектов обеспечивает многократное повышение производительности труда разработчиков информационных систем, сокращение сроков их создания и высокое качество проектов. Под ***мобильностью – переносимостью*** далее понимается [22, 32, 57]:

- 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 ИС;
- 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 ИС.

Основным стимулом развития и применения мобильных программ и данных явилась необходимость улучшения экономических показателей при создании и эксплуатации сложных информационных 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 средства 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 мини-

мальными изменениями на различные операционные системы и аппаратные платформы;

- экономная реализация совместной работы и расширения функций ПС и БД (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 информационным системам (мобильность пользователей);
- снижения зависимости заказчиков конкретных информационных систем от определенных поставщиков аппаратных и операционных платформ, а также от разработчиков некоторых программных продуктов.

Для достижения перечисленных целей в обеспечении мобильности ПС и БД требуются различные ресурсы при их реализации. Потребность в конкретных ресурсах и рентабельность их использования зависит 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, обеспечивающих их эффективную мобильность на определенные типы платформ, например, в соответствии с концепцией и стандартами открытых систем (см. гл. 4);
- 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.

При анализе эффективности переноса приложений на различные аппаратные платформы следует учитывать две противоречивые тенденции их развития. ***Первая тенденция*** состоит в унификации и системной преемственности аппаратных платформ с последовательным развитием гаммы моделей ЭВМ, со-

храняющих возможность непосредственного использования ранее созданных ПС и БД на новых, более совершенных моделях. Одновременно развиваются ряды операционных систем (ОС), в которых расширяются функции, но сохраняются унифицированные интерфейсы с приложениями и внешней средой. Подобная унификация обеспечивает относительно простой перенос программ и данных на новые модели ЭВМ, с унифицированными ОС. Эта тенденция поддерживается созданием стандартов де-факто на операционные системы и де-юре на интерфейсы с ними комплексов программ и данных.

Вторая тенденция состоит в создании несовместимых, полностью новых аппаратных платформ и разнообразных переносимых операционных систем, реализуемых на различных платформах. При унифицированной ОС на различных аппаратных платформах перенос ПС и информации БД может ограничивать эффективность их функционирования и применения. При этом выбор конкретных ПС и БД зачастую диктуется имеющейся унифицированной ОС и ограничивает возможности пользователей. В некоторых случаях на применяемых аппаратных платформах размещаются разные типы ОС, и для переноса ПС и БД вместе с ними приходится проводить перенос ОС. Таким образом, появилась и развивается проблема выбора и ***обеспечения переноса операционных систем*** на различные платформы.

Переносимые ОС освобождают пользователей от необходимости применения ПС и БД только определенной фирмы и позволяют выбирать наиболее эффективные сочетания типов ОС, управляемых ими ПС, и аппаратных платформ. Однако при этом значительно сложней обеспечить системную интеграцию аппаратуры, ПС и переносимых ОС, чем при унифицированных средствах одной фирмы. Поэтому переход на свободно переносимые ОС, выбираемые пользователем, применительно к решаемым задачам на различных аппаратных платформах, не только сулит гибкость построения информационных систем, но и порождает сложные проблемы системного взаимодействия. Возможность выбора свободно переносимых ОС стимулирует конкуренцию и борьбу за их качество и разнообразие функций, однако связана с серьезными трудностями организации взаимодействия ПС с периферийным оборудованием, в сетях и с различными конечными пользователями. При возрастании сложности ИС и числа, ис-

пользуемых в них компонентов выбор применяемой ОС становится все более консервативным и однозначным.

Задачи повторного использования и переноса программ и данных охватывают:

- встраивание готового программного средства и информации базы данных в создаваемую новую систему при условии, что поставщики ПС и БД гарантируют их 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.

В общем случае невозможна разработка ПС и БД, ***эффективно переносимых на любые ЭВМ***, в различные операционные системы и для любой внешней среды. При переносе свойства ИС практически всегда изменяются, что следует учитывать при анализе целесообразности переноса, а также могут быть необходимы отладка, испытания и сертификация получаемых ПС и БД в новой среде [22, 32, 55]. 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 ПС и БД в новом окружении или на новой платформе;
- сертификации перенесенных ПС и БД на новую платформу и функционирующих 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 пере-***

нос, так как при создании мобильных ПС и БД 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 ИС. Анализ мобильности программ и данных в системах касается различных классов задач, однако далее, для определенности, конкретные методы и средства обеспечения мобильности рассматриваются преимущественно при-менительно 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.

Анализируя переносимость комплексов программ с исходной аппаратно-операционной платформы на новую, следует иметь в виду различные аспекты мобильности приложения: собственно программы, данные, пользовательское окружение и документацию. В зависимости от условий, на которые ориентируется ***проект создаваемой системы, различают*** [22, 34]:

- горизонтальную мобильность, т.е. переносимость ПС между компьютерами, аппаратными платформами одного класса, имеющими относительно небольшие архитектурные различия;

- вертикальную мобильность, т.е. переносимость ПС между аппаратно-операционными платформами, существенно, разных классов в распределенных системах;
- переносимость "на будущее" – свойство приложений функционировать в среде открытых систем, в которой могут быть произведены изменения аппаратно-операционных платформ при сохранении интерфейсов прикладных программ с внешней средой.

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;
- сборку или интеграцию компонентов и комплексное, квалификационное тестирование ИС 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*** имеющихся ПС и БД:

- модели предметной области и спецификаций требований, возможно, реализуемые разными способами на этапе проектирования ПС и БД;
- проектные спецификации требований на этапе разработки ПС и БД;
- исходные тексты программ на языках программирования, применявшихся при разработке повторно используемых программных компонентов;
- объектные коды программ, если обеспечена структурная, аппаратная совместимость между исходной и целевой (реализующей) платформами;
- структуры файлов и информации 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 БД 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 и БД для авиационных, ракетных и космических систем, в которых ограничения веса и габаритов вызывают повышенные требования к эффективности использования вычислительных ресурсов [15, 21].

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 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 перекомпилляция 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-технологии и CASE-системы;
- языки баз данных и системы управления базами данных;
- форматы данных, форматы внешних электронных сообщений;
- форматы переносимых электронных документов.

Значительная часть этих задач решается путем применения соответствующих стандартов информационных технологий, действующих как международные или национальные нормативные документы, или открытых спецификаций, отражающих сложившиеся промышленные стандарты "де-факто" (см. гл.4). Для проверки соответствия создаваемых ПС и БД применяемым стандартам, требуется иметь тесты соответствия в составе целевой или CASE-системы, поддерживающей создание комплекса программ. Функционирование программ на выбранных плат-

формах обычно гарантируется поставщиками этих ПС и БД, что должно быть явно отражено в документации поставщика.

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 не рентабельными. 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, решающего все функциональные задачи определенной сложной системы и полностью сохраняющего 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.

Приведенные варианты переноса значительно различаются возможностью выбора технологии, стандартов и инструментария. В первом случае методы и средства переноса почти полностью определяются степенью совместимости аппаратных платформ и мобильностью переносимых систем. При переносе небольших компонентов, имеющаяся среда, куда переносятся программы и данные определяет необходимость использования технологии и инструментария, которые применялись при их создании. Методологически наиболее сложным случаем является перенос дополнительных, функциональных компонентов между несовместимыми аппаратными и операционными платфор-

мами в программную среду, которая уже содержит часть комплекса программ и массивов данных информационной системы.

Проектирование систем ***с использованием повторно применяемых компонентов*** становится ***особенно рентабельным для крупных ПС***, содержащих сотни или тысячи модулей, и с большими объемами обрабатываемой информации. Поэтому ниже анализируются преимущественно крупные разработки объемом в сотни тысяч строк текста. 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 и на той же или различных платформах.

На первый взгляд, повторное использование исходных текстов компонентов на однотипных реализующих ЭВМ на любых языках программирования представляется тривиальным. Однако для обеспечения переноса текстов программ необходимо при их первичной разработке ***подготовить возможность их последующего многократного использования*** в различном операционном и внешнем окружении. Для этого должна быть унифицирована технология разработки модулей и групп программ, унифицированы структуры межмодульных связей и технология комплексирования, стандартизирована система идентификации и специфицирования программ и данных, а также дисциплина испытаний и документирования компонентов. Программные компоненты должны быть подготовлены для отчуждения от их первичных разработчиков и от комплексов программ, в составе которых они первично отлаживались, испытывались и применялись. Таким образом, возникает необходимость проведения ряда общесистемных и организационных работ, а также анализа некоторых дополнительных затрат на обеспечение возможности эффективного переноса программ и данных в различную внешнюю среду.

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. Поэтому далее в основном под переносом программ и данных будет понимать метод их создания в новой аппаратной и операционной среде реализации

и функционирования ***путем прямого использования текстов программ и данных***, реализованных на языке программирования в исходной среде применения.

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.

При выборе методологии, технологии и инструментальных средств целесообразно учитывать ***практические задачи и оценки переносимости*** программ и данных:

- как создавать программы и данные, обладающие свойством рентабельной переносимости в заданных пределах;
- как перенести в создаваемую новую систему существующие программы и данные, которые первоначально создавались, возможно, без учета всего комплекса факторов, влияющих на эффективную переносимость;
- как оценивать эффективность переноса программ и данных при планировании и управлении проектом, учитывая экономическую эффективность (выигрыш от переноса повторно используемых программ и данных и затраты на перенос) и техническую эффективность (выигрыш и потери в функциональных возможностях ПС за счет их переноса с исходных на новые платформы).

Технология создания и применения мобильных комплексов программ и информации баз данных ***быстро совершенствуется и становится доминирующей*** при создании сложных информационных систем. Исследования открытых систем, разработка технологических инструментальных комплексов и их активное применения позволяют повысить качество и конкурентоспособность программных средств и баз данных и резко сократить трудоемкость, стоимость и длительность их создания. Многие сложные проекты в различных областях применения идут

трудно или очень медленно из-за низкой культуры системного проектирования, а также вследствие пренебрежения современными методологиями, технологиями, стандартами и средствами автоматизации разработки мобильных ПС и БД.

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). ***Исходными данными*** для проведения работы по переносу являются:

- старая среда – ОС-1 и ЭВМ-1;
- новая среда – ОС-2 и ЭВМ-2;
- старая базовая версия – ПС-1 и БД-1.

Специалисты, которые проводят перенос по рекомендациям стандарта **ISO 14764**, должны разработать план переноса, известить пользователей, обучить персонала, 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. Сопроводитель должен конкретизировать следующие задачи:

- проанализировать требования заказчика к переносу;
- определить влияние и роль переносимого программного продукта на систему и среду;
- установить график проведения переноса;
- установить требования к сценариям и наборам тестовых данных для проверки корректности результатов выполненного переноса;
- определить и документально оформить процедуры по переносу;
- определить и уменьшить возможный риск при реализации переноса;
- определить необходимые инструментальные средства для переноса;
- определить степень последующей поддержки для старой среды и системы;
- разработать и/или приобрести инструментальные средства для переноса;
- разделить компоненты программного продукта и данных для автоматизации преобразований при переносе;
- установить приоритеты преобразования программных компонентов, продуктов и данных при переносе;
- преобразовать программные продукты и данные при переносе;
- перенести программные продукты и данные в новую среду;
- верифицировать и тестировать результаты переноса;
- провести опытную, параллельную эксплуатацию программного продукта в новой и старой средах;
- проводить последующую поддержку ПС и БД для старой среды.

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 и БД нельзя больше сопровождать и поддерживать;
- описание новой среды и программного продукта с указанием даты, с которой они доступны для заказчика и пользователей;
- описание других доступных вариантов поддержки в случае прекращения сопровождения прежней среды и программного продукта.

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 абонентами – 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. В течение этого периода следует обеспечивать необходимое обучение персонала, по условиям договора. Как часть указанной задачи сопроводитель может:

- провести обследование деятельности пользователей;
- установить соответствующее дополнительное оборудование;
- установить новые компоненты и программные продукты;
- провести предварительные испытания для проверки правильности установки новых технических и программных средств;
- протестировать программные средства при рабочей загрузке параллельно со старой системой;
- собрать и сравнить данные о результатах функционирования новых и старых программных продуктов.

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 программы следует поместить в архивы. Как часть этой задачи, сопроводитель должен:

- опубликовать изменения к графику (Программе) переноса;
- документально зафиксировать специфику пользователей и соответствующие решения по их обучению;
- архивировать старые, исходные программные продукты и данные;
- снять старое оборудование (при необходимости) с эксплуатации.

После завершения переноса целесообразно провести ***итоговый анализ*** для оценки влияния перехода к новой аппаратно-операцион-ной среде на различные аспекты эксплуатации перенесенного программного продукта. Результаты анализа должны быть разосланы соответствующим заинтересованным сторонам для информации, руководства и использования в работе. Как часть указанной задачи сопроводитель должен:

- проанализировать результаты параллельной эксплуатации систем, ПС и БД;
- определить области потенциального риска применения новых, перенесенных ПС и БД;
- определить специфику новых пользователей;
- документально зафиксировать извлеченные уроки от переноса;
- создать и опубликовать отчет по анализу влияния переноса на применение программного продукта.

Данные, использованные или связанные с прежней средой, должны быть ***доступными для защиты и аудиторской проверки*** в соответствии с условиями договора. Как часть указанной задачи следует:

- сохранить старые программные продукты и данные;
- создать копии старых программных продуктов и данных;
- гарантировано хранить соответствующие носители в безопасном месте.

Контроль за результатами работы по переносу следует проводить посредством процесса совместного анализа. При выполнении работ по переносу необходимо использовать вспомогательные и организационные процессы базового стандарта жизненного цикла ПС **ISO 12207**:

- документирование;
- управление конфигурацией;
- обеспечение качества;
- верификация;
- аттестация (валидация);
- совместный анализ;
- аудит;
- решение проблем – устранение дефектов;
- обучение.

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.

1.6. Факторы, влияющие на эффективность переноса программ и данных на иные платформы

Семейства аппаратных платформ информационных систем быстро совершенствуются по величине доступных ресурсов. В таких семействах перенос приложений с младших на старшие по

ресурсам модели ЭВМ, при сохранении операционной среды требует минимальных затрат, связанных с самими процедурами переноса и минимально необходимой проверкой ПС и БД на новой платформе. При переносе даже сложных критических ИС в этом случае могут потребоваться только дополнительные испытания и корректировка эксплуатационной документации. Однако трудоемкость переноса при таких ситуациях ***вряд ли превысит несколько процентов*** от полной повторной разработки аналогичных ПС или БД. Тем не менее, она может быть достаточно значительной и ее следует учитывать при системном анализе рентабельности проектов ИС.

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.

Значительное влияние на эффективность переноса может оказать ***опыт коллектива специалистов***, создавшего унаследованную систему, и качество документации. Тем не менее, трудоемкость переноса подобных ПС и БД на новую платформу может достигать 80-90% от полной разработки её при старой технологии. Применение современных технологий и средств автоматизации может сделать рентабельным полную разработку новой мобильной системы с учетом имеющегося системного задела и сократить трудоемкость в несколько раз относительно первоначальной.

Программы организации вычислительного процесса и ввода-вывода обычно являются наиболее машинно-зависимыми, что усложняет их перенос между разнотипными ЭВМ. При применении для разработки программ в унаследованных системах языков высокого уровня трудно обеспечить необходимую

эффективность транслируемого объектного кода, а следовательно, высокую скорость их исполнения. Различия операционных систем, принципов организации вычислительного процесса и контроля функционирования, а также драйверов ввода-вывода могут практически полностью исключить эффективный автоматизированный перенос соответствующей части унаследованных программ между разнотипными ЭВМ.

Важнейшим фактором, влияющим на рентабельность переноса, является ***степень унификации интерфейсов*** компонентов ПС и БД с операционной и внешней средой и с пользователями. Применение одних и тех же стандартизированных языков программирования и аттестованных компиляторов позволяет значительно сокращать затраты при переносе. Мобильность текстов программ и данных должны подготавливаться в процессе системной разработки проектов ИС. При этом целесообразно фиксировать класс или типы ЭВМ, между которыми предполагается обеспечивать мобильность программ и данных. В соответствии с синтаксисом и семантикой языка программирования в некоторых случаях может быть полезно формализовать и выполнять дополнительные соглашения, определяющие правила организации последовательных вычислений, ввода-вывода, применения регистров и т.д. При соблюдении этих правил могут быть созданы ПС и БД, эффективно переносимые на уровне идентичных исходных текстов, на языке высокого уровня между разнотипными ЭВМ.

Для обеспечения эффективной мобильности программных компонентов и информации баз данных необходима стандартизация языков проектирования программ и их интерфейсов, оформления и испытания программных модулей, а также гарантии их качества. Стандарты разных уровней должны ***регламентировать технологии*** разработки, поставки, сопровождения и эксплуатации программ, правила структурного построения ПС в целом и их отдельных компонентов, а также массивов данных, состав и формы документации, методы испытаний и определения качества, гарантии поставщика. Это позволяет значительно сокращать дублирующие разработки, внедрять сборочное программирование и вести накопление на предприятиях компонентов высококачественного программного продукта для их многократного использования в качестве типовых комплектующих изделий.

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. главу 4).

При анализе рентабельности переноса программ и данных в первую очередь целесообразно ***учитывать следующие факторы***:

- степень подобия архитектуры и соотношения основных параметров аппаратных платформ, между которыми предполагается перенос;
- степень подобия и унификации интерфейсов операционных платформ, между которыми предполагается перенос комплексов программ и данных;
- наличие или отсутствие ориентации на будущий перенос при разработке исходных программ и информации баз данных;
- объем и комплексированность ПС с операционной и внешней средой при предполагаемом их совместном переносе.

Оценку эффективности создания ПС и БД путем переноса с иных аппаратных платформ можно проводить, ***используя методы***:

- приближенного анализа факторов, влияющих на возможность и целесообразность переноса программ и данных, для исключения заведомо нерентабельных вариантов и выделения ситуаций, когда необходима полная разработка ПС или БД с использованием только системного задела;
- детального анализа и сопоставления возможных технико-экономических показателей и графиков работ при создании ПС по полному циклу разработки, и посредством переноса на иную аппаратную или операционную платформу.

Такая последовательность анализа позволяет предварительно исключать неэффективные варианты и доводить до конкретных реализаций ситуации, в которых создание ПС и БД посредством переноса дает значительный экономический эффект. При

анализе мобильности программ необходимо выявить факторы и параметры программ, среды их разработки и реализации, которые в наибольшей степени влияют на процесс, технологию и эффективность переноса. Эти факторы и параметры иногда трудно измерить количественно и связать непосредственно с эффективностью переноса. Однако, приводимый ниже ***качественный анализ и факторы позволяют ориентироваться в вариантах переноса***, которые могут дать достаточно заметный положительный эффект [15, 20, 51].

Степень машинной ориентировки ПС при оценивании эффективности переноса следует учитывать, так как в некоторые прикладные ПС входят компоненты, непосредственно ориентированные на архитектуру ЭВМ и ее ОС. В результате программные и информационные компоненты сложного прикладного ПС можно разделить на ***машинно-ориентированные и машинно-независимые***. Чем меньше доля машинно-ориентированных компонентов в ПС и чем слабее все компоненты связаны с особенностями характеристик операционной системы, тем рентабельней может быть автоматизированный перенос совокупности программ на другую реализующую ЭВМ. При этом обычно оказывается, что некоторую часть ПС не выгодно непосредственно переносить на новую аппаратную платформу, а целесообразно ее разработать с учетом особенности архитектуры этой ЭВМ.

Некоторые программные и информационные компоненты, разработанные на не стандартизированных языках высокого уровня (ЯВУ) могут иметь особенности, неявно отражающие архитектуру ЭВМ и ОС, для которых они первично создавались. Эти обстоятельства осложняют непосредственный, автоматизированный перенос текстов программ для использования на новой ЭВМ с другой архитектурой и ОС. Поэтому при автоматизированном переносе текстов программ, созданных на ЯВУ, следует учитывать возможность и необходимость некоторой доработки текстов. Доля возможных ручных работ специалистов по доработке исходных текстов, их трансляции и последующего контроля, значительно влияют на целесообразность и эффективность переноса программ и данных.

Различие архитектуры и ресурсов ЭВМ, между которыми предполагается перенос программ и данных, является важнейшим фактором, определяющим эффективность переноса. Значи-

тельные трудности может вызывать или делать практически не рентабельным перенос ПС, существенное различие производительности и памяти рассматриваемых ЭВМ. Если ресурсы ЭВМ-2 (новой), на которую предполагается переносить программы значительно меньше, чем те, которые имеются у ЭВМ-1 (старой), на которой они разрабатывались и достаточно полно использовались, то программа может оказаться или принципиально не реализуемой на ЭВМ-2, или не эффективно исполняемой. Таким образом, ресурсы ЭВМ-2, на которую подготавливается перенос программ, как правило, ***должны быть не меньше, чем у ЭВМ-1***, для которой они первоначально разработаны.

Особое значение для переносимости имеет специфика реализации взаимодействия с внешней средой и процедур ввода-вывода в ядре операционных систем сопоставляемых ЭВМ. Программы, реализующие эти функции, должны отражать специфику конкретной системы ввода-вывода и особенности источников и потребителей информации. Такая ориентация программ ввода-вывода может делать их практически не переносимыми и требовать выделения в автономные заново разрабатываемые, автоматизировано непереносимые компоненты.

Характеристики операционных систем по степени влияния на переносимость можно рассматривать в двух вариантах.

В первом варианте архитектура ОС оказывает доминирующее влияние на интерфейсы и особенности реализации прикладных программ. Перенос ОС должен обеспечить её эффективное использование на новой ЭВМ-2 для разработки и функционирования комплекса программ. Поэтому основное внимание уделяется унификации и переносу одной и той же ОС в различные по архитектуре ЭВМ. Примером является широко распространенная ОС UNIX и ее версии для разных аппаратных платформ. Однотипность ОС обеспечивает унифицированную среду для пользователей прикладных и системных программ, а также стимулирует применение унифицированных интерфейсов между компонентами ОС и прикладными программами.

Во втором варианте полный перенос ОС считается не целесообразным и необходим перенос только прикладных программ и БД. Эта ситуация наиболее характерна при смене типов ЭВМ в системах управления реального времени объектами (бортовые ЭВМ) или технологическими процессами. Стремле-

ние сохранить программный задел проблемно-ориентированных программ и обеспечить высокую эффективность ОС при новой архитектуре ЭВМ приводит к необходимости переработки или замены ОС с сохранением структуры интерфейсов с программами и внешней средой. В обоих вариантах для сохранения программного задела при переносе, определяющим является унификация интерфейсов прикладных программ со стандартизированной или иной заменяемой операционной системой [60, 62].

Языки программирования отражаются негативно на переносимости программ тем сильнее, чем больше различается архитектура ЭВМ и чем ниже уровень языка. Повышение уровня языка программирования сопряжено с повышением универсальности текстов программ и уменьшением их машинной зависимости. Одновременно ***уменьшается эффективность объектного кода и расширяется исполняемый текст программ***, получаемых в результате трансляции. Таким образом, за снижение трудоемкости разработки программ путем переноса на языках третьего и четвертого поколений приходится расплачиваться увеличением памяти для их размещения и времени при исполнении на новой ЭВМ. Поиски компромисса приводят к выделению проблемно-ориентированных областей применения ПС, различающихся требованиями к допустимому расширению программ в процессе переноса.

В тех случаях, когда требуется особенно эффективное использование ресурсов ЭВМ, на которую предстоит перенос программ, и одновременно осуществлять его автоматизировано, необходимо использовать стандартизированные языки программирования высокого уровня. Однако при этом возможны различия версий одного и того же языка программирования и особенностей компиляторов для его реализации на различных ЭВМ. Стремление получить после трансляции программы, высокоэффективные при исполнении на конкретной архитектуре ЭВМ-2, иногда приводит к отступлениям от стандартов и к вводу в описание языка некоторых особенностей, учитывающих специфику ЭВМ. Кроме того, ряд описаний и стандартов языка имеют не полностью определенные и неоднозначные для понимания фрагменты, которые по-разному могут использоваться при разработке текста программ и приводить к различным результатам при их трансляции и исполнении. Вследствие этого некоторые ***языки могут иметь особенности и диалекты***, вы-

зывающие необходимость доработки текстов программ при их переносе на новую ЭВМ. Эта проблема решена в языке Ада, где изначально запрещены версии и диалекты. Однако и в этом стандарте имеются фрагменты, допускающие неоднозначную интерпретацию, что привело к необходимости выпуска специальных комментариев для уменьшения разночтений при разработке текстов программ и компиляторов [2].

Реализации компиляторов на различных ЭВМ зависят от архитектуры и ресурсов этих ЭВМ, а также от квалификации специалистов, создающих компиляторы. В результате один и тот же текст программ преобразуется в различный объектный код, даже при тождественной архитектуре ЭВМ. Для обеспечения полной переносимости программ необходима аттестация компиляторов. Такая аттестация должна гарантировать корректное и однозначное восприятие и преобразование всех конструкций языка. Впервые подобная аттестация реализована для языка Ада. Она поддержана комплексом аттестационных тестов, которым должен удовлетворять любой допускаемый к эксплуатации компилятор. В результате стандартизация языка Ада (с поддерживающими комментариями) и аттестация компиляторов обеспечивают высокоэффективную автоматизированную переносимость программ на различные типы ЭВМ [2, 36].

Средства автоматизации проектирования и переноса комплексов программ и данных и их доступность разработчикам могут значительно влиять на эффективность переноса. Подобные методы и средства составляют современную программную инженерию. В ней можно выделить следующие ***важные для технологии переноса факторы***:

- наличие и качество проблемно-ориентированных типовых технологий для основных классов ПС и БД, поддерживающих разработку и перенос программ и данных на современных языках программирования высокого уровня типа Си, Ада, Паскаль;
- степень использования методов и средств повышения производительности труда при создании ПС на основе системного анализа и проектирования, а также сборочного программирования, использующего ресурсосберегающие принципы многократного повторного использования программных компонентов на различных аппаратных платформах;

- эффективность методов и средств обеспечения качества создаваемых прикладных ИС на основе индустриализации технологии и процессов разработки, а также гарантирования качества путем тестирования, независимой аттестации и сертификации программных средств и баз данных.

Наличие и применение международных стандартов открытых систем в процессе разработки программных средств и баз данных способно обеспечивать архитектурную, техническую и программно-информационную совместимость и эффективную переносимость, разрабатываемых и применяемых информационных систем в соответствии с международными нормативно-техническими документами (см. главу 4).

Факторы, влияющие на эффективность переноса баз данных– в ряде случаев особое значение имеет не столько перенос программ, сколько перенос данных. Информация о процессах, происходящих во внешней среде, может иметь большие размеры и трудоемкость первичного накопления и актуализации, что определяет необходимость её тщательного хранения. Возможны ситуации, когда подобные данные являются уникальными и невосстанавливаемыми. Однако первичные аппаратная или операционная платформы, в которых они накапливались и обрабатывались, может требовать замены. Формирование и наполнение информацией баз данных достаточно сложный и трудоемкий процесс, технико-экономические показатели которого сильно зависит от структуры, состава, объема, связности и других характеристик исходной информации. Для осуществления переноса различных данных большого объема, необходимы методы и средства автоматизации, которые обеспечивают достаточную эффективность этого процесса [22, 57].

При анализе БД, как объектов переноса, целесообразно рассматривать ***два компонента***: системы управления данными (СУБД) и совокупность данных, упорядоченных по некоторым правилам. При этом одна и та же система управления базой данных 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 они могут создаваться независимо и рассматриваться при полной разработке и переносе как ***два объекта, которые различаются***:

- номенклатурой и содержанием показателей качества, определяющих их назначение, функции и потребительские свойства;
- технологией и средствами автоматизации разработки, переноса и испытаний объектов;
- категориями специалистов, обеспечивающих:
  * создание БД (разработчики СУБД и разработчики исходной информации БД);
  * эксплуатацию БД (администраторы СУБД и администраторы данных);
  * применение БД (операторы-пользователи функциональных возможностей СУБД и пользователи доступных данных);
- эксплуатационной и технологической документацией, поддерживающей жизненный цикл объектов.

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 её 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 не удовлетворительные 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 дли-

тельность переноса БД в этом случае значительно возрастают и требуют тщательного планирования и организации работ, приближающихся к созданию совершенно новой БД. Обычно одновременно должно быть обеспечено сохранение или повышение качества функционирования БД на новой платформе.

В общем случае при переносе баз данных можно выделить ***две принципиальные проблемы***, которые различаются сложностью и методами решения. ***Первая проблема*** сводится к обеспечению полного достоверного переноса всей информации, накопленной в ЭВМ-1, на ЭВМ-2 с иной архитектурой. При переносе данных с ЭВМ-1 на ЭВМ-2 с другой архитектурой возникает задача технического использования совместимого носителя данных или канала связи между ЭВМ. В результате, накопленная в ЭВМ-1 информация, должна быть доступна для автоматизированной транспортировки в ЭВМ-2.

Форматы, кодировка, длина слова и другие особенности записи данных зависят от архитектуры ЭВМ и типа СУБД и могут требовать преобразования исходных данных для эффективного их использования после переноса в новую ЭВМ-2 с иной СУБД. Также различными на этих ЭВМ могут быть структуры файлов, их разметка и система управления ими. Современные сети телекоммуникации обеспечивают возможность надежной транспортировки больших объемов данных между ЭВМ с различной архитектурой. Для этого в операционных системах обеих ЭВМ должны иметься средства приема и передачи файлов, унифицированной структуры. Эти операции упрощаются, если в обеих ОС структура файлов организована в соответствии с международными стандартами.

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.

Вторая проблема заключается в обеспечении переноса функций и процедур системы управления базой данных. Для этого необходимо иметь полные и достоверные сведения о структуре записей информации, их связях, взаимодействии и логике использования в исходной ЭВМ. 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.

В современных базах данных используются различные структуры, логика размещения и поиска данных. В соответствии с этим СУБД имеют особенности компонентов и процедур, осуществляющих их основную обработку. Эти особенности отражают также специфику языков программирования систем обработки данных в ЭВМ-1 и ЭВМ-2. Перечисленные характеристики могут оказаться совокупностью факторов, существенно ограничивающих эффективный перенос БД.

Приведенные трудности переноса баз данных ***могут быть облегчены стандартизацией*** соответствующих компонентов аппаратной платформы, операционной среды, СУБД и языков программирования. Однако такая широкая и полная стандартизация вряд ли возможна и обычно приходится проводить локальную унификацию для снижения трудоемкости переноса некоторых классов данных. Унификация может производиться либо выбором ОС, СУБД и языка программирования, наиболее пригодных для конкретного переноса данных, либо созданием семейства ЭВМ и ОС с расширяющимися аппаратными и функциональными возможностями. Таким образом, объективная необходимость переноса больших объемов данных выдвигает специфические задачи, решение которых может быть достаточно эффективным только при определенных условиях. Основными из них являются стандартизация языков и методов доступа к данным в БД, а также ***унификация номенклатуры и функций***

СУБД, применяемых для обработки определенных классов данных.

Анализ целесообразности создания информационных систем путем переноса комплексов программ и данных и сравнение с полной, новой разработкой. Для принятия такого решения следует оценить и сравнить эффективность обоих методов создания ИС в конкретных условиях. Для оценки эффективности переноса программ и данных с исходной платформы на новую ЭВМ-2 с ОС-2 следует учитывать ***возможные затраты*** на:

- проведение системного анализа и подготовку возможного решения о проведении переноса конкретного программного продукта или его компонентов с ЭВМ-1 и ОС -1 на ЭВМ-2 с ОС-2, а также на приобретение переносимой версии ПС и БД и их документации;
- выбор, приобретение или разработку инструментального комплекса автоматизации создания программ и его адаптацию на архитектуру ЭВМ-2 и её операционную систему, а также на освоение специалистами инструментального комплекса по переносу на ЭВМ-2;
- выделение и переработку текстов программ машинно-ориентированной части ПС и БД для эффективной реализации их в ЭВМ-2;
- подготовку текстов программ и описаний данных машинно-независимой части ПС для их эффективной реализации в ЭВМ-2;
- трансляцию текстов программ для ЭВМ-2;
- перенос информации базы данных из ЭВМ-1 в ЭВМ-2;
- полное тестирование машинно-ориентированной части переносимых программ для ЭВМ-2;
- контрольное тестирование машинно-независимой части компонентов версии ПС и БД для ЭВМ-2;
- комплексирование, сборку и квалификационное тестирование всех компонентов в версии ПС и БД для ЭВМ-2;
- доработку и выпуск полного комплекта машинных носителей и документации на версии ПС и БД для ЭВМ-2;
- испытания и сертификацию перенесенной версии ПС и БД при ее реализации на ЭВМ-2.

В конкретных ситуациях критическими могут быть различные виды затрат и различные составляющие в каждом из видов.

При этом определяющими для выбора метода создания ИС могут быть не абсолютные значения, а их величины относительно затрат при полной разработке ПС и БД с учетом имеющегося системного и алгоритмического задела в версии для ЭВМ-1.

Оценка целесообразности переноса предполагает, что первоначально ПС и БД разработаны для реализации на ЭВМ-1, работающей с ОС-1, и необходимо создать аналогичные по функциям и качеству ПС и БД на ЭВМ-2 и ОС-2 с иной архитектурой. При этом, в данном случае, доступны для использования в качестве прототипов алгоритмы, спецификации, тексты программ и описания данных для ЭВМ-1, а также тесты и документация на ПС и БД. Создание ПС и БД методом переноса предполагает ***прямое использование текстов программ и данных***, реализованных на ЭВМ-1.

Исходными данными для выполнения анализа являются характеристики обеих ЭВМ и их операционных систем, а также характеристики ПС и БД, предполагаемых к переносу (рис.1.6).

Рис. 1.5

Анализ архитектуры и ресурсов вычислительных средств позволяет оценить степень их различия для выделения вариантов, когда перенос явно не целесообразен. Если рассматриваемые вычислительные машины имеют характеристики, допускающие эффективный перенос, то анализ продолжается в направлении сопоставления операционных систем. При сравнении может оказаться, что их интерфейс с прикладными программами и внешней средой столь сильно различаются, что перенос явно не рентабелен. Это может быть тогда, когда необходимая доработка компонентов интерфейса очень велика и не оправды-

вает перенос ПС и БД, которые целесообразнее разработать заново.

В случаях, когда характеристики интерфейсов операционных систем достаточно близки и не имеют явных признаков, препятствующих переносу, начинается анализ возможных затрат и эффективности предполагаемого переноса программного средства и данных. Обычно мала эффективность переноса небольших ПС (например, объемом меньше нескольких тысяч строк исходного текста). В крупных ПС следует выделять и оценивать долю компонентов, имеющих машинную ориентировку на ЭВМ-1. Если машинно-ориентированная часть велика (например, порядка 20-50% полного объема ПС), то вероятно, что автоматизированный перенос окажется не целесообразным. В противном случае машинно-ориентированные программные компоненты подготавливаются к повторной разработке для ЭВМ-2. Однако возможны ситуации столь резких различий архитектуры ЭВМ, при которых заимствование даже алгоритмического задела по машинно-ориентированным компонентам не целесообразно.

В оставшейся части ПС анализ продолжается для оценки переносимости баз данных.

Различия архитектуры ЭВМ, характеристик ОС и СУБД могут значительно усложнить или сделать не целесообразным прямой перенос накопленных данных, а также использующих их компонентов ПС. На эффективность переноса информации БД существенно влияет структура базы данных, идентичность СУБД и структура операций обращения к памяти. В случае возможности сохранения или малых изменений структуры информации и описаний базы данных проводится анализ текста ПС на языке программирования с точки зрения эффективности его переноса. При этом необходимо оценить степень машинной ориентации диалекта языка, на котором разработано ПС для ЭВМ-1, и его отличия от диалекта языка, предполагаемого для использования для ЭВМ-2. Такие различия языков, а также их реализации трансляторами могут отражаться на эффективности и особенностях, получаемого в результате трансляции объектного кода программ. Учет архитектуры ЭВМ при подготовке текстов программ может влиять на степень использования про-

изводительности информационной системы с ЭВМ-2 и с перенесенными ПС и БД.

Для средних и крупных проектов ПС 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 оценить по использованию памяти и производительности ЭВМ-2. 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 ИС вследствие увеличения объема программ в объектном коде и снижения производительности ИС, особенно когда программы длительно используются на многих экземплярах ЭВМ в реальном времени.

После переноса версии ПС и БД на другую платформу, они могут начать самостоятельную жизнь и развитие, отличное от модификаций исходной версии. Так как перенос ПС и БД часто обусловлен необходимостью увеличения ресурсов ЭВМ, доступных для решения новых перспективных задач, их разработка становится естественным процессом расширения функций программ и данных относительно исходной версии. Поэтому при подготовке технологических средств автоматизации переноса необходимо предусматривать затраты на средства обеспечения всего жизненного цикла ПС и БД от системного анализа до отладки, испытаний и сертификации всех компонентов [16, 25].

Глава 2

УПРАВЛЕНИЕ КОНФИГУРАЦИЕЙ В ЖИЗНЕННОМ ЦИКЛЕ ПРОГРАММНЫХ СРЕДСТВ

2.1. Методы и процессы управления конфигурацией программных средств

Цель управления конфигурацией при 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 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 (см. п. 1.2 табл. 2).

В процессе проектирования ПС должна быть формализована и документально зафиксирована ***Концепция организации***

конфигурационного управления, содержащая в основе [40, 47, 51]:

- ожидаемую длительность поддержки развития и модификации конкретного проекта ПС;
- масштаб и уровень предполагаемых изменений и модификаций;
- возможное число и периодичность выпуска базовых версий ПС;
- организационные основы процессов сопровождения и конфигурационного управления ПС;
- 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 продуктов. Управление конфигурацией (УК) следует организовать так, чтобы персонал знал свои обязанности и имел достаточно независимости и полномочий для выполнения поставленных задач. Для описания политики фирмы, видов ее деятельности и правил, связанных с процессом управления конфигураций, следует использовать формализованные процедуры и документы.

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. 2.1).

Рис. 2.1

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 (см. п. 1.2 табл. 2).

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. п. 1.2). 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 не синхронно по множеству версий пользователей, в результате чего образуется набор версий выполненных изменений, учитывающих текущие состояния версий ПС у конкретных пользователей.

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 достижения ***следующих основных целей*** (стандарты **ISO 12207, ISO 10007, ISO 15864**):

- 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;
- обеспечить контроль над тем, чтобы фиксировались дефекты и ошибки, а изменения регистрировались, утверждались и реализовались;
- гарантировать надежное физическое архивирование, восстановление и сопровождение единиц конфигурации и документов проекта ПС.

Изменения конфигурации ПС и его компонентов должны планироваться и предусматривать в плане управления проектом действия с четкими разделами [40, 47, 64]:

- ***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 специа-

листов. Этому должна способствовать утвержденная иерархия принятия решений на изменения компонентов и ПС в целом должностными лицами проекта (см. п. 1.2 табл. 2), 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*** (см. п. 3.2). 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. В совете должны

бить представители по всем необходимым дисциплинам. Его должен возглавлять руководитель проекта или его представитель. 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. ***Функция совета по конфигурации*** заключается в подтверждении того, что [40]:

- он имеет соответствующие полномочия относительно утверждения корректировок и конфигурационной базы;
- конкретное изменение необходимо;
- последствия изменений приемлемы;
- изменение должным образом документировано и классифицировано;
- план внесения изменения в документы и программное средство является удовлетворительным.

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 документации. Кроме того, каждый вариант модуля, компонента или ПС и их изменений должен соответствовать правилам нумерации (идентификации).

Цель работ по 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 каждой отдельно управляемого компонента конфигурации и для комбинаций единиц конфигурации, которые составляют ПС;
- провести идентификация 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 должен участвовать в выборе ЕК, выполняемому согласно проекту архитектуры системы, должен идентифицировать объекты, которые будут помещены под управление конфигурацией, и должен 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.

Использование более высокого уровня абстракции ЕК ведет к снижению сложности и делает любую проблему более понятной. Единую систему проще собирать из цельных компонентов ЕК, чем из разрозненных версий файлов модулей программ и данных. Несовместимость идентификации модулей ведет к повторным сборкам компонентов проекта и к ошибкам, которые обнаруживаются на более поздних стадиях разработки. Сделать заключение о качестве определенной редакции компонента легче, чем о множестве отдельных файлов модулей. Редакция должна характеризовать одну и только одну версию файла модуля или каталога, входящего в состав компонента. Поскольку редакция компонента содержит согласующийся набор версий, его можно протестировать как единое целое, отметив уровень проведенного тестирования. Данный метод позволяет улучшить согласованность действий и уменьшить число ошибок, когда несколько групп специалистов совместно работают над компонентом.

Превращение физической реализации компонента ЕК в инструмент позволяет использовать его совместно и многократно. Когда имеются различные версии компонента и уровень их качества определен, разработчики могут выбрать одну из редакций для при-

менения в проекте. Отображение логических компонентов проектирования на физические компоненты ЕК помогает сохранить целостность программной архитектуры. Для устранения рисков в итерационном процессе разработки некоторые элементы программной архитектуры реализуются и тестируются на ранних стадиях жизненного цикла. С помощью отображения логических компонентов на физические можно создавать и тестировать отдельные элементы архитектуры, соответствие между архитектурой и ее реализацией позволяет получить более качественный код и понятные интерфейсы между компонентами.

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 уникальными. 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, что позволяет получать сведения о любой версии системы, сравнивать различные версии, находить ошибки и составлять документацию.

Эти факторы необходимо учитывать для решения проблем всего производственного процесса: они способствуют быстрому устранению дефектов в выпущенных продуктах, облегчают проведение стандартных процедур аудита по стандарту **ISO 9000** и, в конечном счете, гарантируют, что проект соответствует системным требованиям, а код в точности реализует проект.

Управление запросами на изменения включает регистрацию, отслеживание и анализ запросов на модификацию ПС, компонентов и данных от субъектов, взаимодействующих с системой (см. рис. 2.1). Оно включает процессы принятия решений для планирования необходимых изменений и процессы реализации для их осуществления. Управление запросами на изменения представляет собой ***центральную часть Концепции управления конфигурацией***. Незарегистрированные запросы на изменения могут быть потеряны или останутся без реакции. Вне отслеживания существующие запросы часто не выполняются или неправильно адресуются, специалист, не использующий анализ и отчеты, столкнется с трудностями при определении статуса проекта, оценке качества продукта и доведении результатов работы до руководства проекта.

Методология УК определяет управление запросами на изменения как процесс, который формирует инфраструктуру предприятия, необходимую для оценки затрат и планирования запрашиваемых изменений в существующем программном продукте. К дисциплинам, тесно связанным с управлением запросами на изменения, можно отнести управление требованиями к системе, тестирование, управление выпуском, управление проектом и организацию поддержки пользователей. В запросе содержится вся информация о происхождении текущей проблемы и ее влиянии на систему, предлагаемом решении и оценке издержек. Процессы управления обычно непосредственно связаны с внутренней организацией работы предприятия. Запросы на изменения, как правило, разделяют на две основные категории: запросы на доработку – совершенствование и дефекты ПС.

Запрос на доработку определяет новое свойство ПС или изменение реализации его функций. ***Дефект*** – это аномалия или брак в работающем программном продукте. 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 и др.). Типы запросов на изменения, могут быть общими, как запрос на доработку и дефект. Такой внешний запрос способен в свою очередь породить несколько внутренних технических запросов на доработку.

Следующий шаг после установления типов контролируемых запросов, уточнение 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 [40]:

- представление и регистрация запроса на изменение;
- оценка запроса его категории и приоритета;
- решение о порядке выполнения запроса;
- реализация корректировок – компоненты системы и про-

граммная документация создаются или модифицируются с целью реализации запроса;

- проверка на соответствие требованиям или на отсутствие исправленного дефекта.

Представление запроса на изменение предполагает его регистрацию. Дефект и запрос на доработку обычно отличаются своим происхождением и набором фиксируемой информации. Запросы на доработку могут поступать из различных источников. В большинстве случаев они приходят от заказчиков и возникают в процессе работы напрямую или косвенно, через опрос сотрудников. Основные данные, регистрируемые при возникновении таких запросов, это важность требуемых изменений для пользователей, подробности запроса и личность инициатора. Последнее требуется, чтобы при необходимости нетрудно было выяснить детали и неясные моменты. Иногда запрос на доработку появляется в самой организации в процессе тестирования или внутреннего использования проекта.

Сведения о дефектах также могут передаваться из различных источников. Многие дефекты обычно обнаруживаются, записываются и исправляются внутри организации. В качестве сопроводительной информации для дефекта регистрируются обстоятельства его возникновения, способ повторного воспроизведения, степень значимости и данные о том, кто обнаружил ошибку. Для дефектов записываются версия программы, параметры операционной системы и конфигурация аппаратных средств пользователя, сообщившего об ошибке.

Оценка запроса заключается в просмотре вновь поступивших запросов и выводы о их характеристиках. Является 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. В общем случае запросы на доработку оцениваются руководителем проекта.

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 кор-

ректироваться только с разрешения руководителей соответствующего ранга. Тем самым должны предотвращаться несогласованные и несанкционированные изменения, способные снизить качество и целостность версий ПС. Изменения этих версий допускаются пользователями или поставщиками в пределах, ограниченных эксплуатационными документами по адаптации к характеристикам и особенностям внешней среды и условий 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. Сообщения о дефектах и изменениях должны описывать необходимые действия, связанные с коррекцией, а так же предполагаемую дату их выполнения. Эти сообщения должны использоваться как входные данные для системы корректирующих действий.

При сопровождении ПС необходимо решать, какие компоненты и версии потребуются для выполнения запланированных функций и действий. Компоненты и их редакции определяют сферу работ над проектом. Базовые редакции определяют начальную конфигурацию для реализации интеграционного потока изменений при развитии проекта. Нужные компоненты можно выбирать по отдельности или указав существующий проект, от которого наследуется перечень применяемых компонентов. Еще один важный аспект определения изменяемой конфигурации – указание возможности модификации компонента либо использования его только для чтения. 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 используется 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 и каждую проблему выполнения работ, необходимых в соответствии с контрактом или описанных в Плане разработки (см. рис. 2.1):

- 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. Дефекты и запросы на доработку почти всегда обрабатываются по-разному. Для запроса на доработку решение обычно выносит руководитель проекта или аналитик. Запросы на доработку оцениваются вместе или по отдельности. 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. В крупных проектах иногда для контроля над изменениями создается один или несколько специальных советов. Задача контрольных советов заключается в нахождении компромисса между качеством продукта и сроками выпуска версии на завершающем этапе разработки программного продукта.

Реализация корректировок при запросе на доработку требует 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.

Руководителю проекта необходимо указывать, как часто специалисты должны собирать обновления, сделанные другими сотрудниками, в свой поток изменений компонента. Для отправки в интеграционный поток разработки, исправления должны соответствовать рекомендуемым или более свежим редакци-

ям. Это правило вместе с частотой определения новых рекомендуемых редакций дает возможность подстраивать процессы УК на протяжении цикла модификаций, увеличивая или уменьшая степень изоляции отдельных участников проекта. При этом каждый обязан обновить поток изменений до рекомендуемых редакций перед отправкой их на интеграцию. Он должен собрать все сделанные модификации в свой поток и убедиться, что они согласуются с его собственными изменениями, – при сборке это позволит сократить количество проблем и противоречий, возникающих в интеграционном потоке.

К***онтроль 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 не утверждение 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 обра-

ботать 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 ПС 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.

Хотя все системы УК обеспечивают контроль версий на уровне файлов, именно разработчик корректировок решает вопрос о выборе нужных версий ЕК, которые совместно образуют логически законченное изменение и интегрируются в единый блок. Это трудоемкий процесс, который часто ведет к возникновению ошибок, особенно если специалист одновременно занят несколькими модификациями. Такого рода проблемы могут проявиться при сборке и позже, на стадии выполнения интегрированного компонента. Некоторые инструменты УК позволяют записывать, над каким запросом на изменение или исправление дефекта идет работа в данный момент. Эти сведения нужны для отслеживания тех файлов компонентов и каталогов, модификация которых образует единое ло-

гическое изменение ЕК. Часто такая информация не используется самим инструментом, но фиксируется для получения отчетов. Главное преимущество сбора информации об изменениях – упрощение процесса интеграции и гарантия целостности конфигурации ПС в любой рабочей среде.

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. В процессе управления конфигурацией ПС может встать вопрос, нужно ли сначала проводить

сборку, а потом создавать базовые редакции компонентов, или наоборот. У каждого варианта есть свои преимущества и недостатки. 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 проводить сборку базовой версий ПС и допускаются на этом этапе мелкие коррективы, то лучше сначала собрать ПС, а потом уточнять и создавать базовые редакции компонентов ЕК.

Операция создания базовых редакций ЕК должна просматривать все отправленные в интеграционный поток действия и изменения. Она определяет, какие компоненты ЕК данного проекта подверглись модификации, и предлагает создать новые редакции для каждого из этих компонентов. Редакция явным или неявным образом объединяет определенные версии элементов, входящих в состав компонента.

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. 2.1). Интегратор первым локализует возникшие при сборке неполадки, предотвращая, таким образом, перенос проблем несовместимости в рабочие пространства специалистов. 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. Её 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. Новая базовая версия должна быть прослежена к той базовой версии ЕК, производной от которой она является. ***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 и обычно выполняются в автоматическом режиме. 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 версии 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 ПС, когда требуется жестко придерживаться запланированных сроков.

Составление отчетов о состоянии конфигурации должно начинаться с того момента, когда будут получены первые данные о комплекте ЕК (см. п. 2.3). Отчеты о статусе конфигурации должны предоставлять информацию о идентификации конфигурации и всех отклонениях от зарегистрированных конфигурационных баз [14, 17, 22]. 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 отражена 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 состоит 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 помеще-

ны 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.

Постановка и выпуск – важные процессы при управлении конфигурацией базовыми версиями программного продукта. ***Постановка*** – процесс помещения объектов ЕК (исполняемых модулей, библиотек, файлов данных, откомпилированных файлов, документов) в базу данных под жесткий версионный контроль. Основное назначение постановки состоит в помещении копий исполняемых модулей и других производных файлов ЕК в хранилище базовых версий для надежного и безопасного доступа к ним. Выполнение процесса постановки определяется типом разрабатываемого программного комплекса. ***Выпуск*** – придание собранным ЕК окончательной формы и предоставление доступа к ним конечным пользователям. Цель выпуска заключается в подготовке базовой версии программного продукта и документов для заказчика. Каждое предприятие может иметь собственный процесс выпуска, зависящий от следующих факторов:

- способа доставки программного продукта заказчику;
- способа сборки версии программного продукта;
- организационной структуры предприятия;
- способа изготовления дистрибутива;
- количества пользователей версией программного продукта;
- частоты выхода новых версий;
- корпоративного использования продукта или его продажи другим предприятиям.

Архивированию и тиражирование базовых версий программных продуктов и документов должны гарантировать использование исключительно санкционированных версий ПС и компонентов, так что официальные версии могут быть получены только из архива. Каждая единица конфигурации должна быть идентифицирована, документирована и выпущена ее офи-

циальная 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 регламентированы ***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 [4, 26]. 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. Априори перечисленные факторы прогнозировать невозможно и оценки этой составляющей затрат целесообразно проводить по результатам начальных этапов сопровождения и модификаций первых версий. Затраты на тиражирование и адаптацию к параметрам среды пользователей зависят от широты распространения программного продукта и достаточно просто могут оцениваться по типовым прецедентам аналогичных проектов или предшествующих версий. Гибкость и кажущаяся простота изменения программ и данных значительно затрудняют оценки необходимого, иногда весьма значительного, финансирования на сопровождение и проведение жесткого конфигурационного управления ПС.

2.2. Этапы и процедуры при управлении конфигурацией программных средств

Конфигурационное управление в значительной степени обеспеченно, 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 офор-мления 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, не скоординированным 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 превышают четыре версии. Для крупномасштабных ПС это со-

ответствует рациональному времени жизни и тиражирования каждой очередной базовой версии около 2 – 4 лет [18].

В стандартах, регламентирующих конфигурационное управление ПС (см. Приложение), представлен ***комплекс процедур и состав специалистов***, организующих эти процессы – рис 2.2. 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 изменениях компонентов проекта. Формализация обмена информацией должна повысить ответственность специалистов за корректность её содержания, оперативность формирования и изменения, качество процессов трансформации и реализации данных и документов в жизненном цикле ПС.

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. 2.2). 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.

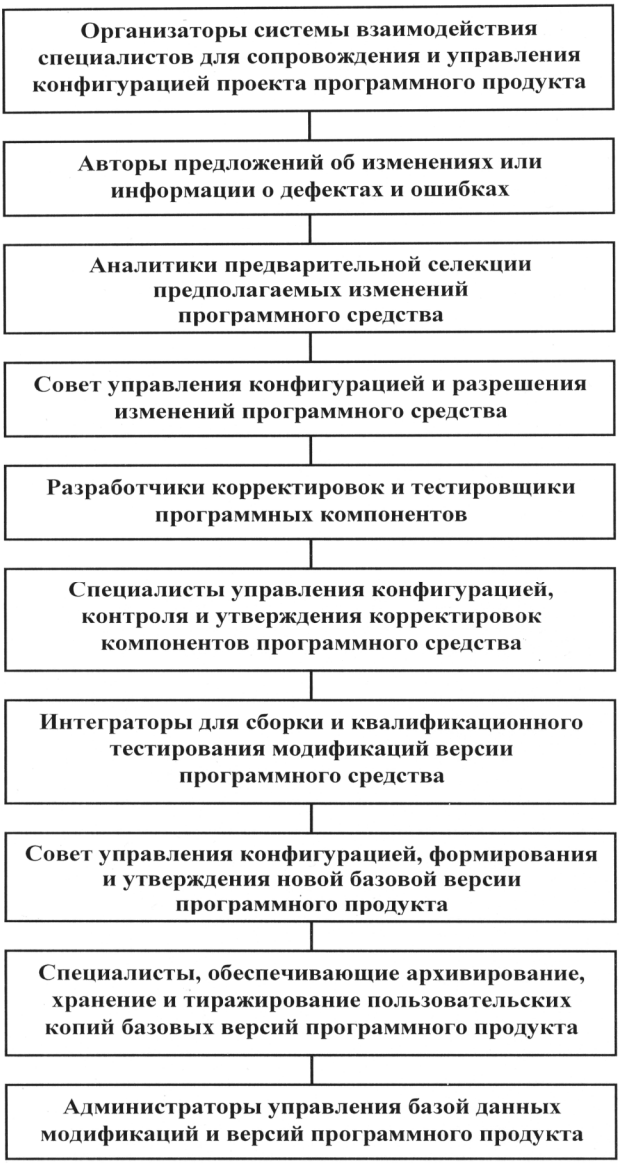

Рис. 2.2

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 не подтверждение, 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. Все эти дефекты подлежат регистрации вместе с тестами и условиями их проявления [9, 11, 21].

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.

Еще на стадии проектирования могут возникать идеи совершенствования программ, которые в то время невозможно реализовать из-за высокой стоимости, ограниченных сроков проектирования или по иным причинам. Идеи изменения могут быть направлены на коренное совершенствование функциональных возможностей программ или некоторые "косметические" улучшения реализуемых функций. 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, поделенные в базе данных на крупные разделы:

- 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.

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. 2.2). 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 де-

фекты. 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-ые 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. 2). 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) [9, 46]. Процесс проведения изменений должен поддерживаться средствами автома-

тизации конфигурационного управления и обеспечения защиты от несанкционированного 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 (см. п. 2.3).

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 кон-

кретных 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. 2.2).

Для проверки корректности внесения изменений в версии ПС, создаваемые с применением повторно используемых компонентов возможно предсказание мест в комплексе программ, где наиболее вероятны вновь внесенные при корректировке дефекты и ошибки и каких типов они могут быть. Поэтому зачастую нет необходимости каждую новую версию ПС подвергать столь же широкому тестированию, как первую или предшествующую версию. 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 ПС или для срочных извещений пользователям***.

Некоторые частные, срочные корректировки, существенно влияющие на надежность или безопасность применения ПС, мо-

гут первоначально сообщаться пользователям до их внедрения в новую версию. Для этого формируются ***предварительные извещения***, которые рассылаются зарегистрированным пользователям с содержанием, объяснениями назначения и особенностей корректировок. Реализация корректировок по частным извещениям должна учитываться, координироваться и сообщаться пользователям при внедрении новой (n+1)-й базовой версии.

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.

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) оформляется 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. В некоторых случаях может быть полезным выпуск

рекламного извещения для пользователей, объявляющего создание (n+1)-й версии ПС и ее основные, функциональные и качественные отличия от предыдущей версии.

Пользователям может поставляться базовая копия (n+1)-й 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 ПС пользователей в крупномасштабных, 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 ПС.

Для независимого удостоверения достигнутого качества проведенных модификаций в испытанной и утвержденной разработчиком версии ПС, по заявке разработчика, заказчика или пользователей она может подвергаться обязательной или добровольной ***сертификации*** [16]. 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 примене-

ния (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 так же 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)-ой 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 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)-ой базовой версии могут быть взаимозависимы и частично альтернативны как непосредственно, так и через некоторые промежуточные программы и данные, что трудно обнаруживать. При подготовке (n+1)-ой 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 приводит к ошибкам функционирования версии m-го пользователя. Особое значение синхронизация выполнения изменений ПС приобретает в распределенных ИС типа клиент-сервер при наличии центра информационной системы и множества уда-

ленных, 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)-ой базовыми версиями в пределах только простейших локальных корректировок ошибок, существенно влияю-щих на качество конкретных версий ПС. Для предотвращения развала взаимодействия версий клиентских программ в распределенной ИС ***следует запрещать несанкционированные изменения пользовательских версий ПС***, минуя совет конфигурационного управления.

Наличие множества базовых и пользовательских версий ПС может значительно расширять характеристики внешней среды, а, следовательно, и номенклатуру исходных данных и применяемых тестов для проверки реального качества функционирования программ. Контроль состояния и обеспечение качества множества эксплуатируемых версий ПС требует использования ряда дополнительных средств тестирования и может существенно его усложнять. Это, в частности, отражается на расширении базы данных проектирования, в которой длительное время должны сохраняться упорядоченные версии ПС и их компонентов, предполагаемые и реализованные изменения программ и описаний данных, а также основная часть тестов, гарантирующих контроль характеристик качества новых версий ПС.

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 ПС. 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.

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 пер-сонала, предупреждение о проведении и завершении переноса архива, оценивание влияния новой среды и архивирование соответствующих данных (см. п. 1.5). После

планирования переноса, пользователям должно быть направлено уведомление о планах и работах по переносу базовых версий ПС и управления конфигурацией 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.

После завершения переноса должен быть выполнен итоговый анализ для оценки влияния перехода к новой среде УК на различные аспекты эксплуатации перенесенного программного продукта. Как часть указанной задачи разработчик должен: проанализировать результаты параллельной эксплуатации УК в старой и новой среде; уточнить специфику новых абонентов; документально зафиксировать последствия и дефекты переноса; создать и опубликовать отчет по анализу влияния переноса УК на характеристики ПС. Данные, использованные или связанные с прежней средой, должны быть доступными для аудиторской проверки.

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 приемосдаточными испытаниями, которым предшествуют опытная эксплуатация и межведомственные либо государственные испытания. Кроме того, разрабатывается учебно-методический план, подготавливались 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.

Технической основой сопровождения и управления конфигурацией являются ***системы управления базами данных*** (СУБД), адекватные целям и функциям проектов, структурированные по целям, назначению и содержанию данных в выделенных подсистемах (рис. 2.3). 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 сопровождения и УК, а также Руководство по её применению***, настроена выбранная СУБД на управление основными взаимодействующими подсистемами базы данных, с учетом класса и масштаба предполагаемого проекта ПС (рис. 2.4). По мере развития жизненного цикла проекта комплекса программ, подсистемы БД 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 специа-

листов, текущее время выполнения процедур развития проекта и оформления изменений в подсистемах БД.

Эта ***информация в подсистемах базы данных сопровождения и УК*** должна быть защищены от случайных и преднамеренных искажений, путем организованного санкционирования, дублирования и контроля модификаций, истории их создания и изменения, в процессах жизненного цикла ПС.


Документооборот менеджеров проекта комплекса программ
База данных предварительных требований и ресурсов для разработки программного средства
Документооборот системных аналитиков и проектировщиков требований на компоненты комплекса программ
База данных спецификаций требований на модули, компоненты, интерфейсы и характеристики качества программного средства
Документооборот программистов и тестировщиков компонентов комплекса программ
База данных готовых текстов и результатов тестирования модулей, программных компонентов и данных
Документооборот верификаторов и системных интеграторов функциональных компонентов комплекса программ
База данных верификаторов, тестировщиков и интеграторов функциональных компонентов и данных программного средства
Документооборот квалификационных испытателей качества и оформителей завершенного программного продукта
База данных для квалификационного тестирования, архив испытанных и завершенных версий и характеристик программного продукта
Документооборот специалистов сопровождения, конфигурационного управления версиями и корректировками программного продукта
База данных сопровождения, конфигурационного управления, корректировок, версий и архивации программного продукта

Рис. 2.3

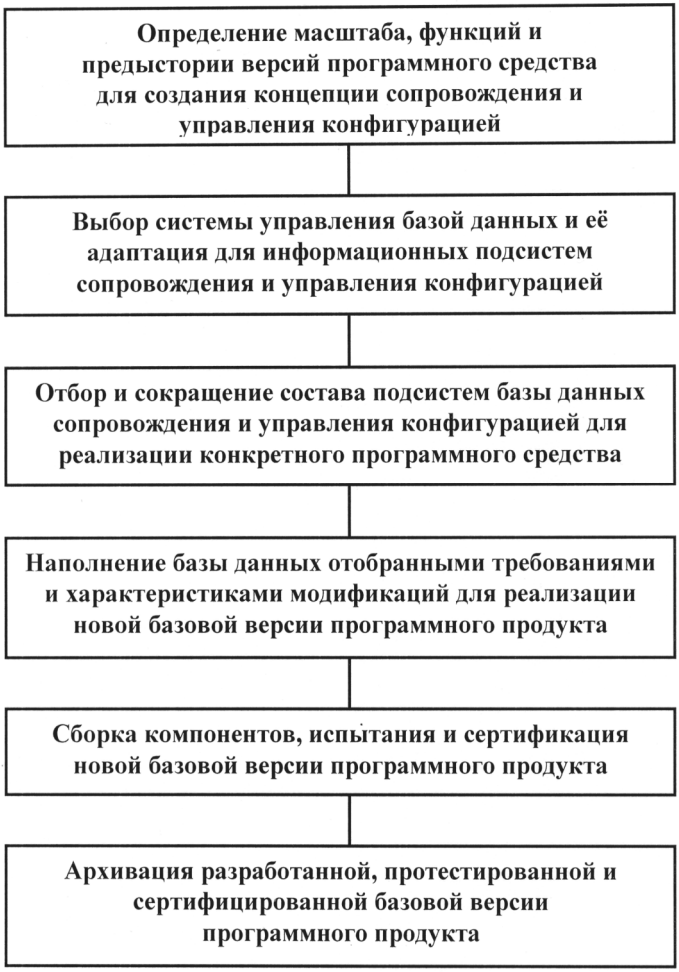

Рис. 2.4

Необходимо гарантировать сохранность версий изменений, с учетом их важности для результатов всего проекта. Особенно защищенным 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.

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. 2.3 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. 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 управление сопровождением и конфигурацией ПС***, а также категории ответственных лиц за их поэтапную реализацию, контроль и сохранность информации.

Оценка размера программного продукта является весьма важной, поскольку она является частью одной наиболее сложной задач возможных модификаций проекта и установки реальных ***ориентиров затрат*** на сопровождение и УК для заказчика. Нереальные ожидания, основанные на неточных оценках требуемых ресурсов, представляют собой одну из ***частых причин провала сопровождения проектов*** вследствие превышения ограничений ресурсов. Так как, величина и достоверность определения размера возможных модификаций ПС, составляют ключевой фактор последующего анализа влияния ограничений ре-

сурсов, то целесообразно применять несколько доступных методов для его оценивания [4, 26]. Конкретизация функций, структуры, состава компонентов ПС, позволяет более достоверно определить размеры возможных модификаций групп программ и, суммируя их, оценивать размер ресурсов, необходимых для сопровождения комплекса программ. Особенно сильно на снижение суммарных затрат на сопровождение влияет использование готовых компонентов и документов из предшествующих разработок. При анализе аналогов могут быть выделены компоненты, пригодные для повторного применения в новом проекте или в версии программного продукта. Это позволяет оценить возможную долю использования готовых компонентов и тем самым определить эффективный размер комплекса программ и документов, подлежащий модификациям и размер информации в базе данных.

При наличии прототипов проекта и ряда детальных спецификаций требований, для создания новой базовой версии может отсутствовать необходимость её системного, предварительного или детального проектирования, а разработка и тестирование программных компонентов выполняться в процессе расширения предшествовавшей базовой версии ПС. Тем самым процессы и Руководства управления конфигурацией при развитии и совершенствовании версий ПС могут формализоваться, начиная с некоторых промежуточных этапов жизненного цикла комплекса программ.

Выше, подчеркивалось, что проводимый в книге анализ сопровождения программных средств в основном ориентирован на жизненный цикл сложных крупномасштабных проектов комплексов программ высокого качества [25, 45, 49, 50]. Многие проекты ПС являются более простыми, и их процесс 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. 2.4). Известные функции, потенциальные пользователи и концепции существующих версий ПС позволяют прогнозировать направления совершенствования и уменьшения моди-

фикаций для нового проекта или базовой версии, имеющих прототип. При этом, возможно, потребуется выбор новой СУБД и её адаптация для реализации совокупности нового сокращенного набора подсистем обеспечения сопровождения. Дальнейшая адаптация Руководства сопровождения проекта ПС может включать отбор и уменьшение состава компонентов базы данных, а также исключение некоторых требований и характеристик модификаций в них, для упрощения реализации конкретного проекта ПС в выделенных подсистемах БД.

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. 2.4).

В системах и комплексах программ реального времени в ряде случаев могут отсутствовать компоненты административного управления, а также сокращены требования и функции пользователей. В программах административных систем может доминировать деятельность администраторов, действующих совместно с оперативными пользователями основных функций программного продукта. В некоторых автоматизированных системах реального времени программный продукт является органическим компонентом системы управления и внешней среды, и оперативное сопровождение должно выполнять в основном контрольные функции инсталляции и оценки целостности функционирования программного продукта в системе. Эти особенности сопровождения требуют более тщательной его подготовки, так как не всегда пользователи обладают достаточно высокой квалификацией и для эффективной эксплуатации сложного программного продукта требуется полное и детальное изложение содержания этих документов сопровождения и управления конфигурацией.

Эффективность процессов разработки и совершенствования версий ПС в значительной степени определяется их организацией и информационным обеспечением баз данных. Регламентированное конфигурационное управление и сопровождение повышает качество проекта ПС, сокращает непроизводительные затраты труда и времени на каждом этапе жизненного цикла комплекса программ, обеспечивает унифицированный, гибкий процесс модификации и развития множества версий проектов, платформ и технологий ИС [25, 40, 41]. Выбор технологии разработки программ и СУБД в значительной степени зависит от перспективы последующего сопровождения и совершенствования ПС. Контроль среды поддержки жизненного цикла ПС состоит в обеспечении гарантий того, что инструментальные средства, используемые для создания и модификации программ, – идентифицируются, контролируются и могут быть получены из соответствующих источников. Для этого должна оцениваться возможная длительность сопровождения и тираж различных версий у пользователей ПС, которые следует учитывать при выборе технологии, баз данных и инструментария для разработки компонентов и комплекса программ в целом.

2.3. Документирование и архивирование при сопровождении и управлении конфигурацией программных средств

Организация документирования при сопровождении и управлении конфигурацией комплексов программ, состоящих из многих компонентов, каждый из которых может иметь разновидности или версии, должна обеспечивать учет их функций и интерфейсов для управляемого и контролируемого развития структуры, состава компонентов и функций. Для этого необходима точная и достоверная документация о состояниях ПС и его компонентов, всех предполагаемых и выполненных изменениях, позволяющая установить текущее состояние версий программ в любой момент времени и историю их развития [4, 29, 41].

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 информационных 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 (см. Приложение, стандарты **ISO 9294, ISO 15910, РД 50-34.608**).

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;
- надежно учитывать и регистрировать все анализируемые, подготавливаемые и проведенные изменения в версиях ПС;
- снабжать руководителей и исполнителей проекта обоснованной, обобщенной и детальной информацией для принятия решений на изменения программ, а также для контроля выполнения принятых решений;
- обеспечивать заказчиков и пользователей объективными сведениями о наиболее существенных корректировках и о новых версиях программного продукта.

Стандарт **ISO 9126** регламентирует ***характеристики качества*** программного продукта, его изменений и документов. Один из шести основных показателей качества жизненного цикла ПС (см. п. 3.3), 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 [1, 23, 45]. В жизненном цикле ПС эта 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 воспри-

ятие 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.

Ниже основное внимание сосредоточено на ***анализе содержания и практичности документации при сопровождении ПС*** и только косвенно отмечаются некоторые особенности практичности эксплуатационной документации, связанные с сопровождением. 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. Следует учитывать, что малый объем документации может снизить качество и полноту отражения модификаций и использования новых функций сложного ПС, а очень большой объем – также может ухудшить сопровождение и управление конфигурацией, вследствие трудности выделения из множества второстепенных деталей документов и освоения, наиболее существенных свойств и особенностей изменений версии программного продукта.

В практичности следует учитывать всё разнообразие характеристик внешней среды сопровождения и 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 совершенствования и применения новых версий ПС. В эту группу показателей входят атрибуты с различных сторон отражающие функциональную понятность, удобство освоения или простоту разработки и использования модификаций. Оценки практичности зависят не только от собственных характеристик версий ПС, но также от организации и адекватности документирования процессов их модификации.

Практичность – применимость информации баз данных – зачастую значительно определяет функциональную пригодность и полезность расширения функций и применения новых версий БД 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 сопровождения входят субхарактеристики и атрибуты с различных сторон отражающие функциональную понятность, удобство освоения, системную эффективность и простоту использования данных. Практичность зависит не только от собственных характеристик информации БД, но также от организации и адекватности документирования процессов их изменения и эксплуатации.

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. Понятность зависит от качества документации и субъективных впечатлений от дополнительных, новых функций и характеристик программного продукта. Ее можно описать качественно четкостью функциональной концепции сопровождения и управления конфигурацией, широтой демонстрационных возможностей, полнотой, комплектностью и наглядностью представления в документации возможных изменений и новых функций, особенностей их реализации и 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. Кроме того, эта

субхарактеристика должна отражать распознаваемость модифицируемых параметров и адаптируемость версии комплекса программ к конкретной среде и условиям применения. Она должна обеспечиваться корректностью и полнотой описания исходной и результирующей информации, а также всех деталей применения баз данных для сопровождения и управления конфигурацией.

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. Она соответствует управляемости, устойчивости к ошибкам и согласованности с ожиданиями и навыками специалистов процессов сопровождения. Эта субхарактеристика должна учитывать физические и психологические особенности сопроводителей и отражает уровень контролируемости и комфортности условий модификации ПС, возможность предотвращения вторичных ошибок при корректировках. Должны обеспечиваться простота управления модификациями функций ПС и достаточный объем параметров управления, реализуемых по умолчанию, информативность сообщений, наглядность и унифицированность управления экраном, а также доступность изменения функций и минимум операций, необходимых для запуска определенного задания и анализа результатов корректировок ПС и компонентов. Кроме того, удобство использования характеризуется рядом динамических параметров: временем ввода и отклика на задание, длительностью решения типовых задач, временем на регистрацию результатов изменений.

Простоту модификации 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. Однако некоторые атрибуты этой субхарактеристики доступны для более полной количественной оценки путем измерения трудоемкости и длительности соответствующих процессов подготовки и обучения квалифицированных специалистов к полноценному и эффективному сопровождению и управлению конфигурацией базы данных изменений и версий 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. Атрибуты изучаемости зависят от возможности предварительного обучения и совершенствования знаний в процессе сопровождения, от возможностей оперативной помощи и подсказки при использовании модификаций и версий ПС, а так же от полноты, доступности и удобства использования руководств и документов по сопровождению. Качество изучаемости ПС зависит от внутренних свойств и сложности сопровождаемого комплекса программ, а также от субъективных характеристик квалификации конкретных специалистов.

На значения изучаемости существенно влияют демонстрационные возможности справочных средств обучения сопровождению, качество и объем документации, а также электронных учебников, которые можно оценивать соответственно по числу страниц документов или занятых учебниками килобайтов памяти на ЭВМ. Изучаемость можно отражать трудоемкостью и продолжительностью изучения специалистами соответствующей квалификации, методов и инструкций управления конфигурацией, применения версий и модификаций ПС. Эти атрибуты может характеризовать трудоемкость от единиц до сотен человекочасов и продолжительность от единиц до тысяч часов, необходимых для освоения квалифицированного сопровождения особенно сложных модификаций комплексов программ. Естественно, для создания учебных пособий необходимы определенные затраты труда и времени разработчиков документов, которые в некоторой степени пропорциональны сложности возможных новых функций в версиях ПС. Изучаемость информации БД зависит от внутренних свойств и сложности структуры БД, а также от субъективных характеристик квалификации конкретных специалистов сопровождения.

В ***документах сопровождения*** целесообразно использовать ***иерархическую систему идентификации*** версий ПС, его компонентов и описаний данных. Каждый модуль или массив переменных в своем идентификаторе или описании должны содержать обозначения компонентов более высокого уровня, (напри-

мер, документов функциональной группы программ и комплекса), а также учитывать очередность выполненных модификаций и номер конкретной версии. Присвоение базовых идентификаторов для наиболее крупных составляющих ПС, вплоть до документов утвержденных версий модулей и массивов переменных, целесообразно осуществлять централизовано, группой специалистов конфигурационного управления. Некоторый произвол в идентификации документов компонентов может допускаться только в процессе предварительной подготовки корректировок программ и данных до комплексирования их в базовую версию программного продукта, предъявляемую на испытания.

План и поддерживающее его ***Руководство по документированию сопровождения и конфигурационного управления*** конкретного крупномасштабного проекта ПС (рис. 2.5) 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.

Для управления конфигурацией, развитием комплекса программ и испытаний в базе данных документов должны собираться сведения, которые состоят из следующих ***функциональных частей*** [21, 40, 47]:

- 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 предлагаемых 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;

Рис. 2.5

- 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 другое и, возможно, из одного сектора базы данных проекта в другой. Первичные сообщения от испытателей или пользователей о дефектах и ошибках следует регистрировать в специализированных таблицах, отражающих их формализованные параметры, а также на содержательном уровне вместе с тестовыми данными, при которых зарегистрирован дефект. Также на содержательном уровне или в виде спецификации требований необходимо регистрировать в базе данных предложения по совершенствованию качества программ. По результатам анализа и тестирования от специалистов управления конфигурацией эти данные получают признаки отвергнутых изменений или подлежащих дальнейшей проработке. Отвергнутые изменения через некоторое время могут исключаться из состава предлагаемых изменений.

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 ***Отче-***

тов 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 должен 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. 2.3). Для каждого изменения должны документироваться содержательная аннотация, а также общие характеристики и достигнутые на испытаниях показатели качества очередной базовой 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 большими 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.

Представленные общие рекомендации по документированию ПС формализованы ниже набором ***шаблонов конкретных документов*** для сопровождения и управления версиями комплексов программ. Рекомендуемый состав и содержание 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 комплекса приведенных документов. Кроме них, пользователям должен поставляться с каждым программным продуктом комплект эксплуатационной документации, оформленный в соответствиями с требованиями стандартов и особенностями проекта ПС [29, 41, 64].

Шаблоны документов сопровождения и управления конфигурацией версиями программного продукта. К ряду документов по решению руководителя сопровождаемого проекта или компонента ПС целесообразно перед функциональной частью публиковать дополнительно, не отмеченный в шаблонах, ***титул-идентификатор документа*** содержащий:

- наименование-идентификатор предприятия разработчика ПС;

- наименование-идентификатор заказчика сопровождения системы и программного продукта;
- наименование-идентификатор системы и внешней среды;
- наименование-идентификатор сопровождаемого программного продукта;
- идентификатор компонента или модуля модифицируемого программного средства;
- идентификатор версии компонента или модуля программного средства;
- идентификатор автора или ответственного за разработку и за модификацию компонента и документа;
- дату и время выпуска версии или последнего изменения документа;
- некоторые общие сведения или комментарии о процессе, продукте или документе по выбору автора, руководителя проекта или заказчика управления конфигурацией.

1. Описание среды жизненного цикла и конфигурации программного средства:

- аппаратная и программная среда разработки, генерации, повторной верификации или модификации ПС на протяжении всего жизненного цикла;
- инструментальные средства разработки ПС: компиляторы, редакторы связей и загрузчики, средства обеспечения целостности данных;
- среда и методики верификации и тестирования компонентов, используемые при корректировке программного средства;
- аттестованные инструментальные средства обеспечения и корректировки жизненного цикла ПС и соответствующая документация об аттестации этих средств;
- идентификаторы конфигурации функциональных компонентов и программного средства; исходные тексты программ и исполняемый объектный код;
- ранее разработанные компоненты ПС, если они используются в данном программном средстве;
- комплекс технологических документов обеспечения жизненного цикла ПС;
- инструкции для компоновки исполняемого объектного

кода, инструкции для компилирования и редактирования связей; процедуры, используемые для восстановления при регенерации, тестировании или модификации компонентов и ПС;

- контроль целостности данных для исполняемого объектного кода, документы управления конфигурацией, генерируемые в процессе управления конфигурацией, включая отчеты управления конфигурацией, указатели состояния конфигурации ПС и среды жизненного цикла комплекса программ.

2. План сопровождения версий программного продукта:

- введение:
  * причины необходимости сопровождения программного продукта;
  * место проведения сопровождения;
  * описание сопровождаемой системы;
  * определение исходных состояний программного продукта;
  * определение организации, проводящей сопровождение программного продукта;
- концепция сопровождения программного продукта:
  * описание концепции;
  * описание уровня поддержки системы;
  * установление периода, длительности сопровождения программного продукта;
  * адаптация (практическое применение) процессов сопровождения;
- организационные работы по сопровождению программного продукта:
  * роли и обязанности сопроводителя до поставки программного продукта по: определению инфраструктуры процесса; установлению процесса обучения; установлению процесса сопровождения;
  * роли и обязанности сопроводителя после поставки программного продукта по: анализу дефектов и модификаций; реализации изменений; рассмотрению и принятию модификаций; переносу программного средства в новую среду; снятию программного средства с эксплуатации; решению проблем – ликвидации

дефектов; обучению персонала (сопроводителя и пользователя); усовершенствованию процесса;

* роль пользователя по: приемочным испытаниям; взаимосвязи (интерфейсам) с другими организациями;

- ресурсы:
  * состав специалистов для сопровождения конкретного программного продукта;
  * определение инструментальных программных средств, необходимых для поддержки эксплуатации системы с учетом системных требований и требований к среде программной инженерии и тестирования;
  * определение технических средств, необходимых для поддержки эксплуатации системы с учетом системных требований и требований к среде программной инженерии и тестирования;
  * планы обеспечения качества; управления проектом; управления конфигурацией; проведения верификации и аттестации; процедуры тестирования и отчеты о тестировании; обучения; руководства по сопровождению; руководство пользователя;
- процессы сопровождения, выполняемые сопроводителем и адаптация применения сопровождения к условиям программного продукта;
- обучение: определение уровня обучения, необходимого для сопроводителей и пользователей;
- протоколы и отчеты по сопровождению программного продукта:
  * перечень запросов пользователя по сопровождению, предложения о модификациях и отчеты о проблемах – дефектах;
  * приоритеты запросов, предложений, отчетов;
  * контрольные данные, собранные при работах по сопровождению программного продукта.

3. План управления конфигурацией программного средства:

- описание среды управления конфигурацией, включая процедуры план-графика, инструментальные средства, методы, стандарты, организационную ответственность и интерфейсы специалистов;

- описание процессов управления конфигурацией в жизненном цикле ПС, которые обеспечивают реализацию целей данного процесса;
- компоненты изменения конфигурации, которые должны быть идентифицированы, методы идентификации документов жизненного цикла ПС, связь идентификации компонентов, ПС и системы;
- содержание и идентификация сообщений об изменениях ПС, процедуры регистрации сообщений о дефектах и взаимодействие отчетов о дефектах и контроле изменений;
- контроль изменений и базовая версия, обеспечивающие целостность компонентов конфигурации и базовой версии программного продукта;
- методы оценки и определения приоритетов в устранении дефектов, утверждении изменений, реализации решений об изменениях и связь этих методов с отчетами о дефектах и контролем изменений;
- отчет о текущем состоянии конфигурации, определение места хранения информации, как она воспроизведена для отчетов и когда она будет доступна специалистам;
- архивирование, получение из архива и выпуск официальной базовой версии программного продукта, контроль целостности, способы внесения информации в архив и получения из архива, метод и полномочия для выпуска версии комплекса программ;
- контроль среды жизненного цикла, инструментальных средств для разработки, комплексирования, верификации, тестирования и загрузки ПС;
- контроль целостности технологических документов жизненного цикла ПС;
- определение состава документов жизненного цикла ПС, генерируемых в процессе управления конфигурацией, включая отчеты управления конфигурацией компонентов, указатели состояния конфигурации и среды жизненного цикла ПС.

4. Программа управления конфигурацией программного продукта:

- введение:

* описание системы или объектов конфигурации, к которым применяется Программа;
* график с указанием этапов и сроков выполнения, наиболее важных для управления конфигурацией видов деятельности;
* цель и область применения Программы управления конфигурацией программного продукта;
* применяемые исходные документы и их приоритетность;

- политика и процедуры:
  * политика в области практических методов управления конфигурацией и связанные с ним управленческих дисциплин;
  * организация управления конфигурацией вместе с согласованными обязанностями совета по конфигурации, комитетов, групп консультантов поставщика и других заинтересованных организаций;
  * согласованные критерии выбора объектов – единиц конфигурации (ЕК);
  * частота составления отчетов для внутреннего пользования и для заказчика, их распределение и управление ими;
  * согласованная терминология;
- идентификация конфигурации:
  * правила нумерации (ЕК) применительно к техническим требованиям и изменениям;
  * установленные конфигурационные базы, графики работ и типы документов, на которые должны даваться ссылки;
  * указания на использование и присвоение номеров или другой идентификации прослеживаемости;
  * процедуры выпуска версий программного продукта;
- контроль за конфигурацией программного продукта:
  * положения, касающиеся организации, состава и срока полномочий совета по конфигурации и его связей с аналогичными советами заказчика и субподрядчиков;
  * процедуры по контролю за изменениями (ЕК) до установления контрактной конфигурационной базы программного продукта;

* процедуры рассмотрения изменений, начиная с заявки и кончая утверждением изменения после внесения в объект конфигурации;

- отчетность о статусе конфигурации программного продукта:
  * процедуры по сбору, регистрации, обработке и сохранении данных, необходимых для составления отчетов о статусе конфигурации программного продукта;
  * определение содержания и формата всех отчетов об уп-равлении конфигурацией программного продукта;
- проверка конфигурации программного продукта:
  * перечень проводимых проверок и их взаимосвязь с графиками работ по проекту;
  * используемые процедуры проверки конфигурации программного продукта;
  * полномочные органы и используемые дисциплины утверждения модификаций программного продукта;
  * форматы отчетов о проверках.

5. Отчеты пользователей о выявленных дефектах и предложениях по корректировке комплекса программ:

- рекомендации пользователям по выявлению и регистрации условий проявления и содержания дефектов эксплуатируемых версий программного продукта;
- идентификация и регистрация аномального поведения программного продукта, несогласованности процессов с планами и стандартами разработки, недостатки документации жизненного цикла ПС;
- описание дефекта, достаточное для его понимания и устранения, описание возможных корректирующих действий, предназначенных для устранения зарегистрированного дефекта;
- идентификатор пользователя, представившего отчет о дефектах и/или предложениях;
- дата фиксирования дефекта или предложения на изменение ПС;
- номер и параметры адаптации пользовательской версии программного продукта, на которой обнаружен дефект;

- идентификация компонента конфигурации и/или этапа жизненного цикла ПС, где был обнаружен дефект;
- подробное описание сценария и исходных данных, при которых выявлен дефект и документы результатов его регистрации;
- предположение о причине, вызвавшей проявление дефекта;
- идентификация компонента конфигурации, который необходимо модифицировать, или описание процесса, который должен быть изменен;
- описание возможных корректирующих действий, предназначенных для устранения зарегистрированного дефекта;
- предложение по модификации ПС и его компонентов для устранения дефекта или совершенствования функционирования программ.

6. Описания выявленных дефектов и предложений по совершенствованию функций версии программного средства:

- отчеты пользователей о дефектах и предложениях;
- идентификатор разработчика, которому передан отчет пользователя для анализа дефекта или предложения;
- дата начала анализа отчета пользователя;
- идентификатор сценария проявления повторяемости дефекта на пользовательской версии и необходимости дальнейшего анализа дефекта на базовой версии программного продукта;
- тесты, исходные данные и сценарий, при которых проявляется выявленный дефект;
- результаты анализа предложения на выполнение изменения, причин и источника выявленного дефекта;
- рекомендации о возможных способах устранения дефекта или о реализации предложения по совершенствованию программ и базы данных;
- оценки сложности, трудоемкости, эффективности и срочности модификации программ и базы данных;
- оценки влияния предлагаемых изменений на возможность эксплуатации версий ПС, имеющихся у пользователей.

7. Описания подготовленных и утвержденных корректировок и обобщенных характеристик новой базовой версии программного продукта:

- идентификатор специалиста, который разработал модификацию компонентов программ и базы данных;
- дата разработки предлагаемой модификации;
- причина изменения программ и/или базы данных (дефект, совершенствование);
- содержание изменений программ и базы данных;
- содержание изменений документации на компоненты или версию программного продукта;
- корректность результатов тестирования базовой версии программного продукта с разработанными изменениями;
- дата и ответственное лицо, утвердившее реализацию модификации версии программного продукта;
- квалификация решения на изменение: частная модификация вследствие исправления дефекта или издание новой версии программного продукта;
- результаты испытаний модификации и обобщенные характеристики базовой версии программного продукта после внесения изменений;
- решение по распространению пользователям результатов проведенной модификации или новой версии программного продукта;
- решение по целесообразности сохранения сопровождения предшествующих версий программного продукта;
- адрес хранения корректировок, документов и квалификационных тестов выполненной модификации и/или новой базовой версии программного продукта;
- идентификация новой конфигурации, протоколы об установлении новой базовой версии программного продукта и её регистрации в архиве;
- архивирование, получение из архива и выпуск официальной версии: контроль целостности базовой версии программного продукта, способы внесения информации в архив и получения из архива, метод и полномочия для выпуска версии комплекса программ;

- отчеты об истории выполнения изменений версий программного продукта;
- протоколы о передаче новой версии программного продукта в архив;
- протоколы о выпуске новой версии программного продукта для применения пользователями.

8. Извещение пользователям о выпуске новой версии программного продукта и/или о прекращении сопровождения предшествующей версии:

- краткое обоснование причин модификации или прекращения сопровождения версии программного продукта;
- описание содержания и характеристики основных изменений в новой версии программного продукта;
- рекомендации по корректировке, приобретению или замене пользовательской версии программного продукта.

9. Описание новой базовой версии программного продукта:

- идентификация системы и комплекса программ, к которым применяется данный документ, включая регистрационные номера процедур утверждения конфигураций и номера базовых версий программного продукта;
- краткий обзор назначения и спецификации требований новой версии системы и программного продукта, особенности истории разработки, эксплуатации и сопровождения системы и комплекса программ, идентификация заказчика, пользователей, разработчика, а также организации, осуществляющей сопровождение, регистрацию текущих и планируемых мест установки системы и программного продукта для пользователей;
- идентификация физических носителей, содержащих новую зарегистрированную и утвержденную базовую версию программного продукта и связанную с ней документацию;
- идентификация комплекта новых зарегистрированных, утвержденных файлов, содержащих базовую версию программного продукта;
- перечень всех изменений, внесенных в файлы и документы после выпуска предыдущей базовой версии программного продукта;
- полный комплект документов новой базовой версии

программного продукта;

- инструкция по установке и инсталляции новой версии программного продукта.

10. План передачи и внедрения новой базовой версии программного продукта пользователям:

- общий обзор результатов разработки, комплекта требований и особенностей новой базовой версии системы и программного продукта; заказчиков, пользователей, разработчиков и организаций, осуществляющих сопровождение, запланированные рабочие места и перечень передаваемых документов;
- контроль комплектности и состояния аппаратного и программного обеспечения, а также другие ресурсов, необходимых для поддержки всего жизненного цикла передаваемого нового комплекса программ;
- запланированные сроки установки новой версии программного продукта у определенных пользователей;
- комплект системы файлов и документов, относящихся к передаваемой новой базовой версии программного продукта;
- детальное описание ресурсов, необходимых для инсталляции передаваемого программного продукта, требования к квалификации и составу персонала, чтобы специфицировать, разработать, документировать, тестировать, оценивать, контролировать, копировать и распространять новую базовую версию программного продукта;
- перечень рекомендуемых мероприятий, в том числе консультации и лекции, которые должен проводить разработчик в целях поддержки применения передаваемого нового программного продукта;
- описание процесса подготовки персонала, который будет осуществлять поддержку передаваемого программного продукта: тематика, дата, продолжительность и место проведения теоретических и практических занятий;
- порядок внедрения, включающий в себя все работы, необходимые при передаче и инсталляции новой версии программного продукта со стороны организаций, осуществляющих сопровождение.

***11. Отчет о результатах эксплуатации снятой с сопровождения базовой версии программного продукта и её архивации*:**

- причины и дата решения о прекращении сопровождения определенной базовой версии программного продукта и извещение пользователей;
- идентификатор ответственного лица, принявшего решение о прекращении сопровождения определенной версии программного продукта;
- дата и идентификатор лица, выполнившего архивацию определенной версии программного продукта;
- идентификаторы физических носителей информации архива, содержащих подлинники и дубликаты файлов и документов, снятой с сопровождения базовой версии программного продукта.

12. Отчет о результатах тиражирования базовых версий, конфигурациях и параметрах пользовательских версий программного продукта:

- идентификаторы базовых версий программного продукта, поддерживаемых сопровождением и распространяемых пользователям;
- адреса архивов, содержащих физические носители файлов и документации каждой базовой версии программного продукта распространяемой пользователям;
- краткая характеристика и адреса архивов, содержащих квалификационные тесты базовых версий программного продукта;
- перечень идентификаторов пользователей, которым передана на эксплуатацию определенная базовая версия программного продукта;
- идентификатор базовой версии, которая адаптировалась для эксплуатации каждым пользователем;
- параметры среды пользователя, на которые адаптировалась определенная базовая версия программного продукта;
- характеристики активности обращений пользователей к поставщику за консультациями и модификациями комплекса программ.

Глава 3

МЕТОДЫ И СРЕДСТВА ОБЕСПЕЧЕНИЯ СОПРОВОЖДЕНИЯ И УПРАВЛЕНИЯ КОНФИГУРАЦИЕЙ ПРОГРАММНЫХ СРЕДСТВ

3.1. Ресурсы, необходимые для обеспечения сопровождения и управления конфигурацией программных средств

Прогнозирование необходимых основных ресурсов – труда, времени и числа привлекаемых специалистов для сопровождения и управления конфигурацией 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 ***не предсказуемо*** и имеют большие флюктуации в зависимости от характеристик проекта, квалификации специалистов, применяемого инструментария, и ряда других трудно учитываемых факторов. Априори оценить и прогнозировать такие затра-

ты 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 управление конфигурацией 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.

В данном разделе акцент сделан на анализ трудоемкости, длительности и числе специалистов, необходимых для реализации локальных модификаций программных продуктов. Достаточно достоверное прогнозирование требований, ресурсов и длительности реализации модификаций комплексов программ возможно только при применении жестко регламентированной технологии сопровождения. В этом случае процесс разработки изменений структурируется на четкие этапы (см. п. 1.4), которые характеризуются определенной долей в общих затратах и сроках выполнения модификаций. После разработки технического задания на изменения в процессе структурного проектирования и распределения ресурсов ЭВМ, размер изменений размера комплекса программ и используе-

мой памяти для модификации ПС может определяться с точностью 20 – 30%, что позволяет значительно повысить достоверность прогнозирования общих затрат и их составляющих при сопровождении. После уточнения технологии, средств автоматизации сопровождения и технологических ЭВМ, появляется возможность достоверно учесть эти исходные данные для подготовки уточненного сценария разработки изменений. Регистрация этих данных может служить дополнительным ориентиром для оценки ***полных ресурсов и стоимости разработки модификаций***. Таким образом, перед началом наиболее трудоемких этапов сопровождения сложного ПС могут быть получены достаточно полные экономические исходные данные, которые позволяют планировать эти этапы, а также прогнозировать необходимые ресурсы и сроки [6, 52, 58].

Неопределенность оценок масштаба изменений комплексов и компонентов программ является одним из важнейших факторов, ***влияющим на требуемые ресурсы для сопровождение ПС***. Точность оценки размера корректировки или новой версии ПС значительно повышается после формулирования начальных требований спецификаций заказчиков и проведения анализа модификации. Величина возможного изменения размера каждого компонента при сопровождении может быть получена путем опроса экспертов, разрабатывавших подобные системы, либо путем использования данных опроса потенциальных разработчиков подобных систем.

Первым этапом прогнозирования необходимых ресурсов при сопровождении является создание ***комплекса требований*** к конкретной модификации функций программ, 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 ПС;
- 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 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 конкретно 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 модификаций и сопровождения комплекса программ.

При технико-экономическом обосновании (ТЭО) сопровождения и управления конфигурацией 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 [4, 19, 26]. 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.

При оценивании масштаба модификаций следует определить приоритеты новых функций для установления состава работ, согласованного между заказчиком и разработчиком корректировок, которые ***обязательно*** должны быть выполнены, и для определения нового базового уровня масштаба конкретной версии продукта. Сокращение масштаба модификаций 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 изменений, чтобы обеспечить как качество, так и своевременность разработки новой базовой версии программного продукта. При этом следует учитывать, что:

- именно ***заказчики несут финансовую ответственность*** за выполнение обязательств по сопровождению перед внешними клиентами и пользователями (пусть и в несколько сокращенном, при необходимости, масштабе);
- комплекс программ, его важнейшие, новые функции и удовлетворяемые требования модификаций принадлежат заказчику, а ***не коллективу сопровождения или поставщику новых версий программного продукта***.

Привлечение заказчика помогает менее болезненно решать проблемы управления масштабом изменений проекта и реали-

зуемыми новыми функциями с учетом ограничений ресурсов. Любые дополнительные функции за пределами базовых, которые реализует разработчик модификаций, будут восприняты заказчиком как превзошедшие ожидания. В зависимости от этапа сопровождения сложного комплекса программ и достоверности исходных данных о характеристиках и особенностях изменений ПС целесообразно выбирать и применять разные методики и сценарии технико-экономического обоснования проекта и прогнозирования затрат. С самого начала работ над сопровождением ПС важно вести постоянный учет данных о его ***действительной*** трудоемкости, стоимости и развитии затрат на корректировки и сравнивать эти данные с реальными оценками характеристик изменения ПС по причинам:

- несовершенства исходных данных для оценивания затрат (оценки размера, рейтинги влияния факторов на модификации) определяют важность для руководителя проекта пересматривать их оценки учитывая новую информацию, чтобы обеспечить более реальную основу для дальнейшего управления сопровождением;
- вследствие несовершенства методов оценивания ресурсов на сопровождение ПС следует сравнивать оценки с действительными значениями и использовать эти результаты для улучшения прогнозирования затрат;
- проекты ПС имеют тенденцию к изменению характеристик и экономических факторов и руководителю проекта необходимо идентифицировать эти изменения и выполнять реалистичное обновление оценок затрат при сопровождении.

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 продвижения 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 из-

менение некоторых выделяемых ресурсов, либо прекращение модификации данного ПС.

Для уменьшения возможных методических ошибок оценок затрат на сопровождение ПС ***необходимо определить основные понятия и единицы измерения*** масштаба изменений комплексов программ. Поэтому модификации ПС должны начинаться с прогнозирования размеров изменений или новой версии программного продукта, оценивание, достоверность и ошибки которых 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 большими 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.

Размер или масштаб комплекса программ и его изменений в настоящее время приводится в публикациях в различных единицах, что может отличать их численные значения для одних и тех же модификаций программ в несколько раз [4, 13, 19]. Влияние ошибок единиц измерения размера корректировок программ можно значительно уменьшить, если учесть их принципиальные особенности и, прежде всего, выделить ***две группы единиц измерения масштаба*** [24]:

- группу, характеризующую ***размер*** исходных или модификаций текстов программ, которые ***разрабатываются и анализируются человеком***, отражающую сложность, трудоемкость и длительность создания корректировок ПС и его новых компонентов;
- группу, отражающую ***размер*** модифицированных программ и данных***, размещаемых в реализующей (объектной) ЭВМ***, и характеризующую объем памяти и производительность ЭВМ, необходимые для рабочего функционирования и исполнения сопровождаемого комплекса программ после формирования новой базовой версии программного продукта в соответствие с его назначением.

Эти две группы единиц отражают размер корректировок программ, с разных позиций, и должны использоваться в зависимости от целей анализа масштаба модификации ПС. Хотя между ними есть корреляция, но в общем случае дополнительные размеры программы, измеренные различными группами единиц, трудно сопоставлять из-за наличия между ними промежуточного звена – преобразователя (транслятора) с различными, не полностью определенными и трудно учитываемыми характеристиками. Это обусловлено ***особенностями трансляторов***, которые преобразуют исходные тексты программ, разработанные человеком – программистом, в разделы памяти реализующей ЭВМ, заполненные командами, константами или выделенные под данные, а также особенностями языков программирования, структуры и содержания машинных команд.

Оценки масштаба крупных изменений ПС должны быть проанализированы и скорректированы для установления в концепции и договоре между заказчиками и сопроводителями ***компромиссного масштаба изменений***, допустимых при разработке спецификаций требований к модификациям с необходимым качеством. Некоторые требования могут потребовать изменения (обычно увеличения) масштаба, и соответственно ресурсов на этапах проектирования корректировок версии для обеспечения возможности реализации требуемой обновленной функциональной пригодности. Таким образом, ***требования*** к модификациям функциональной пригодности и конструктивным характеристикам ПС ***должны быть согласованы с масштабом*** и всеми видами ресурсов, требующимися для их реализации. Для этого

необходимо использовать адекватные методы и единицы измерения масштаба модификаций ПС [13, 19, 57].

Первая группа задач при оценивании масштаба корректировок ПС состоит в выборе базовых унифицируемых единиц измерения исходного текста и исполняемых (объектных) программ, при применении которых имеется ***наибольшая корреляция*** этих видов измеряемых размеров программ. С этих позиций наиболее адекватной единицей измерения размера изменений программ является число операторов на машинно-ориентированном ассемблере, которое имеет корреляцию с числом команд в объектном коде реализующей ЭВМ, близкую к единице. В этом случае при сопоставлении измеренных размеров программ влияет единственный фактор – архитектура (одна и та же) реализующей ЭВМ. Этот фактор может учитываться путем небольших коэффициентов перевода, определяемых путем анализа особенностей структуры и системы команд ЭВМ, а также конкретного ассемблера.

Недостатки измерения ***размеров крупных модификаций текстов*** комплексов программ, прежде всего, отражаются на ошибках определения трудоемкости и длительности их разработки. Однако исходные тексты содержат конструкции, которые не исполняются при рабочем функционировании программ: комментарии разных уровней, указания, заголовки и дополнительные описания, требующие для подготовки определенного труда специалистов. Эта группа единиц измерения размера ПС в той или иной степени отражает сложность разработанного изменения текста программ, которые могут оцениваться ***числом***:

- операторов языка программирования уровня ассемблера;
- строк текста программы на языке программирования высокого уровня или строк кода в терминах Lines of code – LOC;
- функциональных точек.

Основной труд специалистов вкладывается в разработку текста модификации программ, и желательно, чтобы выбранная единица измерения была бы в наибольшей степени ***адекватна трудоемкости сопровождения*** (соответственно бюджету). При этом должна учитываться не только трудоемкость непосредственной подготовки текста корректировок и разработки новых компонентов программы, но и отражать трудоемкость комплексного создания новой версии всего сложного программ-

ного продукта. Кроме того, единица измерения размера должна быть наглядной и просто измеряемой.

Вторая группа единиц измерения размера программ отражает ресурсы памяти и производительности объектной, реализующей ЭВМ, которые необходимы ***для функционирования измененной версии ПС*** в процессе эксплуатации и определяют требуемые ресурсы ЭВМ. Размер программ, размещаемых в ЭВМ для исполнения после корректировок, влияет на характеристики и стоимость машин, которая зависит от объема памяти и производительности ЭВМ, что особенно важно учитывать в системах реального времени. Трудоемкость разработки при этом отходит на второй план и основную роль играют затраты на эксплуатацию ПС. Для измерения размера сопровождаемых комплексов программ при их реализации на ЭВМ ***применяются следующие единицы***:

- байты, занятые текстом программ в объектном коде и информацией базы данных в памяти ЭВМ;
- команды (операции) реализующей ЭВМ, составляющие текст исполняемой программы в объектном коде;
- объем файлов информации баз данных.

На основе оценок масштаба модификаций ПС возможна ***организация и распределение ресурсов для сопровождения программного средства.*** Требования к организационным ресурсам сопровождения программного средства обычно устанавливает заказчик при помощи поставщика – разработчика. При этом должны быть определены требования к ресурсам персонала сопровождения, среды и финансов (см. **ISO 14764**). Для этого могут применяться ***две методики*** [4, 26, 45]:

- первичная оценка ТЭП при подготовке концепции и технического задания на новую версию комплекса программ на основе экспертных данных размера модификаций ПС, производительности труда или стоимости разработки одной строки текста новых программ или прототипов;
- прогнозирование ТЭП при предварительном и детальном проектировании модифицированных версий ПС на базе расчетных значений трудоемкости и длительности разработки изменений комплекса программ по данным модели COCOMO с учетом влияния различных дополнительных факторов [45].

В первой ***методике*** реализован метод прогноза ТЭП с учетом ***экспертной оценки минимального числа факторов***. Дан-

ная методика экспертной оценки затрат на модификации, может применяться, когда определены цели и общие функции изменения версии ПС, сформулированные в концепции и первичных требованиях с достоверностью около 30 – 40%. Учитывая полноту и достоверность доступных характеристик и требований к сопровождению и управлению конфигурацией ПС, должны быть определены ***цели и возможная достоверность технико-экономического обоснования затрат на сопровождение***. Наименее точный из перечисленных факторов полностью определяет достоверность расчета технико-экономиче-ских показателей модификаций ПС поэтому желательно, чтобы значения точности экспертного оценивания перечисленных факторов были сбалансированы. При наличии перечисленных исходных данных и положительной оценке целесообразности экспертного анализа ТЭП сопровождения ПС может реализовываться ***методика, состоящая из следующих шагов*** (рис. 3.1):

Рис. 3.1

- экспертная оценка размера – масштаба, числа строк предполагаемого текста изменяемых программ, с учетом размера повторно используемых компонентов и характеристик возможного языка программирования (этапы 1.1 – 1.3);
- экспертная оценка возможной средней производительности труда специалистов при разработке изменений программ и/или стоимости разработки одной сроки текста новых программ версии ПС (этапы 2.1. – 2.3);

- расчет возможной полной трудоемкости и длительности требуемых модификаций версии программного продукта, а также среднего числа специалистов, необходимых для их реализации (этапы 3.1 – 3.3);
- обобщение основных технико-экономических показателей и полной стоимости сопровождения версии ПС, анализ результатов и технико-экономическое обоснование рентабельности продолжения сопровождения и управления конфигурацией программного продукта (этапы 4.1 – 4.2).

Достоверность прогнозов затрат ресурсов зависит, прежде всего, от ***точности экспертной оценки исходных данных***: размера – масштаба модификаций версии ПС, от достоверности экспертной оценки производительности труда специалистов или оценки стоимости разработки одной строки текста изменений программ. Экспертные оценки зависят от компетенции и объективности экспертов, их ***оптимистичности, пессимистичности***, от знания существенных особенностей проекта. Оценки одного эксперта трудно проверять и контролировать на достоверность. Из-за индивидуальных особенностей экспертов предпочтительным является оценивание характеристик сопровождения несколькими экспертами и получение средних оценок необходимых ресурсов на совещании группы экспертов для формирования единой оценки [37].

Экспертная оценка размера модификаций ПС наиболее сложная задача в этой методике. Точность оценок должна быть сбалансирована с достоверностью экспертного определения или выбора характеристик других компонентов, используемых при расчете ТЭП (удельной трудоемкостью или стоимостью строки изменения текста программы). Исходными данными является ***концепция проекта и комплекс требований*** к иерархическому набору приоритетов изменяемых функций, которые могут быть разбиты на предполагаемые фактические модификации компонентов ПС. В дальнейшем разбиение может детализироваться, формируя упрощенный или более точный уровень абстракции и взаимодействия модифицируемых компонентов.

Один из путей оценки размера модификаций ПС, заключается в сравнении его функциональных задач и свойств с уже существующими аналогами. Конечно, данный метод не является точным, поскольку при корректировках компонентов прототи-

пов могли использоваться различные языки программирования, относящиеся к разным областям приложений. Также могли применяться разнообразные алгоритмы, имеющие различные уровень сложности, совместно с не проверенными функциональными свойствами, а также с различной степенью адекватности моделирования реальности модификаций.

Экспертная оценка удельных затрат на изменение строки текста программ приводится для ряда примеров проектов ПС в [13, 41, 64]. По мнению некоторых специалистов, несмотря на появление новых методов и инструментальных средств разработки сложных ПС, средняя производительность при их создании и изменении за последние двадцать лет осталась почти неизменной и составляет около 3000 строк кода на одного разработчика ПС в год (порядка 250 строк на человеко-месяц). Это отражает то, что уменьшение времени, затрачиваемого на цикл разработки модификаций, не может быть достигнуто за счет значительного повышения производительности труда отдельных специалистов. Причем это слабо зависит от усовершенствований языка программирования, организационных усилий со стороны менеджеров, от наличия или отсутствия некоторых отдельных видов инструментария и автоматизации работ, хотя значительную роль играет увеличившаяся доля повторно используемых компонентов. На самом деле, при достаточно высоком уровне технологии (см. п. 4.1 CMM 3 – 4 уровень), большое значение может иметь возросший размер и сложность изменений функциональных задач комплексов программ, а также значительное повышение требуемого качества сопровождения ПС.

В ***качестве ориентиров*** при экспертной оценке требуемых ресурсов для сопровождения и управления конфигурацией можно использовать следующие данные ***средней трудоемкости*** разработки модификаций сложных комплексов программ:

- отдельные, разрозненные сведения в публикациях о методах, ресурсах и технологиях разработки комплексов программ;
- результаты статистических исследований технико-экономи-ческих характеристик ЖЦ проектов ПС, обобщенные в базовой модели COCOMO [45];

- результаты исследований экономических характеристик разработки отечественных проектов ПС, выполненных по программе ПРОМЕТЕЙ [20].

Так как разработчики сложных комплексов программ не заинтересованы раскрывать реальную экономику сопровождения ПС, и эти данные склонны относить к коммерческой тайне, опубликованные экономические характеристики носят отрывочный, не упорядоченный и не всегда достоверный характер. Кроме того, в публикациях обычно не представляются детальные сведения об особенностях и требованиях к качеству объекта разработки, применявшейся технологии и инструментальных средствах, характеристиках коллектива специалистов и других факторах. Поэтому такие данные трудно обобщать и использовать для прогнозирования экономики сопровождения ПС. Весьма общие данные опубликованы в виде широких диапазонов производительности труда [41]: для простых ПС – 8 LOC на человеко-день, и 4 LOC на человеко-день для сложных ПС. Там же приведены 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 полезно использовать ***как ориентиры*** при первичных определениях затрат на модификации ПС.

Более точные оценки производительности при разработке программ различного размера и классов на основе обобщения статистических данных множества проектов представлены в базовой модели COCOMO [45]:

- для встроенных комплексов программ (СРВ) 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;
- для программ административных систем размером 30 тысяч строк оценка производительности составляет около 220 строк на человеко-месяц, а для ПС размером 500 тысяч строк – 160 строк на человеко-месяц.

Эти данные находятся в середине представленных выше диапазонов и их целесообразно использовать при экспертной оценке полной трудоемкости разработки корректировок соответствующих ***новых версий*** ПС (этапы 3.1 – 3.3 на рис. 3.1). При

использовании готовых повторно используемых компонентов обобщенная производительность труда при сопровождении возрастает и зависит от доли таких компонентов в модификациях комплекса программ. В зависимости от характеристик объекта и условий выполнения модификаций возможен экспертный выбор величин производительности труда для последующего прогноза полной трудоемкости сопровождения ПС.

Эти данные можно использовать для оценки ***полной стоимости сопровождения*** конкретного ПС в течение заданного времени. Однако при этом необходимы удельные данные 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 модификации текста комплексов программ реального времени, составляющие около 100$ и более, а для административных систем – около 20 - 50$.

Экспертная оценка длительности разработки изменений сложных ПС может базироваться на экспериментальных графиках, или на формулах модели COCOMO. Основой для расчета длительности целесообразно использовать рассчитанную ранее трудоемкость разработки модификации ПС, от размера которой не линейно зависит длительность её реализации.

Требования к коллективу специалистов являются важным фактором, определяющим стоимость сопровождения (см. п. 3.2). При оценке ресурсов специалистов для сопровождения ПС обычно применяют два основных метода: параметрические модели и использование практических знаний. Лучшие результаты дает использование практических знаний при наличии соответствующих архивных опытных данных. При этом предполагают, что для оценки сопровождения используют стандартную методологию. Должно быть проведено отдельное исследование штатного состава персонала сопровождения. Экспертная оценка необходимого числа специалистов может рассчитываться путем деления полной трудоемкости разработки модификаций ПС на длительность их реализации. Для примера полной разработки крупного проекта ПС реального времени, размером 500 тысяч строк, не-

обходимое число специалистов достигает 160 человек [4], 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 сопровождения конкретного проекта.

Ресурсы инструментальной среды при сопровождении и управления конфигурацией ПС, определяют ряд специальных работ, для выполнения которых необходимы отдельные системы и средства (см. п. 3.5). Необходимо наличие отдельных сред разработки модификаций и сред тестирования корректировок (см. п. 3.4). 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 управления конфигурацией 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.

Финансовые ресурсы – для обеспечения эффективного сопровождения программного продукта сопроводитель должен получить от заказчика финансирование для:

- выплаты зарплаты персоналу;
- обучения персонала;
- ежегодного возобновления лицензий на сопровождение инструментальных программных средств;

- командировок;
- публикации (издания) соответствующих документов;
- приобретения и модернизации технических и программных инструментальных средств разработки корректировок и тестирования ПС.

Практический опыт подсказывает, что стоимость сопровождения и управления конфигурацией ПС и способность сопроводителя выполнять их в значительной степени зависят от того, что требуется от заказчика и реализуется (или не реализуется) при первоначальной разработке программного продукта. В большинстве случаев сопроводитель не подключается к процессу разработки исходной базовой версии ПС по условиям договора или по другим причинам, особенно когда этот сопроводитель является третьей (независимой) стороной.

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 управлению конфигурацией 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 её 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.

3.2. Организация специалистов для сопровождения и управления конфигурацией программных средств

Важнейшим фактором при ***оценивании ресурсов, необходимых на сопровождение и управления конфигурацией в ЖЦ ПС*** являются люди – ***специалисты***, с их уровнем профессиональной квалификации, а также с многообразием знаний, опыта, стимулов и потребностей. Быстрый рост сложности, повышение требований и ответственности за качество комплексов программ привели к появлению ***новых требований к специалистам,*** обеспечивающим все этапы жизненного цикла ПС. Каждый специалист ограничен в своих возможностях, способностях и квалификации, что отражается на специфических дефектах результатов его деятельности. Специа-листы в соответствии со своей квалификацией и ролью в проекте вносят в него положительный вклад, а также специфические дефекты и ошибки, что целесообразно учитывать при прогнозиро-вании процессов сопровождения проекта (см. п. 1.2 и табл.1).

Разработка модификаций сложных ПС, особенно, на начальных и завершающих этапах, характеризуется ***высокой долей творческого труда***. Дефекты, трудоемкость и длительность отдельных операций и частных работ существенно зависят от индивидуальных особенностей их исполнителей и характеристик конкретного проекта. Отсюда принципиальной особенностью планирования сопровождения комплексов программ является необходимость активного участия руководителей и заказчиков при отборе и подготовке исполнителей изменений, а также при составлении планов на базе исследованных характеристик прототипов завершенных разработок ПС и их ***личного опыта***. Структура коллектива разработчиков модификаций, обычно, в значительной степени отражает структуру сопровож-даемого ПС. Особенно это заметно при сопровождении крупных комплексов программ размером 10^5 – 10^6 строк. При этом группы функциональных программ локализуются по соответствующим группам специалистов с целью минимизации связей и упрощения взаимодействия, как между группами специалистов сопровождения, так и между создаваемыми ими модификациями программ и документов.

Недостатки или отсутствие достоверного обоснования необходимых ресурсов специалистов для сопровождения проектов новых ПС, могут являться причинами острых конфликтов между заказчиками и разработчиками. Поэтому целесообразно ***активно привлекать заказчиков*** к управлению сопровождением и конфигурацией проектов, чтобы обеспечить своевременность разработки версий программных продуктов в условиях ограниченных ресурсов. Необходимы согласия заказчиков при принятии основных решений о модификации функций ПС, и только они могут реально определить, как получить комплекс программ с необходимым улучшением функций и высокого качества, выполненный в срок и в пределах выделенного бюджета.

Уровень ***квалификации заказчика*** и корректность техническо-го задания на совершенствование функций и модификации ПС может весьма сильно влиять на выделяемые ресурсы коллектива специалистов для сопровождения комплекса программ. Особенно сильно на достоверность технического задания и возрастание затрат может влиять попытка заказчика форсировать сроки разработки очередных базовых версий программного продукта. 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 сопровождения и необходимых затрат на реализацию допол-нительных требований на изменения. Даже квалифицированные заказчики, вынуждены иногда корректировать техническое задание на любых этапах управления конфигурацией, что может влиять на снижение производительности при сопровождении на 10 – 20%. По тем или иным причинам даже при испытаниях заказчик зачастую обнаруживает, что решаются не совсем те дополнительные задачи и не совсем так, как нужно, вследствие чего необходима переработка готовых модификаций программ, что отражается большим ущербом вследствие дефектов исходных требований заказчика (см. табл. 1).

При проектировании и создании высококачественных комплексов программ, прежде всего, необходима организация и ***тесное взаимодействие представителей заказчика и разработчиков*** проекта. Взгляды и требования заказчика, в основ-

ном, отражаются в функциональных и потребительских характеристиках версий 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. Организация четкого взаимодействия и сокращение этих зон неоднозначностей требует проведения определенных мероприятий и контактов по обмену знаниями, взаимному повышению квалификации и обучению. Представители заказчика, участвующие в оценке и прогнозировании модификаций проекта, должны обучаться формализации автоматизируемых технологических процессов сопровождения, для которых предназначены соответствующие инструментальные средства, и иметь представление об эффективных путях их применения.

При сопровождении комплексов программ большими коллекти-вами велика роль ***квалификации руководителей*** разработки, что непосредственно отражается на успехе и на интегральных дефектах развития проекта ПС. Практически в каждом успешном проекте сопровождения и управления конфигурацией ПС ***должен быть лидер***. Лидером изменений программного продукта может быть: менеджер продукта, менеджер проектирования, руководитель проекта. Лидер должен [5, 19, 24]:

- руководить процессом выявления и формирования требований к модификациям функций программного продукта;
- рассматривать конфликтующие пожелания, поступающие от различных участников сопровождения ПС и находить компромиссы, необходимые для определения набора новых функций, представляющих наибольшую ценность для максимального числа участников проекта, прежде всего, заказчика и пользователей;
- вести переговоры с руководством системы, пользователями и разработчиками и поддерживать равновесие между тем, чего хочет заказчик, и тем, что может предоставить команда сопровождения за ресурсы и время, отведенные для их реализации;

- осуществлять проверку спецификаций изменений программного средства, чтобы удостовериться, что они соответствуют базовой концепции сопровождения, представленной детальным развитием функций программного продукта;
- осуществлять управление изменением приоритетов задач модификаций, а также добавлением и исключением новых функций ПС.

Процесс разработки изменений крупных программных продуктов чрезвычайно трудоемок, а области применения специалистов весьма различные. Поэтому трудно ожидать, что некий способ организации команды специалистов будет во всех случаях предпочтительнее, чем альтернативные варианты. Тем не менее, определенные общие элементы присутствуют во многих успешных командах. Руководство сопровождением крупномасштабного проекта ПС должны осуществлять один или два лидера – менеджера с различными функциями – рис. 3.2:

- ***менеджер сопровождения проекта программного средства*** – этот специалист, обеспечивает коммуникацию между заказчиком и проектной командой, его задача – определить и обеспечить удовлетворение требований заказчика с учетом доступных ресурсов;
- ***менеджер – системный архитектор модификаций программного средства*** – управляет коммуникациями и взаимоотношениями в проектном коллективе, является координатором изменения компонентов, разрабатывает базовые, функциональные спецификации и управляет ими, ведет график управления конфигурацией проекта и отчитывается перед заказчиком за его состояние, инициирует принятие критичных для хода проекта модификаций.

Затраты труда при реализации изменений крупномасштабного ПС полезно распределить по ***двум категориям специалистов***: непосредственно разрабатывающим корректировки компонентов и ПС в целом и обеспечивающим технологию и качество модификаций программного продукта.

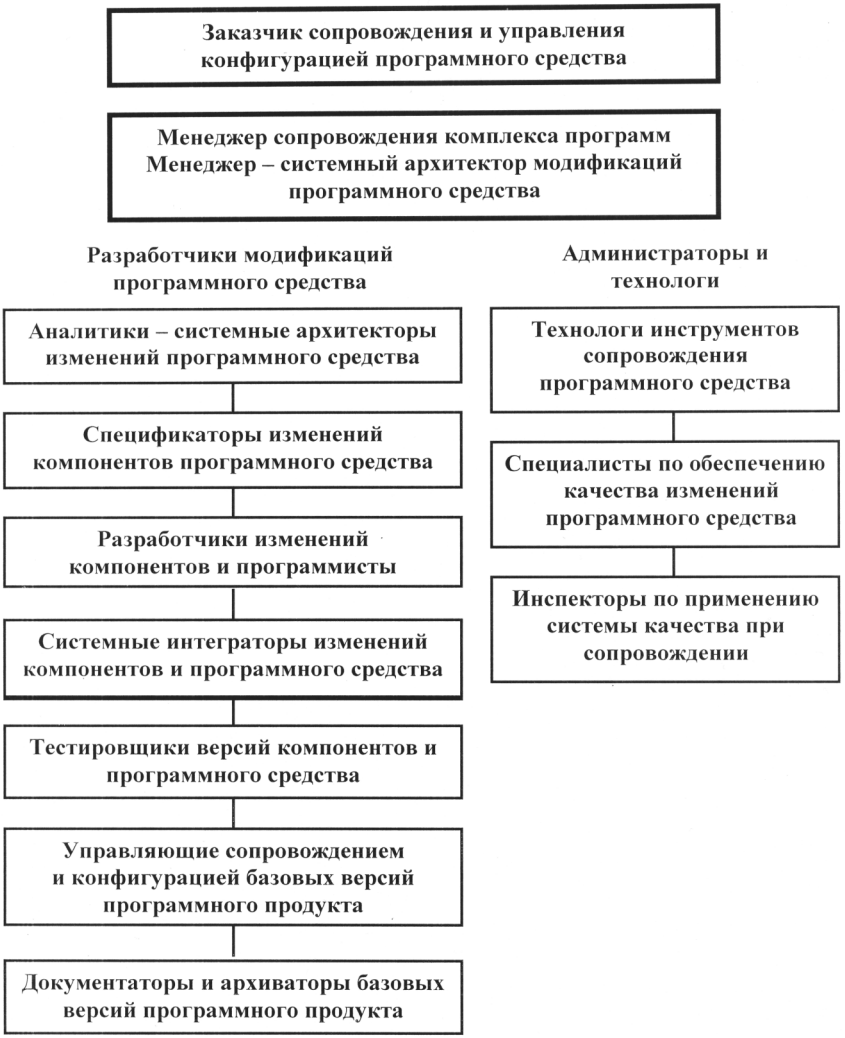

Рис. 3.2

Организационное разделение специалистов, осуществляющих разработку изменений ПС (первая категория), и специалистов, контролирующих и управляющих их качеством в процессе всего ЖЦ (вторая категория), должно обеспечивать эффектив-

ное достижение заданных характеристик, а также независимый, достоверный контроль затрат ресурсов при сопровождении ПС.

Специалисты первой категории непосредственно создают и модифицируют компоненты и ПС в целом с заданными показателями качества. В процессе разработки изменений их функции заключаются в тщательном соблюдении принятой в предприятии технологии сопровождения и управлении конфигурацией, в формировании всех предписанных руководствами исходных, промежуточных и отчетных документов. При этом предполагается, что выбранная технология способна обеспечить необходимые значения конструктивных показателей качества программного продукта, а достижение заданных функциональных характеристик гарантируется тематической квалификацией соответствующих специалистов, регулярным контролем и сокращением возможных дефектов и ошибок в процессе сопровождения (см. табл. 1). Разделение труда специалистов этой категории в крупных проектных коллективах приводит к необходимости их дифференциации по квалификации и областям деятельности, каждая из которых характеризуется определенными типами деятельности и возможных дефектов (см. рис. 3.2).

Трудоемкость творческой части затрат на сопровождение ПС в ***основном определяет человеческий фактор*** — квалификация и организация коллектива специалистов. Следствием этого является большой разброс трудоемкости, производительности труда и дли-тельности выполнения модификаций аналогичных ПС разными коллективами. Непрерывно повышается сложность сопровождаемых ПС, что вызывает возрастание затрат творческого труда на единицу размера программного продукта. В перспективе, не смотря на автоматизацию и повышение инструментальной оснащенности технологии разработки программ, ***доля творческого труда при сопровождении новых версий крупномасштабных ПС возрастает***. Даже при сокращении суммарных затрат на разработку программных компонентов за счет автоматизации нетворческого труда, все более определяющей для технико-экономических показателей модификаций ПС становится доля затрат на творческий труд и возрастают требования к творческим способностям специалистов сопровождения.

Коллектив сопровождения ПС должен иметь в своем составе квалифицированных, ***проблемно-ориентированных анали-***

тиков и системных архитекторов, способных переводить функциональные требования заказчика в конкретные спецификации и технические требования к изменениям комплекса программ и его компонентов [5, 19]. Системные аналитики по сопровождению сложных ПС должны иметь, прежде всего, хорошую подготовку по системному анализу алгоритмов и комплексов программ, по методам оценки эффективности проектов, организации и планированию крупномасштабных разработок модификаций программ и баз данных. Им необходима высокая квалификация по архитектурному построению, комплексной отладке и квалификационным испытаниям версий программных продуктов определенных классов, умение организовать коллектив для решения общей целевой задачи сопровождения ПС. Это позволит на ранних этапах исключать или сокращать дефекты, обусловленные различием представления ими целей и задач корректировки ПС, а также их показателей качества.

Крупномасштабные ПС являются одними из наиболее сложных объектов, создаваемых человеком, и в процессе их совершенствования ***творчество*** как поиск новых методов, альтернативных решений и способов осуществления заданных требований, а также формирование и декомпозиция этих требований составляют значительную часть всех трудозатрат на сопровождение. Индустриализация управления конфигурацией 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 по изменениям ПС. Следствием этого является значительное сокращение доли затрат на творческий труд в непосредственных затратах на сопровождение комплексов программ [1, 21, 64].

Непосредственной реализацией модификаций ПС и компонентов при сопровождении и управлении конфигурацией занимаются следующие специалисты, которые характеризуются рядом возможных дефектов их деятельности (см. рис. 3.2):

- ***спецификаторы*** подготавливают описания изменений функций соответствующих компонентов с уровнем детализации, достаточным для корректной разработки корректировок текстов программ программистами и их интерфейсов, которые могут содержать преимущественно алгоритмические ошибки;

- ***разработчики программных компонентов – программисты*** создают модифицированные компоненты, удовлетворяющие спецификациям, реализуют новые возможности программного продукта, отслеживают и исправляют программные дефекты и ошибки, при разработке сложных систем это требует детального знания методов, технологии и языков программирования, а также проектирования баз данных;
- ***системные интеграторы*** сложных проблемно-ориентиро-ванных ПС работают, в значительной степени, отличными от программистов методами, и имеют на выходе документы модификаций крупных компонентов и сборки версий комплексов программ, с соответствующими системными ошибками интерфейсов и модификаций;
- ***тестировщики*** обеспечивают проверку реализаций изменений функциональных спецификаций, пользовательских интерфейсов, разрабатывают стратегию, выполняют и документируют тестирование для каждого компонента и версии сопровождаемого ПС, должны быть административно независимыми от программистов и спецификаторов, характеризуются соответствующими уровнями оставшихся не выявленными программных, алгоритмических и системных ошибок;
- ***управляющие сопровождением и конфигурацией, инструкторы интерфейсов*** отвечают за снижение затрат на модификацию и сопровождение программного продукта, за обеспечение максимальной эффективности разработчиков изменений по взаимодействию компонентов и реализации версий ПС, принимают участие в обсуждениях совершенствования пользовательского интерфейса и архитектуры программного продукта, что также не может быть без ошибок проектирования, планирования и документирования проекта;
- ***документаторы и архиваторы*** процессов и компонентов сопровождения ПС, обеспечивают подготовку и издание сводных обобщающих технологических и эксплуатационных документов результатов сопровождения и базовых версий программного продукта в соответствие с требованиями стандартов и с возможными дефектами документов.

Анализ и мониторинг характеристик, последствий и частости выявленных дефектов сопровождения конкретного комплекса программ может служить ***ориентиром для оценки индивидуальной квалификации*** и качества работы определенных

специалистов. Следствием такого анализа может быть выделение некоторых специалистов, отличающихся большим числом дефектов, для их замены или дополнительного обучения с целью сокращения числа ошибок соответствующего типа. Накопление, классификация и обобщение характеристик дефектов определенных классов позволяет прогнозировать изменение ошибок по этапам сопровождения ПС, а также необходимое распределения состава и квалификации специалистов.

При выборе заказчиком надежного поставщика-разработчика сопровождения и управления конфигурацией проекта ПС необходима ***оценка тематической и технологической квалификации*** возможного коллектива специалистов, а также его способности реализовать сопровождение с заданными требованиями и качеством. Тематическую квалификацию специалистов в области сопровождения ПС определенного функционального назначения приближенно можно характеризовать средней продолжительностью работы в данной проблемной области основной части команды, непосредственно участвующей в разработке изменений алгоритмов, спецификаций, программ, баз данных и документов. Важнейшую роль играет комплексная квалификация руководителей – менеджеров сопровождения и системных аналитиков функциональных компонентов, и в меньшей степени непосредственных разработчиков изменений 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 сопровождения. При низкой тематической квалификации допускаются наиболее грубые системные ошибки, требующие больших затрат при доработке корректировок программ [20, 56].

Для сокращения и устранения дефектов модификаций посредством адекватных контрмер необходима четкая организация коллектива специалистов и автоматизация процессов исправления дефектов, которые позволяют избегать множества вторичных ошибок, обусловленных недостаточной координацией проводимых корректировок и формирования новых версий сложных ПС. Этому должна способствовать утвержденная дисциплина и иерархия принятия решений на координированные

изменения компонентов и ПС в целом, должностными лицами сопровождения, поддержанная методами и средствами защиты от несанкционированного доступа при выполнении корректировок документов специалистами различной квалификации и права доступа к модификациям компонентов на разных уровнях проекта.

Следует установить ***полномочия специалистов*** или групп для санкционирования и выполнения изменений документов – контрмер, корректировок на каждом уровне разработчиков изменений ПС (см. табл. 2). В контрмеры входит последовательность работ, которые необходимо выполнить для того, чтобы: запросить разрешение на изменение; обработать запрос на изменение; проследить изменение; распределить изменения и сопровождать предыдущие версии программного продукта и документации. Изменения, которые воздействуют на программный продукт, уже находящийся на эксплуатации, должны быть предоставлены заказчику в соответствии с установленными контрактом формами и процедурами.

Особую роль для обеспечения высокого качества при модификации программ и данных играют тестировщики. Организа-ционная структура группы тестирования и распределение ответственности за определенные уровни корректировок, имеют важное значение, для постоянного совершенствование процесса и качества тестирования. Различные виды работ, проводимые по программе тестирования, обычно включаются в состав декомпозиции плана работ всего проекта. Необходимый размер группы тестирования может оцениваться по соотношению числа программистов и тестировщиков или при планировании задач сопровождения. Необходимо определять и фиксировать в плане тестирования распределение основных ролей и ответственности между участниками процесса тестирования [9, 11, 46].

Тестировщики должны обладать специфической квалификацией для разработки стратегии тестирования, а также для проведения работ по тестированию в нужном и достаточном объеме. Они должны уметь обнаруживать дефекты как в исходных текстах программ и данных, так и в новых корректировках при сопровождении. Хороший тестировщик должен обладать аналитическим складом ума, быть внимателен к деталям, организован и, сталкиваясь со сложностями тестирования, проявляет творче-

ский подход и способности к прогнозированию, быть твердым и гибким при обсуждении спорных вопросов с программистами. Ему целесообразно иметь опыт работы с различными платформами, операционными системами, интерфейсами других продуктов и систем, базами данных и языками разработки комплексов программ. Для эффективного проведения тестирования необходимо, чтобы тестировщики обладали достаточным опытом осуществления принятой стратегии тестирования и применения выбранных инструментов.

На основе проведенного анализа и тестирования персонал сопровождения должен разрабатывать варианты реализации изменений и документально оформлять (см. п. 2.3): сообщение о каждом дефекте или заявку на внесение изменений; результаты их анализа и варианты реализации изменений, ***оценку их влияния на функциональную пригодность***. Следует согласовывать с заказчиком выбранные варианты изменений в соответствии с договором. Регистрация и учет истории этого процесса обеспечивает возможность его контроля и пошагового восстановления выполненных изменений (отката) документов для ***выявления вторичных дефектов***, внесенных в процессе разработки очередной версии модификаций. Такие дефекты обычно обусловлены одновременным, не скоординированным внесением групп изменений несколькими специалистами или потерей некоторых корректировок в определенной версии ПС.

Если предполагается, что программный продукт будет иметь длительный, жизненный цикл сопровождения или ожидаются значительные корректировки в ЖЦ ПС, то следует рассмотреть и учесть наиболее детальные требования к методике организации и к коллективу, ответственному за совершенствование и документирование программ. В стратегии сопровождения следует учесть характеристики системы: количество компонентов программного средства, типы, размер, критичность и безопасность создаваемых и применяемых программных продуктов и документов. Для конкретного проекта они должны быть определены и зафиксированы в Программе управления конфигурацией ПС.

В процессах сопровождения сложных ПС могут участвовать большое число специалистов различных направлений и квалификации, которые при необходимости объединяются в

единый коллектив – ***службу сопровождения и управления конфигурацией программного продукта***. Менеджер сопровождения проекта является высшим должностным лицом, принимающим важнейшие решения по внесению изменений и корректировке конфигурации сложных ПС. Он взаимодействует с заказчиком и пользователями, определяющими модификации или эксплуатацию ПС, для согласования изменений в системе, в контракте и в техническом задании. Заказчик системы должен оценивать и утверждать наиболее крупные изменения, заметно влияющие на условия контракта, технические требования или стоимость программного продукта.

Организационная структура коллектива специалистов при корректировках сложных комплексов программ должна учитывать: цели и функции сопровождения; взаимодействующие организации; службы проектирования, закупок и контрактов; систему обеспечения качества и другие средства, которые могут быть привлечены, охватывая, если необходимо, субподрядчиков и поставщиков. Она должна определять связи между различными видами деятельности, непосредственно входящими в процесс сопровождения, обеспечивать координацию действий с другими работами, а также распределение соответствующих полномочий и ответственности за все действия по управлению корректировками. При организации сопровождения сложного проекта следует идентифицировать инстанцию, уполномоченную ***утверждать базовые версии программных продуктов и документов*** для поставки заказчику и пользователям, и любые изменения к ним.

При сопровождении ПС значительное внимание должно быть уделено ***влиянию организации и взаимодействия коллектива разработчиков*** на качество и трудоемкость модификаций сложных программных средств. В составе организационных характеристик коллектива следует 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 сопровождения ПС. При этом большое значение может иметь личная мотивация и психологические особенности поведения разных специалистов при комплексной работе над управлением конфигурацией проекта. Эти характе-ристики

могут быть обобщены в качественный показатель влияния сложности взаимодействия специалистов в коллективе, которому сопоставлены коэффициенты изме-нения трудоемкости разработки ПС [4, 26, 41]. Наилучшим считается непрерывное корректное взаимодействие 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 изменений ПС, что нельзя не учитывать при прогнозировании и обосновании разработки модифика-ций.

Специалисты второй категории – администраторы и технологи, обслуживающие и сопровождающие технологический инструментарий, который применяется специалистами первой категории, обеспечивают применение системы качества проекта или предприятия, контролируют и инспектируют ее использование. Основные задачи второй категории специалистов 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 сопровождение ПС.

Технологи должны выбирать, приобретать и осваивать наиболее эффективный инструментарий для проектов, реализуемых конкретным предприятием с учетом особенностей сопровождаемых 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 сопровождения.

Специалисты, управляющие обеспечением качества изменений ПС, должны овладеть стандартами и методиками предприятия, поддерживающими регистрацию, контроль, документирование и воздействия на показатели качества на этапах сопровождения комплексов программ. Они должны обеспечивать эксплуатацию системы качества проекта, выявление всех отклонений от заданных показателей качества объектов и процессов, а также от предписанной технологии на этапах сопрово-

ждения и управления конфигурацией. Эти же специалисты должны анализировать возможные последствия выявленных отклонений от требований технического задания или спецификации на изменения 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 применения систем качества предприятия и качества модификаций 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 сопровождения:

- получать и справедливо оценивать объективные данные испытаний характеристик модификаций компонентов, программных 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 модификаций и заказчика сопровождения выводам, основанным на реальных наблюдениях в процессе испытаний характеристик качества изменений ПС.

Для реализации мероприятий по планированию и управлению жизненным циклом и сопровождением концептуально целостных, крупномасштабных ПС и обеспечения их качества в пределах допустимых затрат, необходимы организационные действия менеджеров и системных архитекторов, направленные на подбор и обучение коллектива специалистов разных категорий и специализаций. Перечисленные выше специализации и

квалификации персонала (см. рис. 3.2), участвующего в крупномасштабных проектах ПС, ***требуют соответствующего их отбора, подготовки и обучения***. Должны быть разработаны и документированы планы сопровождения проекта, требования и цели обучения специалистов, а также разработаны руководства, включая документы, используемые для обучения (см. шестую часть стандарта **ISO 15504**, [33]). Персонал, ответственный за выполнение конкретных задач модификации программных продуктов с высоким риском [28, 53, 54] должен быть ***аттестован***, если это необходимо, на основе соответствующего образования, подготовки и/или опыта работы. Может также стать необходимым включение в подготовку специалистов, ознакомление со специфической (проблемно-ориентированной) областью, в которой будет работать данный программный продукт, и повышение квалификации в этой области. Требования к квалификации, обучению персонала и их реализации должны быть документально оформлены в контракте на сопровождение.

Совещания предоставляют заинтересованным специалистам из различных команд и предприятий возможность работать вместе над достижением общей цели сопровождения крупного проекта [37, 64]. Надлежащая их подготовка является залогом успеха совещаний. Первым делом необходимо распространить идею внутри предприятия, разъясняя преимущества проведения совещаний членам команды при сопровождении. Подготовка включает в себя также выявление заинтересованных лиц, которые могут повлиять на процессы ЖЦ ПС и чьи потребности необходимо учесть, чтобы гарантировать успешные результаты модификаций программного продукта.

Необходимо заранее разослать 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 конфигурацией и сопровождения. 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 спо-

собствовать 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 модификации их проекта, чтобы обеспечить как качество, так и своевременность разработки версий программного продукта. Именно заказчики несут финансовую ответственность за выполнение внешних обязательств перед пользователями. Необходимы указания и согласия заказчиков при принятии основных решений, и только они могут реально определить, как, развивая функции ПС, получить полезный комплекс программ высокого качества, выполненный в срок и в пределах бюджета. Привлечение заказчика помогает наименее болезненно решать проблемы управления масштабом и функциями модификаций проекта ПС.

Для защиты, как проекта, так и целей заказчика может понадобиться вести переговоры об объеме работ для команды сопровождения. Во время переговоров с заказчиком о техническом задании и требованиях базового уровня следует руководствоваться принципом – меньше обещать и больше делать. Тогда неизбежные издержки разработки изменений ПС (непредвиденные технологические риски, изменения требований, задержки при приобретении закупаемых компонентов, непредвиденный уход основных членов команды) не приведут к нарушению графика реализации корректировок ПС. Однако заказчики обычно желают получить как можно больше новых функциональных возможностей в каждой версии программного продук-

та. Именно эти функциональные возможности создают добавленную стоимость, которая важна для достижения их бизнес-целей. Следует с пониманием относиться к требованиям заказчиков, поскольку именно они, в конечном счете, добиваются успеха модифицированного программного продукта на рынке.

3.3. Характеристики качества процессов сопровождения программных средств

Обычно заказчики и разработчики первоначально устанавливают требования к каждой модификации характеристики 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 несбалансированным значениям требований к отдельным, взаимосвязанным изменениям характеристик программного продукта, на которые не рационально используются ограниченные ресурсы сопровождения и ЖЦ ПС, или к не адекватно низким их значениям качества. В проектах крупномасштабных ПС это может угрожать значительным повышением стоимости и/или снижением конкурентоспособности нового или модифицированного программного продукта из-за недостаточного уровня отдельных изменений характеристик качества [1, 24, 59].

Характеристики качества программного средства является важным аспектом сопровождения программного продукта. Сопроводители должны иметь Программу обеспечения качества и совершенствования программного средства, охватывающую характеристики качества, установленные в стандарте **ISO 9126**. При сопровождении программного средства должен быть реализован соответствующий процесс для определения, описания, выбора, применения, совершенствования и оценки характеристик модифицированного ПС. В части процесса, связанного с оценкой характеристик программного средства, сопроводитель должен предпринять определенные действия при корректирующем, профилактическом, адаптивном и полном сопровождении в пределах доступных ресурсов. Система показателей (метрик) качества должна содействовать реализации процесса сопровождения и способствовать усовершенствованию данного ПС. Должны быть собраны,

проанализированы и интерпретированы соответствующие данные, способствующие усовершенствованию процесса сопровождения и пониманию методов снижения его стоимости. Целесообразно собирать соответствующие эмпирические данные об оценках, способствующие снижению затрат на сопровождение и весь жизненный цикл ПС. Подробности процесса сопровождения программного продукта должны быть документально оформлены, чтобы персонал сопровождения действовал в рамках эффективного процесса.

Стандарт **ISO 9126** – может использоваться как основа для формального ***регламентирования характеристик качества при сопровождении и модификации ПС***. Модель характеристик качества ПС и компонентов состоит из шести групп базовых показателей, каждая из которых детализирована несколькими нормативными субхарактеристиками :

Функциональная пригодность детализируется:

- 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;
- стабильностью функционирования после корректировок;
- 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. Кроме того, отсутствуют методики измерения характеристик и сопоставления с требованиями спецификаций, а также рекомендации, на каких этапах ЖЦ ПС их целесообразно применять. Описания показателей качества ориентированы на высококвалифицированных системных аналитиков и заказчиков ПС, которым предоставляется возможность выбирать необходимую номенклатуру и способ оценивания характеристик в соответствии с назначением, областью применения и конкретными особенностями создаваемых и сопровождаемых ПС.

Функциональная пригодность – наиболее ответственная, объективно трудно формализуемая и оцениваемая в проекте и при его сопровождении 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.

В процессе сопровождения комплекса программ атрибуты функциональной пригодности должны конкретизироваться в спецификациях на модификации 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 расширении и совершенствовании ПС, число реализуемых новых требований заказчиков. Кроме них, функциональную пригодность отражают множество различных специализированных критериев, которые тесно связаны с конкретными решаемыми задачами и сферой применения определенного комплекса программ. Их можно рассматривать как частные критерии или как факторы, влияющие на основной показатель качества сопровождения ПС. Ниже предполагается, что рассматриваемые характеристики качества сопровождения всегда направлены, в первую очередь (может быть неявно), на ***обеспечение и совершенствование функциональной пригодности программного продукта***.

Требования к процессам оценивания качества сопровождения можно структурировать на главные (функциональные), организационные, проектные, а также выделять внутренние и внешние метрики качества и их измерение, ориентируясь на характеристики и их атрибуты в стандарте **ISO 9126**. Реализация процессов оценивания при модификации ПС по требованиям стандарта должна быть автономной и независимой от специалистов и процессов создания ПС и его компонентов, однако, коррелированной с этапами жизненного цикла и сопровождения конкретного проекта в соответствии с применяемой адаптированной версией стандарта **ISO 12207**. Для обеспечения высокого качества модификаций при сопровождении комплексов программ необходимо решать ***две задачи***:

- обеспечить соответствие полной совокупности требований к корректировкам программного средства, его компонентов и модулей исходным требованиям к качеству новой версии программного продукта или системы;
- создать модифицированные программные модули, функциональные компоненты и программный продукт, соответствующие требованиями, разработанным при решении первой задачи.

Первая задача решается методами верификации и последовательного прослеживания соответствия требований на изменения ПС при их ***детализации*** от требований к функциональным характеристикам модифицированного программного продукта или системы до требований в спецификациях к измененяемым программным компонентам и модулям ПС. Для решения второй задачи применяются методы тестирования и последовательной ***интеграции*** модулей, программных компонентов и комплекса программ до полного соответствия новой версии программного продукта исходным требованиям к качеству системы и внешней среды. Таким образом, образуются два взаимодействующих процесса: верификации и декомпозиции требований к модификации и тестирования для проверки соответствия ПС этим требованиям (см. п. 3.4).

В стандарте **ISO 9126** установлены ***понятия сопровождаемость и мобильность – переносимость***, которые целесообразно использовать для непосредственного учета и оценки при разработке модификаций ПС. Группы конструктивных характеристик качества ПС – Сопровождаемость и Мобильность трудно измерять количественно, и они доступны в основном ***качественным оценкам их свойств***. В ряде проектов для субхарактеристик Сопровождаемости и Мобильности при проектировании могут доминировать технико-экономические меры ***трудоемкости*** (человеко-часы) и ***длительности*** (часы) для процедур, обеспечивающих реализацию атрибутов этих субхарактеристик. Однако для некоторых атрибутов в этой группе характеристик приходится применять порядковые меры экспертных бальных шкал с небольшим числом градаций. Они могут служить ориентирами при выборе и установлении требуемых значений этих показателей качества в спецификациях на изменения ПС.

Первым компонентом для анализа и требований к качеству сопровождаемости СУБД является комплекс программ. 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. ***Вторым компонентом БД*** является собственно накапливаемая и обрабатываемая информация. В сопровождении систем баз данных доминирующее значение приобретают сами данные, их хранение и обработка. Для оценивания качества сопровождаемости информации БД может сохраняться общий,

методический подход к выделению адекватной номенклатуры стандартизированных в **ISO 9126** базовых характеристик и субхарактеристик качества ПС. Однако их содержание для применения к качеству ИБД при проектировании требуется уточнять. Выделяемые показатели качества должны иметь практический интерес для пользователей БД и быть упорядочены в соответствии с приоритетами практического применения. Кроме того, каждый выделяемый и модифицируемый показатель качества ИБД должен быть пригоден для достаточно достоверного оценивания или измерения, а также для сравнения с требуемым значением при испытаниях.

Сопровождаемость: приспособленность ПС к модификации и изменению конфигурации. Модификации могут включать исправления, усовершенствования или адаптацию ПС к изменениям во внешней среде применения, а также в требованиях и функциональных спецификациях заказчика [8, 26, 51]. 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. **ISO 14764** и **ISO 15846**).

Требования к сопровождаемости количественно можно оценивать для субхарактеристик изменяемости и тестируемости экономическими категориями допустимой трудоемкости и длительности реализации этих задач при некоторых средних условиях. 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 поддержки ЖЦ данного ПС. 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. 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 требуется 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 ориентирована архитектура, внешние и внутренние интерфейсы программ. Если ПС должен модифицировать конечный пользователь, изменяемость может быть предпосылкой и частью простоты использования.

Стабильность: способность ПС предотвращать и минимизировать непредвиденные негативные эффекты от его изменений, возможность локализовать и ограничивать область влияния изменений программ и данных. Эта внутренняя субхарактеристика определяется архитектурой ПС, унифицированностью его интерфейсов, корректностью технологической документации и может существенно влиять на функциональную пригодность, надежность и адекватность применения комплекса программ при изменениях.

Тестируемость: свойство ПС, обеспечивающее простоту проверки качества изменений и приемки модифицированных компонентов программ (см. **ISO 12119**). 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 межмо-

дульных и внешних интерфейсов, от полноты и корректности технологической документации. Возможность локализации изменений и унификация интерфейсов компонентов с некорректируемой частью ПС, позволяет снижать сложность, трудоемкость и длительность их тестирования, упрощает подготовку тестов и анализ результатов. В этой субхарактеристике учитываются, в основном, техническая и организационная составляющие процесса тестирования модификаций и не входит функциональная часть их подготовки.

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 базовой версии сложного комплекса программ. Эти экономические шкалы по существу, хотя и не явно, могут отражать также атрибуты анализируемость и стабильность, и применяться для интегрального оценивания сопровождаемости в целом. Они влияют на динамическое развитие и совершенствование функциональной пригодности версий ПС и могут не учитываться при стабильной эксплуатации конкретной версии.

Сопровождаемость информации БД в программном продукте 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 изменений 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 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 ориентирована 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. Её атрибуты зависят от четкости формализации в 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: подготовленность программного продукта к переносу из одной аппаратно-операционной среды в другую [32, 57]. 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 круп-

ных 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 размером, трудоемкостью и длительностью необходимых доработок компонентов и операций по адаптации, которые следует выполнить для обеспечения полноценного функционирования ПС после переноса на иную платформу. 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. 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 использования. Также 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. Замещаемость не предполагает, что заменяемый компонент ПС способен полностью выполнять функции предшествующего 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.

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.

Мобильность информации 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. Особенности и трудоемкость переноса ИБД зависят, прежде

всего, от характеристик совместимости архитектуры и содержания информации переносимой БД между рассматриваемыми платформами (см. п. 1.6). 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. Эти задачи значительно упрощаются при одновременном сокращении затрат при применении идеологии и концепции открытых компьютерных систем, поддержанных комплексом международных стандартов POSIX, а также современных версий ОС и СУБД, как стандартов де-факто (см. главу 4) [22, 36, 60, 62].

Стандарт **ISO 14598** целесообразно применять для ***оценивания качества программных средств на различных этапах жизненного цикла***: лабораториям по сопровождению и тестированию комплексов программ и их компонентов, поставщикам, потребителям, пользователям и сертификационным организациям. В стандарте изложена общая схема процессов оценивания характеристик качества при разработке и модификации программ, которую составляют:

- формализация исходных требований для оценки значений модификации характеристик качества программного средства;
- формализация принципов и особенностей оценивания при проведении экспертиз и изменений характеристик качества программного средства;
- планирование и проектирование процессов оценивания характеристик качества в жизненном цикле и при сопровождении программного средства в соответствии с потребностями пользователей этих характеристик;
- реализация процессов измерений и оценивания качества версий программного средства, оформление и использование результатов.

Атрибуты характеристик качества при модификациях ПС имеют различные меры и шкалы, вследствие чего они в боль-

шинстве могут быть непосредственно ***не сопоставимы между собой***. Они предварительно выбираются и согласовываются с заказчиком при сопровождении и последовательном, почти независимом анализе каждого атрибута качества, для использования в контракте и техническом задании. Для обобщенного оценивания качества ПС необходим учет относительного влияния изменений каждой конструктивной характеристики, на функциональную пригодность. При этом первоначально не всегда учитываются ресурсы для их реализации в конкретном ПС. Это часто приводит к выдвижению ряда не рациональных требований к модификациям, которые значительно отличаются: либо по степени влияния на функциональную пригодность, либо по величине ресурсов, необходимых для их реализации.

Для целенаправленного эффективного управления качеством сопровождения сложного ПС целесообразно иметь механизм объединения разнородных характеристик в некоторый интегральный показатель, отражающий их совокупное влияние на его функциональную пригодность. Наиболее полно ***качество сопровождения системы и ПС характеризуется величиной допустимого ущерба – риска***, возможного при проявлении дестабилизирующих факторов и реализации конкретных угроз безопасности применения пользователями модифицированной версии программного продукта, а также средним временем между возможными проявлениями угроз, ***нарушающих качество и безопасность***. С этой позиции затраты ресурсов разработчиками и заказчиками на контрмеры и обеспечение качества модификаций должны быть соизмеримыми с возможным ущербом у пользователей от его нарушения. Проявления рисков могут быть следствием различных, ***негативных – рисковых явлений*** в системе и ПС, которые в основном оцениваются [28, 48, 54]:

- величиной экономического, материального или социального ущерба системе вследствие возникновения отказовой ситуации после корректировок программного продукта;
- трудоемкостью устранения последствий отказовой ситуации и восстановления нормальной работоспособности системы и ПС, в соответствии с требованиями после отказа;
- длительностью неработоспособного состояния после изменения системы или ПС вследствие каждого отказа до полного восстановления их работоспособности;

- затратами ресурсов и времени, необходимыми после отказа для восстановления нормальной работоспособности системы и программного продукта в соответствии с требованиями.

Основное содержание, размер и требуемое качество модифика-ций ПС практически всегда определяют затраты и риски, связанные с их непосредственной разработкой. Влияние этой части затрат определяется наиболее сложным творческим процессом изменений программ, который зависит от многих факторов. Накопленный опыт сопровождения ПС и обобщение проведенных исследований позво-лили в [28, 41, 53, 54] выделить четыре основные ***группы факторов***, влияющих на ***оценки рисков*** сопровождения комплексов программ:

- факторы, отражающие особенности сопровождаемого комп-лекса программ ***как объекта***, и требования к функциональным характеристикам и качеству модификаций;
- факторы, определяющие ***организацию процесса сопровож-дения*** комплексов программ и его обеспечение квалифицирован-ными специалистами;
- факторы, характеризующие ***технологическую среду*** и осна-щенность инструментальными средствами автоматизации сопровож-дения и управления конфигурацией комплекса программ;
- факторы, отражающие оснащенность процесса сопровожде-ния ПС ***аппаратурными вычислительными средствами***, на кото-рых реализуются комплексы программ и базируются инструмен-тальные системы автоматизации.

В представленных четырех группах распределены факторы, которые наиболее важны при анализе рисков сопровождения ПС. В то же время имеющийся опыт показывает, что обычно отсутствуют отдельные, предсказуемые факторы или методы, способные изме-нять на порядок или более основные риски процесса сопровож-дения. Риски при модификациях ПС могут быть обусловлены ***недостатками или непредумышленными, негативными действиями различных лиц***, участвующих в изменениях или применении версий системы и программного продукта. (Злоумышленные, враждебные действия заинтересованных лиц и защиты от них, при анализе и сокращении рисков сопровождения ПС ниже не рассматриваются). Основными источниками непредумышленных рисков ПС, которые могут приводить к ущербу при их изменении и применении, являются:

- ***заказчики,*** определяющие назначение и функции изменений системы и программного продукта, которые могут задавать некорректные или нереализуемые разработчиками требования к ним, а также ограничивают выделенные и доступные ресурсы для корректировок ПС: бюджет, время, затраты на технологию и инструментальные средства;
- ***разработчики*** модификаций системы и ПС, обеспечивающие их реализацию, которые могут допускать дефекты и ошибки при обосновании изменений, не выполнять согласованные с заказчиком требования к характеристикам и качеству версии комплекса программ, а также превышать допустимое использование выделенных ресурсов;
- ***менеджеры и эксперты управления рисками*** – координаторы взаимодействия заказчиков и разработчиков модификаций, которые уполномочены, принимать решения о необходимости их снижения, путем применения необходимых контрмер, а также о допустимости применения системы и/или ПС с прогнозируемыми или достигнутыми, конкретными уровнями рисков.

Косвенными источниками и причинами рисков функционирования версий программного продукта могут быть также ***пользователи***, некомпетентно применяющие систему или программный продукт с отклонениями от требований документации по функциональной пригодности или с недопустимым использованием ресурсов при их эксплуатации.

Управление рисками, связанными с результатами сопровождения и применения ПС, должно представлять собой формализованный процесс, позволяющий ***систематически идентифицировать, оценивать и смягчать факторы, угрозы и величину последствий возможных рисков***. В процессе управления сопровождением и конфигурацией проекта основное внимание должно уделяться идентификации угроз и рисков, имеющих как внешние, так и внутренние причины, их количественную оценку, разработку откликов и контрмер для сокращения рисков и контроль реализации откликов. Риски при сопровождении и применении программного продукта могут негативно влиять на проект (в том числе на качество) всей системы – рис. 3.3 [28]:

- ***функциональной пригодности,*** назначения, состава и характеристик основных, функциональных задач, решаемых системой и программным продуктом;
- ***конструктивных характеристик*** конечного программного продукта и результатов его применения для целей системы;
- ***нарушения ограничений доступных и используемых ресурсов***, к превышению допустимых затрат, нарушению сроков или же к полному срыву возможности реализации изменений ПС и системы в заданных ограниченных условиях.

Эти три базовые группы характеристик изменений систем и ПС тесно связаны между собой и определяют соответствующие классы рисков. Изменение каждого из них может влиять на другие риски.

Рис. 3.3

Риски функциональной пригодности имеют доминирующее значение и изменения двух других классов рисков обычно должны быть, в первую очередь, подчинены сокращению рисков функционирования системы и комплекса программ. Поэтому анализ рисков и возможных угроз целесообразно проводить систематизировано, начиная с рисков функциональной пригодности программного продукта.

Ущерб, вследствие ошибок функциональных требований к ПС может проявляться двумя видами рисков: недостатками достигнутых ***характеристик*** ПС, и рисков от нарушения ограниченности доступных и используемых ***ресурсов*** в последующем сопровождении.

В жизненном цикле ПС важнейшим риском является ущерб при невыполнении комплексом программ, формализованных в требованиях заказчика, назначения и функций главной характеристики качества – ***функциональной пригодности***. Характеристики, определяющие функциональную пригодность сложных ПС, требуют для модификации каждой из них различных видов ресурсов и величин затрат. Все виды рисков процессов и продуктов сопровождения ПС должны учитывать, прежде всего, их степень влияния на этот основной вид риска. Предъявление заказчиком необоснованных требований к изменению функциональной пригодности, проявления в них конфликтов и внутренних противоречий в содержании модификаций функций и компонентов, при реально доступных ресурсах и возможных условиях внешней среды применения ПС, могут вызывать наиболее существенный ущерб. В зависимости от назначения, функций, критичности требований к системе и ПС, может потребоваться либо полная ликвидация определенных или всех видов рисков, либо сокращение некоторых из них до допустимых пределов, либо игнорирование некоторых и сохранение на достигнутом уровне ущерба вследствие нарушения требований заказчика.

Неопределенность оценок масштаба изменений комплексов и компонентов программ является одним из важнейших факторов, ***влияющим на риски сопровождения ПС***. Точность оценки изменения размера новой версии программного продукта, значительно повышается после четкого формулирования требо-

ваний спецификаций заказчика, проведения анализа требований. Ошибки оценивания масштаба модификаций ПС могут быть больше или меньше реальных значений, что по-разному отражается на ***характеристиках договора с заказчиком*** на сопровождение, а также на факторах и типах рисков. Если масштаб модификации ПС в договоре ***завышен***, то следствием могут быть, прежде всего, ***риски – ущерб экономических категорий*** – превышения требуемых ресурсов: стоимости, трудоемкости, длительности выполнения изменений и числа необходимых специалистов. При этом заказчик терпит убытки, разработчик получает неоправданную прибыль. При реализации модификаций 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 изменений версии программного продукта 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 сопровождения проекта. Недооценка размера модификаций комплекса программ и ресурсов для их реализации, может резко увеличивать число дефектов в версии программного продукта. Кроме того, ***ограниченные ресурсы*** для реализации изменений в полном объеме функциональной пригодности, могут отражаться на ее качестве и удобстве применения, а также на конструктивных характеристиках: на безопасности, надежности, качестве взаимодействия с внешней средой и с пользователями, качестве документации и других эксплуатационных факторах.

Оценки масштаба изменений ПС и рисков должны быть проанализированы и скорректированы для установления в договоре между заказчиками и разработчиками исходного компромиссного масштаба, допустимого для разработки спецификаций требований к модификациям с требуемым качеством. Некоторые требования могут потребовать изменения (обычно увеличения) масштаба, и соответственно ресурсов на этапах предварительного и детального проектирования изменений для обеспе-

чения возможности реализации требуемой функциональной пригодности с допустимыми рисками. ***Требования*** к функциональной пригодности и конструктивным характеристикам ПС ***должны быть согласованы с масштабом модификаций проекта*** и всеми видами ресурсов для их реализации. Для этого необходимо использовать адекватные методы и единицы измерения масштаба изменений ПС [5, 19, 41]. Таким образом, при разработке требований к изменениям характеристик проекта выявилась проблема анализа системной эффективности ПС и обобщения его характеристик, а также оценивания совместного влияния модификаций конструктивных характеристик на функциональную пригодность ПС с учетом затрат на их реализацию.

Для управления рисками при сопровождении и детального сопоставительного оценивания изменяемых характеристик качества ПС целесообразно каждой 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 на изменения, 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 модификации к этому сокращению риска***.

Значения приоритетов – рисков предназначены для указания относительной степени важности соответствующих изменений свойств или атрибутов характеристик качества для функциональной пригодности проекта, или допустимых ресурсов при их реализации. Для конкретного проекта ПС состав и значения приоритетов модификаций следует ***поэтапно адаптировать и уточнять*** с учетом их назначения и функций. Наивысший приоритет (10) следует интерпретировать как обязательное выпол-

нение разработчиком модификаций соответствующего требования к указанному изменению свойства или атрибута качества с отсутствием последующего риска. Низшее значение приоритета (1) означает, что данный малый риск может не учитываться при данном изменении ПС. Промежуточные значения приоритетов должны отражать относительное влияние модификаций соответствующих атрибутов на функциональную пригодность и ее свойства с учетом доступных ресурсов на их реализацию. При этом, возможно, что некоторые требования к изменениям атрибутов качества при их низких приоритетах могут не полностью реализоваться в реальном комплексе программ. Это дает возможность разработчикам творчески подходить к требованиям заказчика при сопровождении и реализации модификаций ПС в условиях ограниченных ресурсов.

Для проведения последовательного оценивания обобщенных приоритетов при управлении рисками и корректировке атрибутов качества ***конкретного проекта ПС***, целесообразно проводить:

- экспертную оценку коэффициента влияния изменения, требуемой конструктивной характеристики качества на функциональную пригодность (в диапазоне 1 – 10);
- экспертную оценку относительных затрат ресурсов на реализацию требуемых значений качества без учета рисков (1 – 10);
- оценку относительного коэффициента влияния изменений конструктивной характеристики, на функциональную пригодность с учетом затрат на ее реализацию – обобщенный уровень приоритета, требуемого значения атрибута качества с учетом рисков (0,1 – 10).

При сопровождении полезно последовательно уточнять риски требований к каждому изменению свойства и атрибута качества ПС с использованием первых двух видов коэффициентов. Для крупномасштабных проектов комплексов программ при уточнении требований к качеству при модификациях целесообразно использовать ***обобщенный уровень приоритета – риска***. При этом набор значений обобщенных уровней приоритетов изменений для выбранных атрибутов качества конкретного проекта ПС, можно разделить на ***три группы***:

- доминирующие модификации характеристик и риски, оказывающие наибольшее влияние на функциональную пригод-

ность при допустимых затратах (обобщенный приоритет > 8);

- показатели изменений конструктивных характеристик и рисков, имеющие достаточное влияние на функциональную пригодность и значительные затраты на реализацию (обобщенный приоритет < 7, но > 4);
- характеристики изменений 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 изменений 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 модификаций 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 изменений 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 сопровождении и управлении конфигурацией крупномасштабных ПС.

Следует подчеркнуть, что выше анализировалось и оценивалось преимущественно ***изменение функциональной пригодности и сни-жение 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 сопровождения и применения ПС, а также потенциальный уровень спроса на рынке. Если затраты на разработку модификаций ПС можно оценивать и прогнозировать с определенной достоверностью, то эффективность применения и особенно будущий спрос на конкретную версию программного продукта со стороны пользователей ап-

риори оценить трудно. Такие оценки могут проводиться на основе специальных маркетинговых исследований 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.

3.4. Верификация и тестирование модификаций при сопровождении программных средств

Затраты на верификацию и тестирование модификаций программных средств могут достигать половины общих трудовых затрат при сопровождении сложных комплексов программ. Поэтому необходима систематизация и оптимизация этих процессов для повышения качества и снижения трудоемкости сопровождения. Подобные задачи должны решаться при разработке процессов тестировании модификаций, расширений и крупных корректировок ПС. Особенностью тестирования в этом случае является локализация изменений компонентов и части ПС для сосредоточения на них усилий тестировщиков. Однако существующие функциональные связи и интерфейсы корректировок с предыдущими версиями комплекса программ могут вызывать необходимость расширять область тестирования, вплоть до охвата всей новой версии программного продукта. Поэтому ниже рассматриваемое тестирование при сопровождении не разделяет процессы выявления дефектов и ошибок в первичных базовых версиях ПС, и в корректировках при создании новых функций и версий.

Исходным эталоном при разработке или сопровождении любой программы являются требования спецификации заказчика и/или потенциального пользователя, предъявляемые к создаваемому или модифицируемому комплексу программ. Эти документы должны устанавливать состав, содержание и значения результатов, которые должен получать пользователь или система при определенных условиях и исходных данных. Любое отклонение результатов функционирования программы от предъявляемых к ней требований и сформированных по ним эталонов следует квалифицировать как дефект или ошибку в программе [5, 9, 19]. Проявление дефекта далее определяется как риск (ве-

роятность), умноженный на стоимость устранения неисправности.

Анализ и формирование требований к тестам проводится для установления ***целей, задач и стратегий тестирования***, а также инструментов тестирования, которые будут применяться при сопровождении. При сопровождении сложных комплексов программ важно иметь пригодные для подготовки тестирования ***системные требования к модификациям или сценарии использования новых функций*** системы. Эти требования необходимо анализировать и определять в терминах требований к тестам, устанавливающих их содержание и корректность. При анализе требований к тестам решаются следующие проблемы: что нужно сделать при определении требований к тестам для модификаций версии ПС; как преобразовать его архитектуру, изменения системных требований или сценарий использования, в требования к тестам; как провести анализ и изменение документации на программы и данные для формулировки требований к новым тестам. Требования к тестам должны содержать подробный перечень того, что должно быть протестировано. При разработке требований к тестам специалисты по тестированию должны выполнить несколько шагов, чтобы получить представление о потребностях заказчика при эффективном сопровождении. Необходимо также изучить новые системные требования к ПС и БД, изменения сценариев использования системы и описание модификаций назначения системы для того, чтобы лучше понять цель её сопровождения. Кроме того, следует определить дополнения новых функций, наиболее значимых для применения системы, и функций повышенного риска.

Стратегию создания требований к тестам можно строить, ***основанной на требованиях*** или на ***поведении системы***. Требования к тестам можно также получить на основе представлений о логике архитектуры системы. Такой подход называется ***структурным***, он может изменяться в зависимости от условий договора или особых требований к безопасности системы. Например, покрытие полной структуры решений может быть необходимо в аэрокосмических и других системах с повышенными требованиями безопасности.

Разрабатывая требования к тестам на основе системных требований или сценариев использования системы, группа тес-

тирования предполагает создать, по крайней мере, одно требование к тестам на каждое системное требование. Соотношение между системными требованиями и требованиями к тестам системного уровня может варьироваться, причем каждому системному требованию может соответствовать одно или несколько требований к тестам, в зависимости от степени риска и детализации требования. Системные требования или сценарии использования системы, разбитые на составные части на уровне спецификации программного продукта или архитектуры, обычно проверяются во время модульного или комплексного тестирования.

Первичные и вторичные ошибки комплексов программ выявляются на ***последовательных этапах верификации и тести-ро-вания***. Для обеспечения высокого качества модификаций при сопровождении ПС необходимо решать ***две задачи***:

- обеспечить соответствие исходным требованиям на модифицируемую систему, всей совокупности разработанных изменений требований к характеристикам программного средства, к его компонентам и модулям;
- создать модифицированные программные модули, функциональные компоненты и программный продукт, соответствующие требованиями, разработанным при решении первой задачи сопровождения.

Первая задача решается методами верификации и последовательного прослеживания ***сверху вниз*** взаимного соответствия требований на изменения, при их ***детализации***, начиная от требований к модифицированным функциям системы, до требований в спецификациях к изменениям программных компонентов и модулей ПС. Для решения второй задачи применяются методы тестирования и последовательной ***интеграции*** откорректированных модулей, программных компонентов и комплекса программ ***снизу вверх*** до полного соответствия модифицированного программного продукта исходным требованиям к системе (рис. 3.4). Таким образом, при создании ПС должны реализоваться два взаимодействующих процесса: ***верификация***, декомпозиция и детализация требований и ***тестирование*** для обеспечения соответствия компонент и комплекса программ этим требованиям.

Основная цель ***верификации ПС и компонентов*** состоит в том, чтобы обнаружить, зарегистрировать и устранить дефекты

и ошибки, которые внесены во время последовательной разработки или модификации требований к программным компонентам разных уровней [27]. Для обеспечения эффективности затрат ресурсов при её реализации, верификация должна быть интегрирована как можно раньше с основными процессами проектирования и сопровождения в жизненном цикле ПС. Она проводится, начиная от общих требований, заданных в техническом задании и/или в спецификации на всю систему, до детальных требований на изменения программных компонентов, данных, модулей и их взаимодействия. Цели верификации ПС и информации БД достигаются посредством последовательного выполнения комбинации из просмотров, анализов, разработки контрольных сценариев и процедур, и последующего их выполнения.

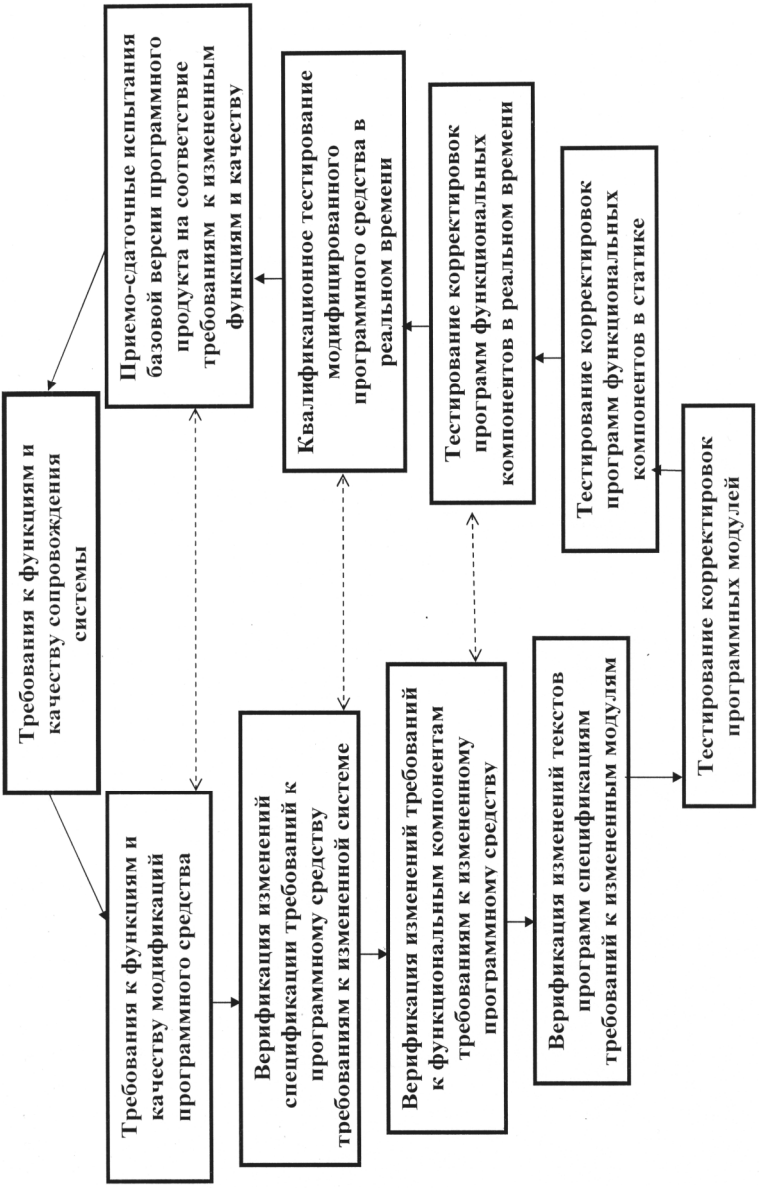

Рис. 3.4.

Такие сценарии предназначены для поэтапной проверки внутренней непротиворечивости и полноты реализации преобразований требований. Требования должны проверяться на корректность системных взаимосвязей компонентов ***сверху вниз*** и

на внутреннюю корректность взаимодействия между требованиями внутри компонента и их пригодности для последующего тестирования и применения.

Последовательная разработка и верификация корректировок спецификаций требований к программным компонентам должна обеспечивать корректность их логических и функциональных связей. Однако этот процесс не гарантирует полноту и корректность всех требований, так как обычно эти работы проводятся частично вручную и могут отсутствовать четкие эталоны для их проверки.

Тестирование при сопровождении используется для анализа процесса и результатов исполнения в основном откорректированных программ. Однако тесты могут также применяться автономно для непосредственной проверки полноты и корректности модификаций компонентов, и соответствия исходным спецификациям требований на изменения комплекса программ, его компонентов и модулей. Тем самым совокупность спецификаций тестов может рассматриваться как ***второе, независимое описание содержания последовательных изменений сопровождаемого комплекса программ***.

Жизненный цикл и развитие тестов при 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 сопровождаемых компонентов комплексов программ, но в иной форме. Их следует рассматривать и анализировать как еще одну форму описания функций программ и данных ПС, которое также может 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 текстов, определение их места и устранение является основной задачей тестирования при сопровождении.

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. 3.5). Тестовые спецификации могут использоваться для проверки согласованности, внутренней непротиворечивости и полноты реализации требований на модификации без исполнения программ.

Для каждого требования к изменению комплекса программ, его архитектуры, функциональных компонентов и модулей должна быть разработана спецификация тестов, обеспечивающая проверку корректности, адекватности и возможности в последующем реализовать тестирование изменений компонента на соответствие этому требованию. Такая взаимная проверка при сопровождении корректировок функций компонентов, отраженных в требованиях и в спецификациях тестов, обеспечивает повышение их качества, сокращение дефектов, ошибок, неоднозначностей и противоречий.

При тестировании изменений программ зачастую оказывается, что не каждое новое требование в спецификации модификации 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 модификации*** программ или данных может быть подготовлен тест. С другой стороны ***не для каждого теста*** при сопровождении может оказаться в спецификации адекватное требование на корректировку ПС. Спецификации тестов должны обеспечивать дополнительный контроль корректности корректировок спецификаций требований и верификацию взаимодействия измененных компонентов на соответствующем уровне описания ПС. Независимая разработка спецификаций тестов на основе спецификаций требований на модификации, создает базу для обнаружения, какие требования не тестировались или принципиально не могут быть проверены тестированием.

Таким образом, ***верификация спецификаций требований на изменения*** программных компонентов и ПС могут использоваться с ***двумя целями*** (см. рис. 3.5):

- для разработки, программирования и проверки изменений текстов программ и интерфейсов взаимодействия программных компонентов разных уровней в модифицированном комплексе программ;

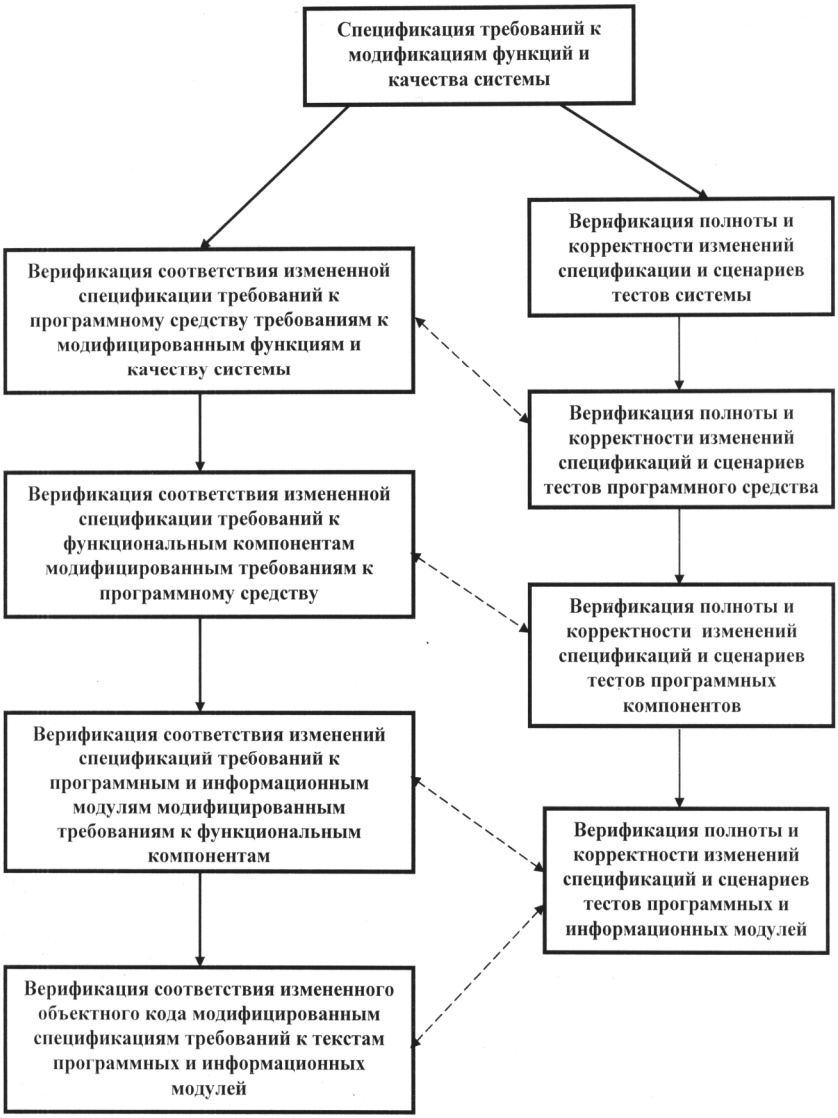

Рис. 3.5

- для создания скоординированного комплекса тестов для совокупности изменяемых компонентов, обеспечивающих вза-

имную проверку реализации спецификаций требований на модификации сопровождаемым комплексом программ.

В результате совокупности спецификаций требований к тестам могут применяться при сопровождении ***как эталоны и вторая адекватная форма описания содержания модификаций 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*** корректировок спецификаций требований и текстов программ, и спецификаций тестов способствуют выявлению и исключению множества вторичных дефектов и ошибок корректировок ПС. Впоследствии, при сопровождении эти спецификации тестов должны использоваться для непосредственного тестирования исполнения требований к модифицированным программным компонентам соответствующего уровня. Кроме того, параллельная и независимая разработка, с одной стороны, изменений спецификаций программ и спецификаций тестов, а также их реализации, с другой стороны, позволяет распараллеливать работы по сопровождению, что ведет к сокращению сроков создания модификаций компонентов и комплексов программ.

Реализация этих целей верификации и тестирования может производиться при сопровождении разными методами и независимыми специалистами – программистами и тестировщиками, что позволяет использовать результаты их деятельности для сравнения корректировок одних и тех же программ, представленных на языках программирования и описанных на языках тестов. Особенности описаний и реализации программ, а также ***мышления программистов при сопровождении*** – на основе функций и процедур исполнения корректировок программ, существенно ***отличаются от представлений и методов описаний тех же функций программ тестировщиками*** – создателями сценариев тестирования модификаций. Они акцентируют деятельность на конкретных процедурах проверки функционирования изменений, возможных результатах и взаимодействии модификаций компонентов ПС. Это позволяет выявлять вторичные дефекты, появляющиеся при корректировках, и повышать качество сопровождения путем сопоставления двух методов и результатов описания одних и тех же программ, за счет

того, что мала вероятность делать одинаковые ошибки в сценариях тестов и в реализации текстов программ.

Методы ***тестирования с исполнением программ***, в большей или меньшей степени, ориентированы при сопровождении на обнаружение ошибок определенных типов, преимущественно в измененной структуре программы и реализуемых новых маршрутах обработки информации. Методы тестирования потоков данных ориентируются на выявление ошибок в модификациях вычислительной части программ и в процессах преобразования различной информации. Такая ориентация позволяет при сопровождении упорядочить последовательность применения методов с целью устранения, прежде всего, ошибок модификаций, в наибольшей степени отражающихся на корректности изменений программ, а также сосредоточиваться на методах, позволяющих решать частные задачи достижения необходимого их качества и соответствия требованиям при минимальных затратах на корректировки.

Комплекс тестов и сценариев тестирования представляет собой определенный продукт, которому должен соответствовать специфический жизненный цикл (ЖЦ). Этот жизненный цикл во многом подобен типовому ЖЦ программных средств и состоит из ряда аналогичных этапов и компонентов. Его целесообразно далее использовать в качестве основы при представлении методов и процессов тестирования модификаций при сопровождении сложных программных средств – рис. 3.6. После системного анализа и верификации совокупности основных требований к сопровождению компонентов и ПС в целом, должны начинаться регламентированные последовательности процедур, обеспечивающих и реализующих предполагаемое и выполненное их тестировании.

Для этого Заказчик анализирует и, в конечном счете, утверждает ***план тестирования***, в котором описаны требования к тестированию и представлена ***матрица соответствия системных требований и тестовых сценариев*** [9]. Планирование тестирования включает как определение требований к тестам, так и разработку процессов управления этими требованиями.

Рис. 3.6

Матрица отслеживания изменений требований явным образом определяет каждое требование, проверяемое группой тестирования, и связанный с ним метод контроля. Матрица должна

отображать тестовые процедуры на системные требования или сценарии использования системы, позволяя тестировщикам подтвердить, что новые системные требования или сценарии использования системы, требующие проверки, полностью и успешно реализованы. Матрица отслеживания требований представляет собой продукт, получаемый при помощи инструмента управления требованиями. Она позволяет проследить системные требования и сценарии использования системы, а также покрытие требований тестовыми процедурами. В матрице соответствия обычно перечисляются тестовые процедуры, относящиеся к модификациям, и показывается соответствие между ними и системными требованиями, а также между требованиями к тестам и системными требованиями или сценариями изменений применения системы. Подпись заказчика под ***планом тестирования*** при сопровождении означает, что он утверждает представленную в матрице степень покрытия требований на модификации, проверяемых группой тестирования. Чтобы получить как можно раньше информацию от заказчика о степени покрытия новых требований, группа тестирования может передавать ему предварительную версию матрицы соответствия требований.

Планирование тестирования предполагает как определение требований к тестам, так и процессов управления ими. Системные требования или сценарии использования системы обычно сопровождаются с помощью инструмента управления требованиями. После определения тестовых процедур во время планирования тестирования и проектирования тестов они заносятся в базу данных инструмента управления требованиями и привязываются к соответствующим системным требованиям или сценариям использования системы. Управление требованиями к тестам включает в себя хранение изменений требований, отслеживание связей, оценку рисков при реализации требований к тестам, выстраивание последовательности требований к тестам и определение методов верификации тестов. Отслеживание связей предполагает отображение тестовых процедур на требования к тестам и выявленных дефектов на тестовые процедуры.

Планирование – компонент методологии жизненного цикла тестирования при сопровождении, включает анализ всех работ и процессов, необходимых для проведения тестирования изменений. Он предполагает, что процессы тестирования, методы, ме-

тодики, персонал, инструменты, план-график и оборудование организованы и эффективно применяются при сопровождении. Группа тестирования модификаций и выявления дефектов начинает подготовку плана тестирования с получения или создания шаблона плана тестирования, а затем уточняет план. Когда план тестирования изменений ПС разработан, обновлен и полностью описывает процессы тестирования, он становится руководящим инструментом для выполнения Программы тестирования. Он уточняется посредством ***корректировок целей, задач и стратегий тестирования***, а также изменений требований к тестам, фиксирует параметры Программы тестирования, которые должны быть документированы.

Планирование работ по тестированию изменений должно учитывать ресурсы и работы, которые необходимо выполнить, чтобы своевременно ***подготовить тестовую среду***. Тестировщики должны определить требования к аппаратному, программному и сетевому обеспечению с целью создания и поддержания адекватных изменений тестовой среды сопровождения. Нужно спланировать работы по приобретению, установке и настройке компонентов или моделей тестовой среды. Создание плана тестирования – итеративный процесс, требующий обратной связи с различными участниками сопровождения ПС и их согласия с определенными в нем процессами, стратегиями тестирования и сроками выполнения работ. Заказчик должен утвердить стратегию тестирования и тестовые процедуры при сопровождении ПС, которые подробно описаны в плане тестирования и определяют, какие сценарии и тесты когда будут выполняться. Кроме того, предполагается, что заказчик согласен с тем, что план тестирования и связанные с ним тестовые сценарии достаточно проверяют покрытие тестами модификаций системных требований или сценариев использования системы. Подробное изучение системных требований или сценариев сопровождения системы вместе с тщательным определением параметров Программы тестирования и требований к тестам необходимы для эффективного выполнения программы тестирования модификаций ПС [9, 21].

Эффективная Программа тестирования, включающая в себя автоматизацию тестирования программного продукта, должна иметь свой собственный жизненный цикл, в который входят планирование стратегий и целей, определение требова-

ний к тестам, анализ, проектирование и кодирование. По аналогии с процессом, которому следуют при разработке и программировании ПС, требования к тестам необходимо определить прежде, чем тесты будут спроектированы. Требования к тестам должны быть ясно сформулированы и документированы, чтобы все участники сопровождения понимали, на чем основаны работы по тестированию.

Как и при создании текстов ПС, Программа тестирования должна быть спланирована и спроектирована для того, чтобы проводимые в рамках тестирования работы привели к созданию наиболее эффективных тестов для проверок модификаций системы. Ресурсы Программы тестирования ограничены, а способов тестирования изменений системы много. Представление работ по тестированию в графической форме, позволяет специалистам по тестированию оценивать границы и масштаб Программы тестирования. Завершив анализ, тестировщики переходят к созданию моделей оформления Программы тестирования, которая в графической форме описывает масштаб Программы. Определившись с моделью Программы тестирования, тестировщики разрабатывают архитектуру тестирования, которая показывает структуру Программы тестирования и организацию тестовых процедур. Структуру Программы тестирования (архитектуру тестирования) обычно представляют двумя способами. Один метод организации тестовых процедур, известный ***как архитектура тестирования на базе архитектуры системы***, разбивает тестовые процедуры по компонентам системной архитектуры по логическому принципу её иерархии. Второй метод ***архитектура тестирования на базе выбранных методов*** связывает тестовые процедуры с различными методами тестирования, представленными в модели Программы тестирования.

Методы тестирования ***белого ящика*** направлены на проверку внутренней работы программы, в то время как методы тестирования ***черного ящика*** обычно сравнивают результаты функционирования ПС с требованиями, используя при этом установленные пользовательские интерфейсы. В процессе тестирования методами белого ящика тестировщик измеряет глубину тестирования, собирая данные о покрытии путей исполнения программ и полноте тестового покрытия. В ходе тестирования методами черного ящика сбор измерений сосредотачивается

на широте варьирования тестирования, включая объем продемонстрированной функциональности и объем проведенного тестирования программного продукта. При выборе тестирования изменений методами белого и черного ящиков в качестве части общей Программы тестирования полезно выбирать инструменты тестирования, которые можно использовать для поддержки разработки и выполнения соответствующих тестовых процедур.

При проектировании необходимо разбить тестовые процедуры на логические группы и определить соглашения по именованию для всего комплекта и компонентов тестовых процедур. Такие тесты могут включать в себя тестовые процедуры со сложными алгоритмами, процедуры, состоящие, из автоматизированных этапов и этапов, выполняемых вручную, и тестовые программные сценарии, адаптированные для использования во многих тестовых процедурах. В процессе детального проектирования в первую очередь необходимо изучить, определение тестовых процедур для анализа изменений на системном уровне. Это исследование позволит группе тестирования выявлять наиболее сложные тестовые процедуры, которые должны быть спроектированы детально [9, 11, 39].

Детализированный проект тестов может принимать форму псевдокода программы тестирования в том случае, если тесты приходится программировать. Он может представлять собой обычную последовательность шагов, которые необходимо выполнить в процессе тестирования модификаций. При использовании программных переменных и больших массивов данных детальное проектирование может предполагать использование цикла для итеративного выполнения тестов, применяющих различные значения, а также списка или таблицы, определяющей виды данных, необходимых для тестирования изменений. После того как будут сформулированы требования к тестовым данным, группа тестирования должна спланировать использование средств получения, генерирования или создания тестовых данных.

Разработка тестов включает создание тестировщиками, сопровождаемых, многократно применяемых, простых и надежных тестовых процедур, что может потребовать не меньше усилий, чем разработка программистами тестируемых текстов программ. Чтобы добиться максимального эффекта от автоматизации тестирования, тестировщики должны вести разработку тес-

тов так же, как их проектирование, параллельно с созданием программистами текстов программного продукта. Схема структуры модификаций комплекса программ является графическим представлением основных работ, которые должны быть выполнены во время разработки тестов. Группе тестирования требуется провести корректировку и уточнение структуры разработки тестов, чтобы отразить приоритеты модификаций и компонентов конкретного программного продукта.

Анализ полноты покрытия тестами требований на изменения при исполнении программ должен определять, какие требования не были протестированы и какие части корректировок структуры программного средства не были исполнены при тестировании. Тестирование структурного покрытия должно устанавливать, не пропущены ли элементы структуры программы, которые не проверены тестовыми процедурами и покрыли ли тесты всю структуру модифицированной программы. Тестовые варианты, основанные на требованиях, могут не полностью покрыть новую, измененную структуру программы. Анализ при сопровождении покрытия версии программы тестовыми данными, основанного на требованиях, должен определить, насколько полно тестирование проверило реализацию всех требований к модификации ПС, и показать потребность в дополнительных тестовых сценариях. Поэтому должен выполняться анализ полноты структурного покрытия и проводиться его верификация. При тестировании изменений модулей и программных компонентов полнота их покрытия тестами редко достигает 90% и хорошо, если составляет около 80% [9, 21].

Тестирование ***измененных потоков управления*** в структуре программы при сопровождении должно быть начальным этапом, так как при некорректной модификации структуры возможны наиболее грубые искажения выходных результатов и даже отсутствие некоторых из них. Оно состоит в проверке корректности последовательностей передач управления и формирования маршрутов исполнения модифицированной программы при обработке тестов. Для тестирования корректировок структуры программ в большинстве случаев требуются относительно меньшие затраты по сравнению с тестированием на потоках данных.

Тестирование ***изменений потоков данных*** при корректировках можно разделить на два этапа. Первый этап тестирования состоит в анализе модификаций обработки данных, определяющих значения предикатов в операторах выработки логических решений в тексте программы. Эти решения влияют на изменения маршрутов обработки информации, что сближает в этой части метод тестирования потоков данных с тестированием структуры программы. Второй этап тестирования модификаций обработки данных, на соответствие заданным требованиям, состоит в проверке изменений вычислений по аналитическим формулам или новых численных значений результатов, в зависимости от изменений числовых или логических значений исходных данных.

При проведении тестирования на основе требований к модифицированному программному продукту, практическим правилом для определения и выбора конкретных тестов может служить критерий, по условиям которого ***на каждое требование модификации полагается использовать, по меньшей мере, один комплексный тест***. Помимо того, что это правило при сопровождении определяет эффективные размеры совокупности тестов, оно упрощает планирование проведения испытаний. На основе анализа технического задания и требований на модификации можно получать представление о том, сколько нужно новых тестов для полного испытания откорректированной версии ПС. Если составлять тесты, руководствуясь этим принципом, можно оценивать, какое время потребуется на разработку, исполнение и анализ полного комплекта тестов при создании новой базовой версии ПС.

Группа тестирования при сопровождении должна составлять план-график разработки и выполнения тестовых процедур как средство определения временных рамок для разработки и выполнения различных тестов. План-график определяет зависимости между тестами и общие сценарии, которые будут постоянно использоваться при тестировании. Перед созданием полного набора тестовых процедур для очередной версии ПС группа тестирования должна провести ***анализ межмодульных связей***. Результаты этого анализа помогут выявить независимые данные, спланировать зависимости между тестами и выделить общие сценарии, которые могут повторно применяться в процессе тестирования последующих версий. Для этого строится матри-

ца межмодульных связей; она показывает взаимозависимость различных тестовых сценариев. Графическое представление её помогает тестировщикам определить возможности многократного применения сценариев в различных комбинациях с использованием различных оболочек, что позволяет свести к минимуму объем работ по созданию и сопровождению версий тестовых сценариев. В ходе разработки тестовых процедур группа тестирования должна проводить ***конфигурационное управление и контроль*** для тестовых сценариев и тестовых данных, а также для каждой отдельной тестовой процедуры. Группы тестовых процедур и сценариев многократного применения обычно хранятся в списке или библиотеке тестовых данных – в тестовом архиве. Необходимо создавать базовые версии тестового архива изменений при помощи инструментов конфигурационного управления (см. главу 2).

Во время выполнения тестовых работ, при разработке, исполнении тестов и обнаружении дефектов версий ПС возможен разброс производительности тестировщиков в 5 – 8 раз [9, 21, 52]. Среди тестировщиков есть представители, которые могут служить примерами специалистов по тестированию. Однако обычным является анализ и применение достаточно квалифицированной группы, которую составляют средние специалисты, обладающие различными достоинствами и навыками. Ни один человек не может быть носителем всех необходимых высоких качеств, однако группа тестирования, будучи единым коллективом, может воплощать максимально возможное количество требуемых качеств. При сопровождении комплексов программ основным лимитирующим ресурсом обычно являются допустимые трудозатраты специалистов, а также ограничения на сроки разработки модификаций базовой версии, на параметры ЭВМ и технологию проектирования изменений. Одним из наиболее важных компонентов планирования верификации и тестирования при сопровождении является оценка трудоемкости и времени, необходимых для его выполнения. Затраты на тестирование значительных модификаций версий сложных ПС могут составлять существенную часть (≈ 50%) стоимости проекта, при этом жизненно важно для успеха этой операции, чтобы тестирование проводило достаточное число специалистов и у них было достаточно времени на качественное выполнение

задач по корректировкам комплекса программ. Ограничения реальных ресурсов на верификацию и тестирование определяют достижимое качество выполнения модификаций и создание базовых версий комплексов программ.

Разработке тестовых процедур должны предшествовать работы по ***настройке и установке тестовой среды, по подготовке необходимой документации***. Тестировщики должны сформулировать инструкции по разработке тестовых процедур, которые будут использоваться при сопровождении для организации разработки тестов. Инструкции должны применяться для разработки как ручных тестов, так и автоматизированных тестовых процедур многократного применения. Инфраструктура автоматизации – это библиотеки функций многократного применения, которые создавались для тестирования в различных проектах либо отдельных версий ПС. Главное в этих функциях – возможность их многократного использования, что сводит к минимуму дублирование работ по созданию тестовых процедур и повышает степень многократного их применения.

Группа тестирования должна придерживаться утвержденного ***графика проведения тестовых процедур***. По окончании выполнения теста следует производить анализ выходных данных тестирования и готовить документацию по результатам тестирования модификаций. Планы модульного, комплексного, системного тестирования и приемо-сдаточные испытания в совокупности представляют собой этапы, необходимые для тестирования при сопровождении системы в целом. Комплексное тестирование версии программного продукта направлено на проверку его внутреннего функционирования. В процессе комплексного тестирования модули интегрируются и тестируются совместно на основе управляющей логики. Поскольку одни модули могут состоять из других, то часть комплексного тестирования, может проводиться в ходе модульного тестирования. В процессе системного тестирования тестировщик проверяет интеграцию отдельных частей, в совокупности составляющих систему в целом. Тестирование на системном уровне обычно проводится отдельной группой испытателей.

Группа тестирования должна проводить тщательный анализ с целью определения ***корректности изменений компонентов функциональности***, которые вызывают наибольшее количество проблем. В результате анализа может потребоваться провести

дополнительные тестовые процедуры и другие работы по тестированию выявленных сомнительных компонентов. Анализ результатов тестирования также может показать, дало ли выполнение тестовых процедур результат для выявления ошибок и дефектов при сопровождении.

В процессе сопровождения большое значение имеет ***история эксплуатации ПС***, развития его версий и соответствующая документация. Еще на стадии проектирования первой базовой версии ПС могут возникать ***идеи совершенствования комплекса программ***, которые в то время невозможно реализовать из-за высокой стоимости, ограниченных сроков проектирования или по иным причинам. Идеи изменения могут быть направлены на коренное улучшение функциональных возможностей программ или некоторые "косметические" улучшения реализуемых функций. Идеи небольших корректировок программ целесообразно накапливать отдельно от предложений по существенному совершенствованию системы. Таким образом, должен создаваться документ – ***исходные данные*** для планирования доработок ПС в процессе сопровождения, содержащий разделы:

- выявленные дефекты и ошибки в программах;
- предложения по совершенствованию функций и улучшению качества эксплуатируемых версий комплекса программ;
- идеи и предполагаемая экономическая эффективность коренной модернизации и расширения функций базовой версии программного продукта.

На основании ***оценки рисков*** группируются дополнительные требования к тестам, чтобы уменьшить влияние функций повышенного риска на модификации системы и возникновение потенциальных помех реализации сопровождения. Некоторые требования к модификациям и тестам могут быть ранжированы довольно высоко в списке приоритетов, поскольку их часто применяют, либо конечные пользователи практически не имеют информации и квалификации в этой области применения ПС. Функции, относящиеся к техническим ресурсам и к внутренним пользователям, а также редко используемые функции могут ранжироваться как низкоприоритетные. Некоторые требования к тестам жизненно важны для пользователей. Если при тестировании не повысить внимание к этим требованиям, могут быть

нарушены договорные обязательства или компания понесет финансовые потери. Важно оценить влияние возможных проблем на требования конечного пользователя.

При разработке модификаций сложных ПС для верификации и тестирования требуются ***значительные ресурсы*** в течение всего ЖЦ ПС, и наиболее критичным ресурсом является допустимое время поэтапного выполнения этих процедур. Выделение и упорядочение компонентов и процедур для тестирования изменений в крупномасштабных ПС зависит от их архитектуры и реального состава, готовых к использованию компонентов. При систематическом ***восходящем тестировании,*** прежде всего, проверяются программные компоненты и/или модули нижних иерархических уровней в функциональной группе измененных программ, к которым последовательно подключаются вызывающие их модули. Последовательное наращивание модифицированных компонентов программ ***снизу вверх*** позволяет проверять при сопровождении работоспособность таких компонентов в их естественном исполнении, без подмены и имитации компонентов нижних уровней.

При восходящем тестировании главная задача – обеспечить укрупнение, ***интеграцию*** и корректное взаимодействие при сопровождении всех компонентов для полного решения задач откорректированным комплексом программ. С учетом особенностей применения методов и технологических этапов ЖЦ программных компонентов, обычно последовательно рассматриваются задачи тестирования модификаций (см. рис. 3.4):

- автономных программных модулей, запрограммированных и подготовленных к тестированию корректировок на уровне исходных текстов программ и на уровне объектных кодов реализующей ЭВМ;
- автономных групп модифицированных программных модулей и компонентов, решающих законченные функциональные задачи;
- основных, функциональных компонентов в составе сопровождаемых программных продуктов.

Тестовые сценарии должны покрывать все новые требования и модификации функциональных компонентов ПС. Следует выполнять все необходимые изменения ПС, связанные с коррекцией дефектов, выявленных в процессе верификации, а также повторное тестирование в необходимом объеме и модифици-

ровать данные сопровождаемого ПС и другие программные продукты, основываясь на результатах интеграционного тестирования.

Если ***изменения в программе или в данных невелики***, то тестировщики обычно стремятся ограничиться компонентами, непосредственно связанными с выполненной корректировкой. Для этого выделяются для тестирования компоненты, которые связаны по информации или по управлению с теми, которые подверглись даже малым изменениям.

Тестирование вновь разработанных модулей и компонентов программ необходимо для проверки при сопровождении корректности внесения изменений в базовые версии программного продукта. При этом возможно ***предсказание*** мест в программе, где наиболее вероятны вновь внесенные при корректировке дефекты и ошибки и каких типов они могут быть. Поэтому не всегда необходимо каждую новую базовую версию программного продукта подвергать столь же широкому тестированию, как первую или предшествующую версию. Тестирование необходимо сосредоточивать на компонентах и функциях впервые вводимых или значительно модифицируемых в данной версии. Таким образом, тестирование должно приобретать управляемый, регулируемый характер. Ресурсы на тестирование ***следует распределять*** в зависимости от причин, мест и характеристик реализуемых изменений в программах и данных.

Группа тестирования иногда может воспользоваться специальными инструментами – ***генераторами тестовых данных***, которые на основе некоторого набора правил автоматически генерируют сценарии и тесты [9, 27, 42]. Эти правила можно получить из спецификаций, из документации базы данных тестов; тестировщики могут изменить их вручную, чтобы привести в соответствие с требованиями. При необходимости генераторы могут быстро создать тестовые данные, например, для проведения нагрузочного тестирования. Некоторые генераторы тестовых процедур обладают высокой степенью интеграции в процесс анализа и проектирования программных продуктов и позволяют разработчикам тестировать функции ПС и БД в соответствии со спецификациями системной архитектуры. Другие генераторы могут извлекать информацию о документированных требованиях из архивов программных продуктов и создавать

тестовые процедуры автоматически. Поскольку генерация автоматизирована, можно приступать к созданию тестовых процедур сразу после завершения записи требований ПС или компонентам.

При тестировании и испытаниях ***корректности модификаций функциональных компонентов,*** сопровождаемых комплексов программ ***реального времени*** целесообразно выделять этапы:

- комплексирование модулей и тестирование автономных функциональных групп программ без взаимодействия с другими компонентами и возможно без подключения к операционной системе реального времени;
- тестирование функциональных групп программ с учетом взаимодействия с некоторыми другими сопровождаемыми компонентами и с базой данных;
- тестирование отдельных сопровождаемых программных компонентов в реальном времени во взаимодействии с другими функциональными компонентами и с основными компонентами операционной системы и базы данных.

После комплексирования откорректированных функциональных компонентов начинаются их тестирование и испытания в составе сопровождаемого комплекса программ в целом. Для них наиболее характерны следующие ***этапы квалификационного тестирования и испытаний базовой версии программного продукта в реальном времени***:

- по данным моделирующего стенда или генераторов тестов, имитирующих отдельные объекты и функционирование 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-пользова-телей.

Внутренние квалификационные испытания качества сопровождаемых программных средств**,** которые зачастую совмещаются с завершением комплексной отладки требуемых модификаций, должны ***оформляться документально*** и являются основанием при предъявлении версии программного продукта заказчику на квалификационные испытания, для завершающего оценивания характеристик качества выполненных изменений или новой базовой версии программного продукта. Сопроводи-

тель должен реализовать и оценить предъявляемые комплекс программ, тесты, результаты тестирования и документацию для пользователя, учитывая:

- полноту охвата испытаниями всех требований спецификаций на изменения к компонентам и к сопровождаемому комплексу программ в целом;
- согласованность с договором, требованиями заказчика и ожидаемыми результатами применения модифицированной версии программного продукта;
- возможность интеграции и тестирования модификаций или базовой версии программного продукта в составе системы;
- возможность функционирования и сопровождения последующих базовых версий программного продукта в соответствии с требованиями договора с заказчиком.

Каждая группа тестирования должна составлять ***отчеты о проблемах и результатах тестирования*** в соответствии с установленными правилами и уровнем ответственности. Составление отчетов о проблемах и их отслеживание значительно облегчается при применении автоматизированных инструментов по контролю дефектов и корректировок. Тест-менеджер несет ответственность за выполнение тестов согласно плану-графику, а также за распределение и перераспределение работ между тестировщиками для разрешения проблем, возникающих в ходе тестирования модификаций. Для эффективного выполнения этих функций тест-менеджеру необходимо отслеживать ход тестирования и готовить сводные отчеты [27] .

Этапы и процессы квалификационного тестирования ПС и БД ***с целью удостоверения для заказчика достигнутых характеристик*** качества сопровождаемого комплекса программ и его компонентов ***в составе системы*** регламентированы в стандартах **ISO 12207, 16326, ISO 15504**. В них выделены три основных, функциональных ***этапа реализации квалификационного тестирования и испытаний***:

- квалификационное тестирование корректировок функциональных компонентов и программного продукта в целом вне аппаратуры системы;
- интеграция и тестирование программного продукта в целом в составе аппаратуры системы;

- квалификационное тестирование и полные испытания сопровождаемой системы в комплексе с модифицированным программным продуктом.

Квалификационное тестирование качества сопровождаемой системы в целом выполняется, чтобы продемонстрировать заказчику, что удовлетворены все требования технического задания на изменения функций и характеристики программного продукта соответствуют условиям договора на сопровождение. Все полученные результаты должны быть включены в отчет о тестировании изменений и в базовые версии программного продукта и системы. Кроме того, при сопровождении должен быть доступен архив разработки версий комплекса программ, чтобы избежать повторения ошибок, допущенных при его разработке.

Комиссия при испытаниях модифицированной версии программного продукта должна руководствоваться ***следующими документами***:

- утвержденными заказчиком и согласованными с разработчиком договором, техническим заданием и спецификациями требований на сопровождение ПС;
- действующими государственными и ведомственными стандартами на жизненный цикл и испытания комплексов программ, на технологическую и эксплуатационную документацию;
- Программой испытаний реализаций модификаций по всем требованиям договора, технического задания и спецификаций на программный продукт;
- методиками испытаний, охватывающими каждый раздел требований технического задания, спецификаций и Программы испытаний модифицированного программного продукта;
- комплектом адекватной откорректированной эксплуатационной и технологической документации на сопровождаемый комплекс программ.

Документирование процессов тестирования модифицированных программных компонентов следует проводить непосредственно при выполнении каждой из соответствующих работ с целями изменения программ. Должен быть описан и документирован, упорядочен и конкретизирован весь технологический процесс тестирования модификаций и частные результаты на каждом этапе сопровождения ПС. Результаты испытаний фиксируются в протоколах (см. **ISO 12119**). Протоколы по всей программе испытаний должны ***обобщаться в акте,*** в результа-

те чего делается заключение о соответствии модификаций сопровождаемой системы требованиям заказчика и о завершении этапа работы с положительным или отрицательным итогом. Завершаются квалификационные испытания предъявлением заказчику на утверждение ***комплекта документов***, содержащих ***результаты комплексных испытаний совокупности изменений или базовой версии программного продукта***.

Сертификация базовой версии программного продукта служит для независимого удостоверения высокого качества, проведенных модификаций и корректировок в испытанной и утвержденной разработчиком базовой версии программного продукта [16, 27]. По заявке разработчика, заказчика или пользователей она может подвергаться обязательной или добровольной сертификации. Комплект, состоящий из копии физических носителей и эксплуатационной документации, а также применявшиеся разработчиками тесты, Программа и методики испытаний передаются сертификационной лаборатории. При сертификационных испытаниях допускается расширение набора и параметров тестов, однако, в пределах, ограниченных технической документацией на конкретную версию программного продукта. Успешно проведенные испытания и полученный ***сертификат качества*** подтверждает соответствие комплекса программ документации, надежность и безопасность применения очередной базовой 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.

Многолетний опыт показал, что единственный универсальный метод верификации и тестирования при сопровождении ПС, создать невозможно и следует применять регламентированную технологию и упорядоченный ряд взаимодействующих и значительно различающихся методов. Каждый метод отличается целевыми задачами верификации и тестирования, проверяемыми компонентами программы и методами оценки результатов. ***Только последовательное и систематическое применение при сопровождении совокупности методов верификации и тестирования позволяет достигать высокое качество*** функционирования версий сложных программных продуктов.

Следует подчеркнуть, что выше анализировалось и оценивалось преимущественно ***изменение функциональной пригодности*** при совершенствовании и модификациях конструктивных характеристик ПС. Однако для заказчика и пользователей доминирующее значение могут иметь номенклатура и особенности реализации некоторых новых функций комплекса программ, которые, как правило, требуют наибольших затрат и определяют основной эффект от применения сопровождаемого комплекса программ, 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 конкретную базовую версию программного продукта со стороны пользователей априори оценить трудно. Такие оценки могут проводиться на основе специальных маркетинговых исследований и опыта эксплуатации аналогичных комплексов программ или достаточно близких их прототипов.

3.5. Инструментальные системы для управления конфигурацией программных средств

Наиболее известными и популярными за рубежом являются следующие системы конфигурационного управления версиями программных средств: Delta корпорации Microsoft, PVCS Version Manager фирмы Intersolv и Clear Case компании Rational Software [40]. Эти системы автоматизируют управление версиями сложных ПС, при этом в PVCS реализован наиболее широкий выбор организационных средств, однако система несколько сложнее в освоении.

Система ***Delta*** проста в установке и в эксплуатации, имеет хорошо сконструированный интерфейс с пользователями, а также возможность распределения работ над одними и теми же программами, позволяет отображать различия между версиями в цветном окне. Удобно реализовано большинство функций УК проектами средней сложности. Для участвующих в проекте разработчиков, Delta выделяет локальные копии головных файлов для текущей работы, нужных каждому из них. При возврате этих файлов в архив определяется состав изменений, которые разработчиком могут быть снабжены комментариями. Пакет позволяет создавать версии файлов на ассемблере, Basic, Си,

Си++ и Фортране. В среде Windows, Delta хорошо взаимодействует с другими инструментальными средствами автоматизации разработки. Предусмотрены отчеты, в которых подробно описываются изменения, внесенные в модули проекта, и графически выделяются различия между двумя версиями. Первым версиям модификаций файлов присваивается номер 1.00, который по мере развития проекта соответственно возрастает. К пакету прилагается руководство и краткая справочная схема. Кроме того, имеются средства оперативной помощи для среды Windows.

PVCS Version Manager является наиболее известным, мощным и полнофункциональным пакетом для конфигурационного управления версиями в процессе разработки крупных проектов ПС. Пакет стал стандартом де-факто в области управления конфигурацией ПС. Система обслуживает весь ЖЦ ПС от анализа и проектирования спецификаций требований до испытаний и сопровождения. Он обладает совершенной системой управления версиями, позволяет распределять работы, поддерживает удобное взаимодействие групп разработчиков со сложными конфигурациями проектов ПС.

Система является мощным инструментом УК, отражающим в динамике развития иерархический план этапов работы над проектом. Решается широкий набор задач администрирования проекта по разработчикам, группам специалистов, функциональным задачам и их компонентам. С помощью компонентов PVCS Version Manager отслеживается вся информация об изменениях, включая автора, время, причину и содержание, что позволяет восстановить любую предыдущую версию ПС или его компонент. Для управления проектом, контроля доступа и модификации все файлы конкретного проекта объединяются в один архив. По мере разработки модулей их можно помещать на разные уровни иерархии проекта, выделяя некоторые из них для тестирования, а другие для окончательного производства, сборки версии и тиражирования. Число уровней не ограничено и определяется потребностями пользователей, причем каждый может быть защищен от несанкционированного доступа с целью внесения изменений в проект только на определенных этапах. Для обеспечения безопасности при групповой разработке в локальной сети имеется механизм административных функций и

защиты файлов. Он реализует различные блокировки, анализирующие конфликты одновременного доступа к объекту нескольких разработчиков и конфликты при поэтапной разработке проекта. PVCS позволяет распределять и объединять работы, когда несколько программистов трудятся над одним и тем же сложным модулем, а также выдает предупреждения, если строки программ вступают в конфликт друг с другом.

Отслеживание корректировок, ошибок и дефектов обеспечивает компонент PVCS Tracker. Он позволяет модифицировать и расширять функции ПС и автоматизировать сопровождение комплекса программ. При разработке ПС могут применяться различные языки программирования, различные СУБД, текстовые и графические системы. В PVCS применяются при инсталляции файлы компонентов конфигурации в формате ASC II, отвечающие за функционирование всей системы и предоставляющие пользователю возможность точно настраивать систему под характеристики создаваемого проекта, как для индивидуальных разработчиков небольших проектов, так и для крупных разработок, в которых участвуют сотни программистов.

PVCS Configuration Builder автоматизирует процесс надежного построения приложений, сокращает циклы сборки версий ПС, предотвращает и устраняет ошибки сборки. PVCS Reporter позволяет получать качественные и полные настраиваемые отчеты о состоянии проекта, его версий, компонентов и изменений. Пакет генерирует комплект подробных отчетов о версиях проекта ПС, а также фиксирует все различия между двумя версиями одного и того же файла, с указанием даты и разработчика, осуществившего изменения.

PVCS поставляется для большинства распространенных аппаратно-программных платформ. К пакету прилагается справочное описание и краткое руководство. Облегчает его освоение то, что его функциональность достаточно полно соответствует обычной технологии разработки ПС, а также наличие краткого самоучителя, который помогает достаточно быстро получить необходимые навыки для работы с пакетом.

Система ***Clear Case*** [40] поддерживает разработку комплексов программ, в том числе, территориально разделенными группами разработчиков и совместима с другими инструментальными средствами проектирования, базирующимися на ОС Unix. Пакет позволяет следить за изменениями, вносимыми

в компоненты ПС, фиксировать их авторов и время, оценивать влияние на другие части программы, а также минимизирует усилия по перекомпоновке, сборке и выпуску новых версий ПС с внесенными изменениями. Он обеспечивает менеджерам возможность управления деятельностью до 1500 программистов в пяти различных странах, при совместном создании проекта ПС объемом до 25 млн. строк.

Clear Case построена на принципах открытой архитектуры, которая позволяет реализовать широкий набор решений в области УК. Она используется в различных средах разработки для определенных типов приложений: экспертных и технологических систем, телекоммуникационных комплексов, финансовых приложений, Web-сайтов и других коммерческих и правительственных информационных систем. Clear Case предлагает, как общие, так и специализированные решения задач управления конфигурацией. Решения общего характера практически не зависят от того, как система Clear Case связана со средой разработки, поэтому определить наилучший способ применения к специфическим потребностям УК достаточно сложно. Набор функций общего характера обозначен как базовый вариант Clear Case. Для тех, кому средства УК требуются на короткий срок при минимуме усилий, в Clear Case предусмотрена поддержка готовой модели использования УК, обеспечивающей специальные решения:

- определить компоненты и редакции, составляющие основу проекта;
- задать правила, по которым будет вестись работа над проектом;
- указать, где будут располагаться физические объекты проекта;
- создать сборку проекта ПС.

Сотрудники компании Rational Software исследовали работу фирм, применяющих общий вариант Clear Case, и обнаружили, что многие из них следуют сходным шаблонам и моделям использования. Эти технологические процессы были изучены, а лучшие их элементы нашли отражение в базовой модели Clear Case, обобщив реализацию и обеспечив простоту конфигурации в разнообразных средах разработки.

Модель Clear Case основана на двух ключевых концепциях, которые проявляются в расширенных моделях использования, построенных на основе Clear Case: управлении конфигурацией на основе действий и управлении компонентами. Управление конфигурацией на основе действий представляет собой создание и управление целостным изменением как единой именованной сущностью, а не множеством версий файлов. Множество версий в Clear Case называется набором изменений. Действие – это именованный объект: определенная задача, дефект, особенность или требование, реализуемые в виде версий файлов в наборе изменений действия. Управление компонентами – метод деления комплекса программ на более мелкие управляемые фрагменты и сводится к объединению множества файлов и каталогов в более крупную сущность, которая может подвергаться версионному контролю как единое целое.

Clear Quest – программный продукт компании Rational, дополняющий Clear Case и обеспечивающий поддержку процесса управления запросами на изменения. Такое решение удобно для крупных проектов, нуждающихся в более прогрессивных средствах и в расширении возможностей средств УК. Система Clear Quest состоит из трех ключевых частей: интерфейс пользователя (клиентская часть); серверное ядро, обеспечивающее доступ к базе данных; программа-дизайнер для создания и настройки процессов УК. В Clear Quest имеется несколько готовых типов записей для дефектов и запросов на доработку. Если процесс управления запросами на изменения отличается от стандартного, можно задействовать предустановленные типы или создать собственные. Clear Quest предоставляет также базовые возможности версионного контроля для УК-схемы.

В модели УК руководитель проекта и интегратор получают необходимые инструменты для управления конфигурацией проекта наиболее удобным способом. Поскольку модель УК предоставляет объекты более высокого уровня и четко определенные процессы, значительное число действий, выполнявшихся ранее вручную, здесь автоматизированы. Таким образом, пользователю не приходится разбираться в разветвленной структуре, предназначенной для управления параллельной разработкой. Функционирование модели Clear Case описано следующим образом [40]:

- программные файлы и каталоги объединяются в версионные компоненты на основе системной архитектуры комплекса программ;
- руководитель создает проект и организует команду специалистов для работы над этими компонентами;
- разработчики вносят изменения в компоненты, файлы и каталоги на основании закрепленных за ними действий (задач, дефектов, запросов на доработку);
- новые версии файлов и каталогов ПС связываются с соответствующими действиями и образуют наборы изменений компонентов;
- действия и связанные с ними наборы изменений отправляются в общую интеграционную область проекта версии комплекса программ;
- новые редакции компонентов разрабатываются и тестируются;
- новые редакции компонентов собираются в единую базовую версию комплекса программ;
- базовая версия программного продукта тестируется и выпускается для поставки заказчику и пользователям.

Процесс функционирования и применения Clear Case может быть рассмотрен применительно к ролям участников проекта и выполняемым ими шагам. Каждая роль важна для успешного управления конфигурацией программного средства в целом, на практике один специалист способен выступать в нескольких ролях либо несколько сотрудников могут разделять обязанности, закрепленные за одной ролью.

Специалист по архитектуре отвечает за физическую реализацию системной архитектуры ПС, определяя, каким образом сгруппировать объекты проектирования и отобразить их на физические файлы и каталоги.

Менеджер по управлению конфигурацией ПС определяет: создание и поддержание работоспособности физической инфраструктуры, необходимой для реализации проектных решений, создание и обслуживание хранилищ, а также импорт существующих файлов и каталогов.

Руководитель – менеджер проекта отвечает за назначение и планирование задач, распределение компонентов по рабочим группам, а также за создание физического проекта Clear Case и

определение правил управления проектом. В крупных разработках обязанности руководителя проекта иногда распределяются между двумя менеджерами (см. п. 3.2). Обычно выделяют роль администратора, отвечающего за организацию работы и правила взаимодействия с заказчиком, и роль технического менеджера, который создает проект, управляет им и курирует работу специалистов. Руководитель проекта отвечает за создание и сопровождение УК-плана. Проектный план должен быть помещен под версионный контроль, и здесь руководитель проекта выступает как разработчик.

Разработчиком считается любой сотрудник, который вносит изменения в какой-либо элемент системы, находящийся под версионным контролем. Он отвечает за определение действий, которые необходимо реализовать, за внесение изменений в файлы и каталоги для реализации действий и отправку этих действий интегратору. Таким образом, в роли разработчика в системе УК может выступать руководитель проекта при корректировке проектного плана, специалист при написании документации, по тестированию при внесении изменений в тестовые сценарии, специалист по архитектуре при модификации проектной модели (см. табл. 2).

Интегратор применяет инструменты сборки, а также процессы и инструменты УК, используемые на предприятии. Он принимает ***действия*** от разработчиков, создает новые редакции компонентов, собирает компоненты системы, обеспечивает тестирование сборки и выпуск новой версии программного продукта. В случае крупных систем, системный интегратор отвечает за сборку всех компонентов в базовую версию программного продукта, проведение её тестирования и учет базовых версий для системы в целом.

Представленные инструментальные средства для конфигурационного управления версиями при разработке ПС пока имеют относительно малое применение в России. Наиболее известен пакет PVCS, однако и он используется весьма редко. Это определяется ***низкой проектировочной культурой отечественных специалистов*** и малым опытом создания крупных проектов информационных систем длительного применения. Для обеспечения качественного сопровождения сложных комплексов программ специалисты должны уметь использовать не только приведенные выше инструментальные средства, но и широ-

кий набор средств автоматизации, поддерживающих реализацию практически всего жизненного цикла программных средств. Эти средства активно развиваются и совершенствуются, их следует анализировать и выбирать с учетом характеристик и особенностей конкретных проектов комплексов программ, которые предстоит сопровождать длительное время. Ориентиром при таком выборе может служить достаточно полный обзор инструментальных средств в [9, приложение В], в котором делается акцент на автоматизацию тестирования и обеспечение высокого качества сложных программных продуктов в течение всего жизненного цикла.

Глава 4

СТАНДАРТИЗАЦИЯ СОПРОВОЖДЕНИЯ, УПРАВЛЕНИЯ КОНФИГУРАЦИЕЙ И ОТКРЫТЫХ СИСТЕМ ДЛЯ ПРОГРАММНЫХ СРЕДСТВ

4.1. Стандарты жизненного цикла, сопровождения и управления конфигурацией программных средств

Наиболее актуальна стандартизация процессов жизненного цикла комплексов программ при коллективной разработке и сопровождении крупных критических систем, к которым предъявляются высокие требования к качеству. В этих случаях особенно необходимо четкое планирование и управление технологическими процессами их ЖЦ. Созданы или разрабатываются стандарты, в той или иной степени отражающие процессы проектирования, поддержки эксплуатации и сопровождения программ и баз данных (рис. 4.1). Они ориентированы на ПС и БД, выполняющие важные функции в системах управления объектами, технологическими процессами или при обработке информации. В таких системах предъявляются особенно высокие требования к качеству и надежности решения задач. Применение таких стандартов полностью при создании и использовании простых программ, узкого или экспериментального назначения не всегда может быть оправдано. Однако они определяют ***современную культуру промышленного проектирования и сопровождения программ высокого качества***.

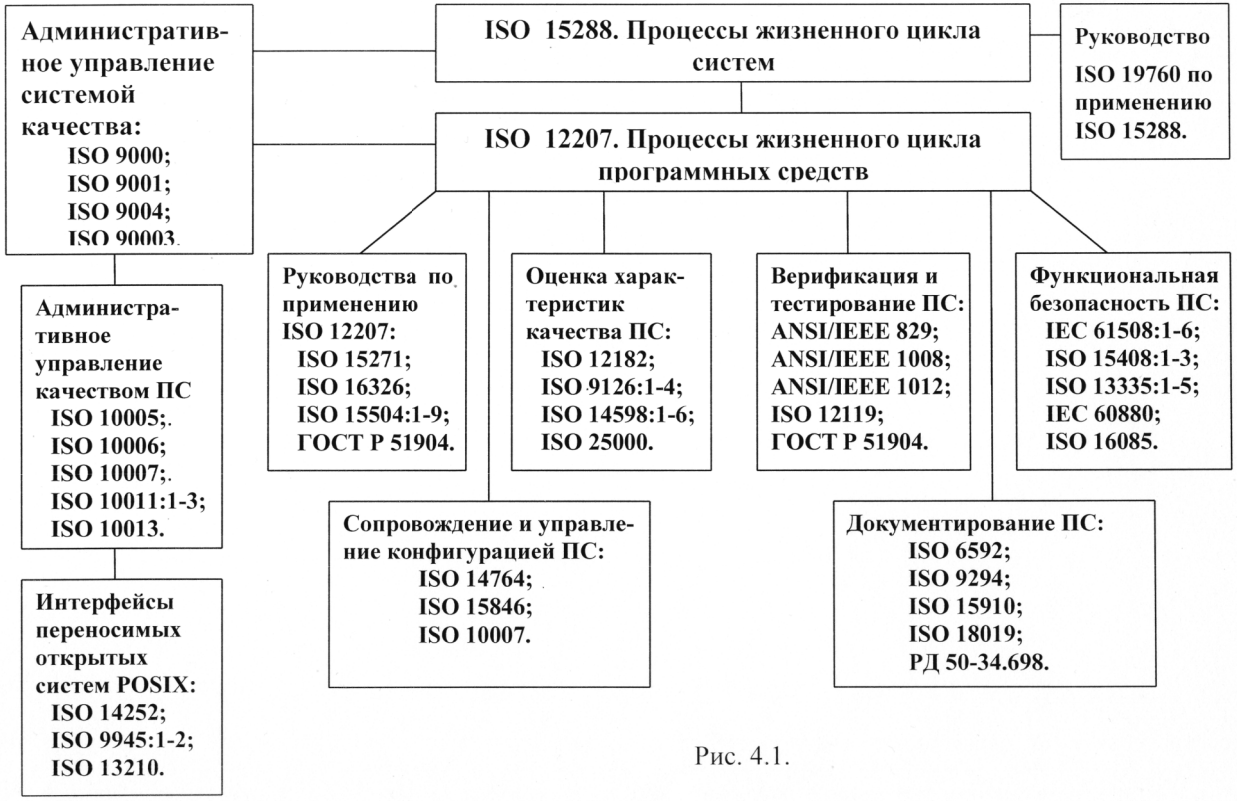
Административное управление системой качества:
ISO 9000;
ISO 9001;
ISO 9004;
ISO 90003.
ISO 15288. Процессы жизненного цикла систем
Руководство ISO 19760 по применению ISO 15288.
ISO 12207. Процессы жизненного цикла программных средств
Административное управление качеством ПС
ISO 10005;
ISO 10006;
ISO 10007;
ISO 10011:1-3;
ISO 10013.
Руководства по применению ISO 12207:
ISO 15271;
ISO 16326;
ISO 15504:1-9;
ГОСТ Р 51904.
Оценка характеристик качества ПС:
ISO 12182;
ISO 9126:1-4;
ISO 14598:1-6;
ISO 25000.
Верификация и тестирование ПС:
ANSI/IEEE 829;
ANSI/IEEE 1008;
ANSI/IEEE 1012;
ISO 12119;
ГОСТ Р 51904.
Функциональная безопасность ПС:
IEC 61508:1-6;
ISO 15408:1-3;
ISO 13335:1-5;
IEC 60880;
ISO 16085.
Интерфейсы переносимых открытых систем POSIX:
ISO 14252;
ISO 9945:1-2;
ISO 13210.
Сопровождение и управление конфигурацией ПС:
ISO 14764;
ISO 15846;
ISO 10007.
Документирование ПС:
ISO 6592;
ISO 9294;
ISO 15910;
ISO 18019;
РД 50-34.698.

Рис. 4.1.

Представленные в Приложении и на рис. 4.1 стандарты предназначены регламентировать управляемое и контролируемое развитие структуры, функций и состава компонентов сложных высококачественных информационных систем и программных средств в течение всего их ***жизненного цикла***. Они образуют ***профиль стандартов*** (см. п. 4.2), которые с учетом адаптации целесообразно использовать, в частности, при сопровождении и управлении конфигурацией программных средств. В данном случае рассматривается профиль нормативных документов, который предписывает базовый состав стандартов для реализации процессов развития и модификации функций и обеспечения качества сложных программных продуктов.

Процессы всего жизненного цикла ***сложных компьютерных систем***, включающих аппаратные компоненты, программные средства и их пользователей, регламентируются международным ***стандартом*** **ISO 15288**. Этот стандарт целесообразно учитывать как головной, в иерархии стандартов, поддерживающих процессы жизненного цикла, сопровождения и управления конфигурацией ***сложных программных средств***. Основные положения стандарта определяют процессы приобретения и поставки систем, выделены процессы предприятия, проекта и технические процессы. В процессы предприятия включено управление: инвестициями, процессами жизненного цикла продукции, ресурсами и качеством систем. В процессы проекта входят: планирование, оценка и контроль проекта, а также управление рисками, конфигурацией и информацией. Технические процессы, в основном, составляют: определение и анализ требований заказчика, проектирование архитектуры, реализация и комплексирование системы, а также верификация, аттестация, сопровождение и ликвидация системы. В приложениях обращается внимание на необходимость адаптации рекомендуемых в стандарте процессов к конкретным характеристикам предприятия и проекта системы, изложены базовые концепции стандарта **ISO 15288**, его соответствие и взаимосвязь со стандартами **ISO 12207** и **ISO 15504**.

При сопровождении и управлении конфигурацией сложных программных средств целесообразно акцентировать внимание в основном на рекомендации стандартов, непосредственно связанных с процессами развития и модификации компонентов и комплексов программ. Серия стандартов **ISO 9000** регламенти-

рует административные системы управления качеством продуктов и в том числе программных средств. К ним примыкают стандарты непосредственного ***управления качеством и переносимых Открытых систем POSIX*** (см. рис. 4.1), которые целесообразно использовать предприятиям, реализующим сопровождение ПС. Они позволяют упорядочить поток изменений в системах и комплексах программ, тем самым повысить их качество, а также сократить затраты ресурсов и длительность реализации предложений по их совершенствованию. Для этого необходима точная и достоверная информация о состояниях систем и их компонентов, всех предполагаемых и выполненных изменениях. Основную группу стандартов (профиль), определяющих жизненный цикл, сопровождение и управление конфигурацией программных средств возглавляет стандарт **ISO 12207**.

Стандарт **ISO 12207** – является основным международным нормативным документом, регламентирующим жизненный цикл и, в частности, сопровождение и конфигурационное управление программными средствами. При реализации этих задач целесообразно применять практически всю совокупность стандартов, представленную на рис. 4.1, которые, по существу, являются развитием и детализацией этого основного стандарта. Они в той или иной степени отражены в данной книге и в цикле монографий [22 – 29] перечисленных в списке литературы. Непосредственно сопровождение и управление конфигурацией ПС регламентируют стандарты **ISO 14764** и **ISO 15846**, а также **ISO 10007**. Остальные стандарты, входящие в профиль и отраженные на рис.4.1, целесообразно использовать при практическом решении основных задач процессов сопровождения ПС.

Непосредственно этим задачам в стандарте **ISO 12207** посвящены разделы: ***5.5. Процесс сопровождения; 6.2. Процесс конфигурационного управления*** и ***6.8. Процесс решения проблем*** – устранения дефектов в составе Вспомогательных процессов. Работы, обеспечивающие сопровождение ПС, представленные в разделе 5.5, 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 в первой главе книги. Процессы управления конфигурацией (п.6.2), представленные в **ISO 12207**, ниже изложены с некоторыми сокращениями текста стандарта.

6.2 Процесс управления конфигурацией

Процесс управления конфигурацией является процессом применения административных и технических процедур на всем протяжении жизненного цикла программных средств для: обозначения, определения и установления состояния базовой версии программных объектов 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.

6.2.1 Подготовка процесса

6.2.1.1. Должен быть разработан план управления конфигурацией. План должен определять: работы по управлению конфигурацией; процедуры и график выполнения данных работ; организацию, ответственных за выполнение данных работ; связь данной организации с другими организациями, например, по разработке и сопровождению программных средств. План должен быть документально оформлен и выполнен.

6.2.2. Определение конфигурации

6.2.2.1. Должна быть определена схема обозначения программных объектов и их версий (объектов программной конфигурации), которые контролируются при реализации проекта. Для каждого программного объекта и его версий должны быть определены: документация, в которой фиксируется состояние его конфигурации; эталонные версии и другие элементы обозначения.

6.2.3. Контроль конфигурации

6.2.3.1. Должны быть выполнены: обозначение и регистрация заявок на внесение изменений; анализ и оценка изменений; принятие или непринятие заявки; реализация, верификация и выпуск измененного программного объекта. Для каждого изменения должны отслеживаться проводимые аудиторские проверки, посредством которых анализируется каждое изменение, его причина и разрешение на его внесение. Должны быть выполне-

ны контроль и аудиторская проверка всех доступных контролю программных объектов, которые связаны с критическими функциями безопасности или защиты.

6.2.4 Учет состояний конфигурации

6.2.4.1. Должны быть подготовлены протоколы управления и отчеты о состоянии, которые отражают состояние и хронологию изменения контролируемых программных объектов, включая состояние их конфигурации. Отчеты о состоянии должны включать количество изменений в данном проекте, последние версии программных объектов, обозначения выпущенных версий, количество выпусков и сравнения программных объектов различных выпусков.

6.2.5. Оценка конфигурации

6.2.5.1. Должны быть определены и обеспечены: функциональная законченность программных объектов с точки зрения реализации установленных к ним требований; физическая завершенность программных объектов с точки зрения реализации в проекте и программах всех внесенных изменений.

6.2.6. Управление выпуском и поставка

6.2.6.1. Должны официально контролироваться выпуск и поставка программных продуктов вместе с соответствующей документацией. Оригиналы программ и документации должны сопровождаться в жизненном цикле. Программы и документация, связанные с обеспечением критических функций безопасности или защиты, должны обрабатываться, храниться, упаковываться и поставляться в соответствии с установленными правилами.

6.8. Процесс решения проблем

Процесс решения проблем (***устранения дефектов***) является процессом анализа и решения проблем (включая обнаруженные несоответствия), независимо от их происхождения или источника, которые обнаружены в ходе выполнения разработки, эксплуатации, сопровождения или других процессов. Целью данного процесса является обеспечение своевременного, ответственного и документируемого анализа и решения всех обнаруженных проблем (*дефектов*) и определения причин их возникновения.

6.8.1. Подготовка процесса

6.8.1.1. Должен быть установлен процесс решения проблем для обработки всех проблем (включая обнаруженные несоответст-

вия), выявленных в программных продуктах и работах. Процесс должен удовлетворять следующим требованиям:

a) процесс должен быть циклически замкнутым, обеспечивающим в соответствии с условиями договора: своевременное документирование и ввод всех обнаруженных проблем в процесс решения проблем; организацию работ над ними; соответствующие уведомления заинтересованных сторон о данных проблемах; определение, анализ и возможное устранение причин их возникновения; реализацию решения данных проблем и их внесение в соответствующие объекты; учет и документирование состояний проблем; сопровождение отчетов о проблемах;

b) процесс должен содержать схему классификации и установления приоритетов проблем. Для каждой проблемы должен быть определен соответствующий класс и приоритет для упрощения анализа причин ее возникновения и решения проблемы;

c) в отчетах о проблемах должен быть приведен анализ причин их возникновения;

d) реализованные решения проблем и их введение в соответствующие объекты должны быть оценены по следующим критериям: какие проблемы решены; какие неблагоприятные причины их возникновения устранены; какие изменения правильно внесены в соответствующие программные продукты и работы; какие дополнительные проблемы обнаружены.

6.8.2. Решение проблемы:

6.8.2.1. При выявлении проблем (включая обнаруженные несоответствия) в программном продукте или работе должен быть подготовлен отчет по проблеме, описывающий каждую выявленную проблему. Отчет по проблеме должен являться составной частью вышеописанного процесса, охватывая вопросы: выявления проблем; их исследования, анализа и решения, а также причин их возникновения; определения тенденций, способствующих возникновению проблем.

Все основные и вспомогательные процессы в стандарте подлежат адаптации и конкретизации применительно к характеристикам определенного проекта. Процессам адаптации в стандарте посвящены Приложения **A** и **B**, которые следует использовать также для процессов сопровождения и конфигурационного управления.

***Стандарт* ISO 15846** обобщает, детализирует и развивает основные концептуальные положения, представленные в стан-

дартах **ISO 12207** и **ISO 10007**. В обширном Введении указывается связь между этими тремя стандартами, назначение и преимущества применения на практике стандарта **ISO 15846**. В пяти начальных разделах отражены традиционные для стандартов **ISO** сведения об области применения и роли процессов управления конфигурацией, нормативных ссылках и аббревиатурах. Последующие шесть (6-ой – 11-ый) разделов начинаются с цитирования соответствующих шести базовых требований раздела 6.2 стандарта **ISO 12207**, представленных выше.

В каждом из этих шести разделов излагаются подробные рекомендации по реализации базовых требований **ISO 12207** и адекватных положений стандарта **ISO 10007** по управлению конфигурацией ПС. В Приложении **A1** и **A2** представлены детальные таблицы взаимного соответствия положений и содержания разделов этих трех стандартов. При этом в ссылках на стандарт **ISO 12207** отражены многие дополнительные, частные требования и разделы регламентирования процессов всего жизненного цикла ПС. Существенным достоинством стандарта **ISO 15846** является достаточно полное, подробное и систематичное изложение практических рекомендаций по управлению конфигурацией сложных комплексов программ, которые целесообразно использовать в крупных современных реальных проектах.

Стандарт* ISO 10007** (ниже с сокращениями и рекомендуется для ознакомления) содержит ***общие руководящие указания по управлению конфигурацией, которые рекомендуется применять на протяжении всего жизненного цикла ***различных систем и видов продукции***, с тем, чтобы обеспечить наглядность функциональных и физических характеристик и управление ими. Кроме того, цель руководящих указаний состоит в углублении общего понимания рассматриваемого предмета; в поддержке организаций, применяющих управление конфигурацией, в совершенствовании их деятельности; в урегулировании подходов к рассматриваемому вопросу во всех отраслях промышленности и укреплении национального и международного сотрудничества.

Стандарт применяется для поддержки проектов, начиная от концепции, охватывая проектирование, разработку, производство, эксплуатация, техническое обслуживание и кончая утилизацией продукции и систем. Он может быть адаптирован к кон-

кретным проектам с учетом объема, сложности и характера работы. Управление конфигурацией является управленческой дисциплиной, использующей техническое и административное руководство для разработки, производства и поддержки объектов конфигурации. Эта дисциплина применима к аппаратному оборудованию и программному обеспечению, перерабатываемым материалам, услугам и относящейся к ним документации.

Основной задачей управления конфигурацией является документальное оформление и обеспечение полной наглядности текущей конфигурации продукции и выполнения требований к её физическим и функциональным характеристикам. Другая задача заключается в том, чтобы все лица, работающие над проектом, в любой момент его жизненного цикла использовали достоверную и точную информацию. Управление конфигурацией следует организовать так, чтобы персонал знал свои обязанности и имел достаточно независимости и полномочий для выполнения поставленных задач.

Политика, виды деятельности и правила в области управления конфигурацией, характерные для конкретного проекта, должны быть определены в Программе управления конфигурацией. В этой Программе могут содержаться ссылки на процедуры управления конфигурацией, описанные в стандарте конкретного предприятия. Для того чтобы процесс был эффективным, важно, чтобы эти виды деятельности составляли единое целое. Структура продукции должна определять взаимосвязь и положения объектов конфигурации при разбиении продукция на компоненты. Объекты конфигурации следует выбирать с помощью процесса разложения продукции на компоненты. Все необходимые функциональные и физические характеристики объекта конфигурации, включая сопряжения, изменения, отклонения и разрешения на отклонения, должны содержаться в четко идентифицированных документах. Для идентификации объектов конфигурации, их деталей, документов, сопряжения, изменений, отклонений и разрешений на отклонение следует установить и применять правила нумерации.

Конфигурационные базы должны быть установлены официальным соглашением в определенные моменты времени и использованы в качестве отправных точек для официального контроля за конфигурацией. После первоначального выпуска документов на конфигурацию объекта все изменения следует кон-

тролировать. Влияние изменения, требования заказчика и подвергнутая влиянию конфигурационная база должны определять степень формальности, соблюдаемой при работе с изменением, и могут стать основой любой системы классификации, используемой для распределения по категориям данного изменения.

Контроль за конфигурацией включает следующие виды деятельности, которые должны быть подробно описаны в ***документированной процедуре контроля за изменением***:

- документирование и обоснование изменения;
- оценивание последствий;
- внесение и проверка реализации изменения;
- утверждение изменения.

В целях защиты ***целостности конфигурации*** и обеспечения основы для контроля за изменением важно, чтобы объекты конфигурации, их составные части и документация на них содержались в такой среде, которая:

- соответствует окружающим условиям, для аппаратного оборудования и программного обеспечения ЭВМ, данных, документов;
- защищает их от несанкционированного изменения или искажения;
- обеспечивает средства для восстановления после повреждения;
- в случае программного обеспечения, данных и документации позволяет осуществлять управляемый поиск копии оригинала.

Составление отчетности о статусе конфигурации должно начинаться с того момента, как будут получены первые данные о конфигурации. Отчетность о статусе конфигурации должна предоставлять информацию о всех идентификациях конфигурации и всех отклонениях от установленных конфигурационных баз. Это позволит прослеживать изменения конфигурационных баз. Записи и отчеты должны стать побочным результатом деятельности по идентификации и контролю. Проверки конфигурации следует проводить до принятия конфигурационной базы с целью гарантии того, что продукция соответствует контрактным или установленным требованиям и что она точно отражена в документах по конфигурации. Проверка

конфигурации может потребоваться для официальной приемки объекта конфигурации.

Процедуры, специфичные для данного проекта, и степень их применения в течение жизненного цикла следует определить в Программе управления конфигурацией. Объекты конфигурации выбираются с помощью процесса разложения. Этот процесс, идущий сверху вниз, разделяет общую структуру продукции на логически связанные между собой и соподчиненные комплексы аппаратного оборудования, программного обеспечения, или их сочетаний, выбираемых для управления конфигурацией. Выбор слишком большого количества объектов конфигурации отрицательно влияет на наглядность продукции, затрудняет управление и увеличивает расходы. Выбор слишком малого количества объектов конфигурации или недостаточная дробность разложения создает трудности для технического обслуживания, а также ограничивает возможности управления конфигурацией. Основным критерием является выбор таких объектов, эксплуатационными параметрами и физическими характеристиками которых можно управлять отдельно, чтобы достичь общих окончательных характеристик объекта.

Все физические и функциональные характеристики, необходимые для определения объекта конфигурации по всему его жизненному циклу, должны быть ***документально оформлены***. Типы документов, как правило, включают в себя технические требования, проектную документацию, перечни, данные по программному средству и руководства по эксплуатации и техническому обслуживанию. Документация, необходимая для объекта конфигурации, зависит от уровня требуемого контроля. Однако любая документация должна включать соответствующую информацию об изменениях и их прослеживаемости. Рекомендуется также присваивать предложениям об изменении ***уникальные идентификационные номера*** на самой ранней стадии, чтобы облегчить их прослеживаемость и идентификацию.

Необходимо выполнять и документировать следующие типичные ***оценки предложенных изменений*** с точки зрения:

- технических достоинств;
- влияния на взаимозаменяемость, средства сопряжения, а также необходимость повторной идентификации;
- влияния на контракт, график работы и расходы;
- влияния на методы производства, испытания и контроля

влияния на поставки и закупки;

- влияния на техническое обслуживание, справочники для потребителя и руководства.

После оценки изменения уполномоченное лицо или группа лиц должны проанализировать документированные оценки и решить вопрос об ***утверждении или не утверждении изменения***. Решение об утверждении должно быть документировано и разослано соответствующим лицам. Предпосылкой правильного ведения отчетности о статусе конфигурации является корректная идентификация и управление изменением. В системе отчетности о статусе конфигурации записывается и представляется информация для руководства процессом управления конфигурацией и связанной с ним деятельностью. Отчетность начинается с появлением первого документа по конфигурации и продолжается в течение всего жизненного цикла продукции. Такие отчеты могут выпускаться по отдельным объектам конфигурации или по готовой продукции.

Проверка функциональной конфигурации проводится путем идентификации отдельных функциональных и эксплуатационных требований к объекту конфигурации, выраженных в его функциональной базе, и последующего подтверждения выполнения требований с помощью изучения зарегистрированных данных анализа, контроля и испытаний. ***Проверка физической конфигурации*** выполняется путем изучения продукции ее соответствия документам по конфигурации. Такая проверка (вместе с проверкой функциональной конфигурации) должна подтверждать, что продукция, определяемая в документах по конфигурации, соответствует физическим и функциональным требованиям.

Проверки системы управления конфигурацией должны выполняться согласно документированным процедурам предприятия. Такие проверки выполняются для того, чтобы:

- удостовериться, что система управления конфигурацией эффективна и отвечает установленным требованиям;
- определить соответствие применяемых практических методов управления конфигурацией процедурам, описанным в соответствующей Программе.

Программа управления конфигурацией должна разрабатываться для применения на предприятии, для проектов или по

контрактным соображениям. Она обеспечивает для каждого проекта процедуры управления конфигурацией, которые должны применяться, и устанавливает, кто и когда должен их выполнять. В случае многоуровневой контрактной ситуации Программа управления конфигурацией главного подрядчика обычно является основной. Заказчик должен также подготовить Программу, которая описывает его участие в деятельности главного подрядчика по управлению конфигурацией. В Приложении **А** стандарта **ISO 10007** представлена рекомендуемая структура и содержание Программы управления конфигурацией.

4.2. Профили стандартов жизненного цикла, сопровождения и управления конфигурацией программных средств

При создании и сопровождении сложных, распределенных, тиражируемых ПС и БД 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 информационных систем (ИС). Для унификации и регламентирования реализации этих функций совокупности базовых стандартов должны адаптироваться и конкретизироваться применительно к определенным классам проектов, их функций, процессов и компонентов. В связи с этим выделилось и сформировалось понятие "профилей стандартов" ИС, как основного инструмента функциональной стандартизации [3, 22, 35, 60].

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 возможности 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 дол-

жен 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 ИС 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 при создании и применении ПС и БД являются:

- снижение трудоемкости, длительности, стоимости и улучшение других технико-экономических показателей проектов ИС;
- 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 ИС задач, ранее решавшихся в системе раздельно;
- обеспечение переносимости прикладных программ и данных между разными аппаратно-программными платформами.

Применение профилей при проектировании ИС позволяет ориентироваться на ***построение систем из крупных функциональных узлов***, отвечающих требованиям стандартов профиля, применять достаточно отработанные и проверенные проектные решения. Профили определяют стандартизированные интерфейсы и протоколы взаимодействия компонентов системы таким образом, что разработчику системы, как правило, не требуется вдаваться в детали внутреннего устройства этих компонентов. Таким образом, проектирование ИС в значительной степени может сводиться к ее компоновке и сборке из стандартизированных узлов.

Применение стандартизированных профилей позволяет заказчику системы освободиться от зависимости от одного по-

ставщика 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) ИС, облегчает повторное использование в проектируемой системе уже разработанных и проверенных программных компонентов. Профили ИС унифицируют и регламентируют только часть требований, характеристик, показателей качества 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.

Состояние и развитие стандартизации в области информационных систем 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 ИС 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 рекомендаций этих документов от современного состояния техники и от текущих потребностей практики и технологии создания сложных систем;
- стандарты современных ИС должны: учитывать необходимость построения ИС как открытых систем; обеспечивать их 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;
- в области ИС функциональными стандартами поддержаны и регламентированы только функционально наиболее простые объекты и рутинные, массовые процессы, такие, как телекоммуникация, программирование, документирование программ и данных;

- наиболее сложные и творческие процессы создания и развития крупных распределенных ИС (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 предварительных условий, учитываемых 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 ИС приводят к созданию и практическому применению многочисленных нормативных и методических документов отраслевого, ведомственного или фирменного уровня.

При практическом формировании и применении профилей ИС в ряде случаев, возможно, использовать региональные, национальные стандарты, стандарты де-факто 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 ИС могут определяться нормативными документами ведомства или предприятия, обязательными для конкретного профиля и проекта.

Целесообразно рассматривать ***две группы профилей*** ИС (рис.4.2):

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

- ***технологические профили***, регламентирующие процессы проектирования, разработки, применения, сопровождения и развития ИС и их компонентов.

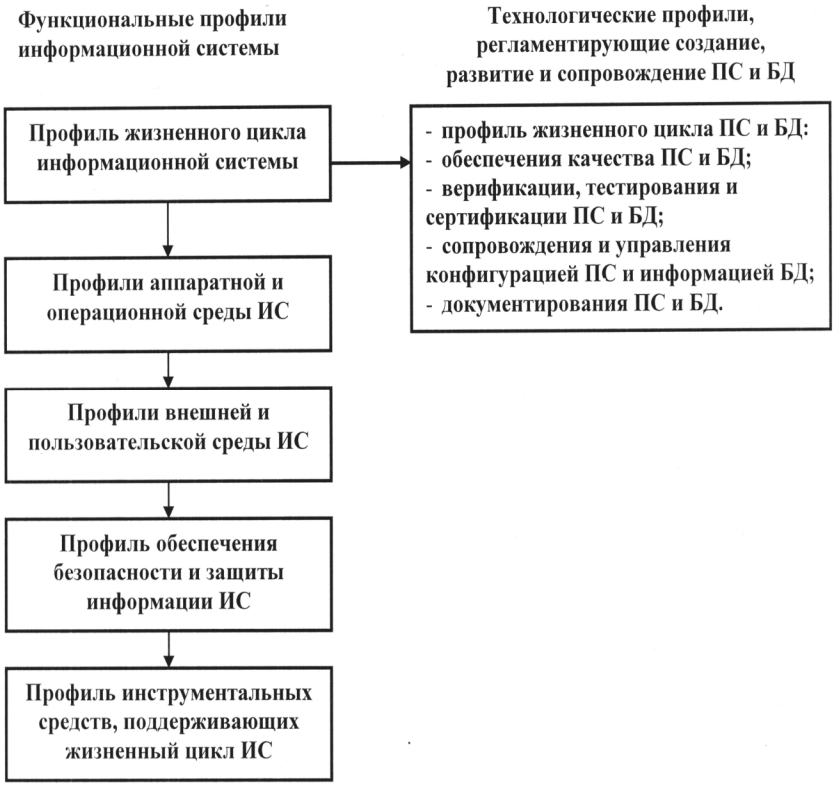

Рис. 4.2

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

- профили конкретной ИС, определяющие стандартизированные проектные решения в пределах данного проекта и являющиеся частью проектной документации;
- профили ИС, предназначенные для решения некоторого класса прикладных задач, которые распространяются на все ИС данного класса в пределах предприятия, отрасли или региона и утверждаются как стандарты предприятий, ведомственные или государственные стандарты.

Особенности организационных структур, различия в размерах и сложности проектов ИС, 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;
- 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.

Каждый профиль и его параметры для применения в конкретном проекте ИС необходимо поэтапно адаптировать и детализировать в соответствии с этапом проекта. ***В соответствии с основными процессами создания, сопровождения и развития системы и её компонентов*** они должны быть поддержаны ***этапами*** развития и применения ***комплекта профилей***, которые включают:

- системный анализ объекта информатизации и создание концепции ИС, когда производится первичный выбор исходного комплекта стандартов, которым должна соответствовать система; выявляется необходимость разработки и состав дополнительных нормативных документов; оформляется содержание и параметры комплектов документов предполагаемых профилей;

- проектирование ИС, когда определяются требования к её архитектуре и структуре и соответственно уточняются положения, параметры и адаптируются стандарты комплекта профилей; при необходимости они дополняются ведомственными нормативными документами; оформляются проекты документов и методических руководств по применению рабочей версии каждого профиля;
- разработку или приобретение готовых компонентов системы, при этом утверждаются и применяются все положения профиля; производится контроль, тестирование и испытания компонентов на соответствие требованиям и документам конкретного профиля;
- сопровождение, актуализацию и развитие системы, когда анализируются положения, параметры и результаты адаптации применяемой версии каждого профиля; выявляются и устраняются её дефекты;
- модернизацию профиля, с учетом появления более совершенных технических и программных средств и новых стандартов ИС; при необходимости осуществляется формирование, документирование и внедрение новой модифицированной и уточненной версии соответствующего профиля.

Разработка и применение профилей являются ***органической частью процессов разработки, сопровождения, модернизации и развития ИС***. Профили характеризуют каждую конкретную систему на всех этапах её жизненного цикла постольку, поскольку они задают гармонизированный набор стандартов, которым должна соответствовать система и её компоненты. Проектированию системы предшествует этап предпроектного обследования 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, остается первоначально на усмотрение разработчика системы, который, руководствуясь требованиями спецификаций, может дополнять и развивать профили, которые впоследствии согласуются с заказчиком.

Таким образом, ***профиль стандартов конкретной системы не является статичным, он развивается и конкретизи-***

руется (возможно, во взаимодействии с заказчиком) в процессе проектирования ИС и оформляется в составе документации проекта системы. 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.

Каждый из выделенных профилей должен для последующего длительного использования пройти стадию формирования, адаптации и параметризации применительно к характеристикам стандартизируемых объектов или процессов создания ИС. ***Подготовка профилей к применению должна учитывать варианты реального состояния проекта системы***:

- планируется создание новой ИС (“с нуля”) в условиях отсутствия задела по системе, компонентам и стандартам данного проекта;
- имеется типовой проект системы и его профили стандартов, и предстоит их адаптация и реализация;
- существует и эксплуатируется реальная – унаследованная ИС, для которой следует подготовить и адаптировать профили с учетом её реального состояния и перспективы развития.

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

- определение целей, которые предполагается достичь применением данного профиля;
- точное перечисление функций продукта или процесса стандартизации, определяемого данным профилем;
- формализованные сценарии применения базовых стандартов и спецификаций, включенных в данный профиль;
- сводку требований к ИС или к её 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 редакций и ограничений, способных оказать

влияние на достижение корректного взаимодействия объектов стандартизации при использовании данного профиля;

- информационные ссылки на спецификации тестов проверки соответствия профилю.

Уровень стандартизации профилей, процессов и объектов их применения отражается не только на технико-экономических показателях системы, но и, что особенно важно, на их качестве. Качество информационных систем тесно связано с методами и технологией их разработки, поэтому важной группой документов в профилях являются стандарты и их рекомендации по непосредственному обеспечению качества ИС.

На этапах жизненного цикла ИС выбираются и затем применяются ***основные функциональные профили*** (см.рис.4.2):

- профиль жизненного цикла информационной системы;
- профиль аппаратной и операционной среды ИС;
- профиль внешней и пользовательской среды функционирования ИС;
- профиль обеспечения безопасности функционирования и защиты информации в ИС;
- профиль инструментальных средств, поддерживающих весь жизненный цикл системы.

Прикладные программные средства является всегда проблемно-ориентированным и определяют основные функции системы. При применении функциональных профилей ИС следует также 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 ИС и профиля инструментальных средств создания, сопровождения и развития программных средств.

Применение функциональных профилей должны поддерживать основные, ***технологические профили*** (см. рис. 4.2):

- 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.

Функциональные профили состоят из профилей компонентов, реализующих функции ИС. Детализация функциональных профилей производится по мере декомпозиции структуры системы на составляющие её компоненты. Выбор и применение функциональных профилей является органической частью процессов проектирования, разработки, сопровождения и развития системы. ***Применение функциональных профилей ИС заключается в выполнении следующих работ***:

- выбор готовых программных и аппаратных средств, соответствующих профилям;
- 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 ИС 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 функциональных профилей и квалификационного тестирования.

Работы, связанные с формированием и применением функциональных и технологических профилей должны быть предусмотрены в составе проекта ИС. Учитывая динамику формирования и применения профилей ПС и БД, по мере детализации структуры системы и ее возможного развития, образуется ***жизненный цикл профилей***. Жизненный цикл профилей ПС можно рассматривать в составе технологических работ отдельно от этапов и работ непосредственной разработки и эксплуатации самих программных средств и баз данных. ***Создание и применение профилей ЖЦ ПС и БД можно разделить на два крупных процесса*** (рис. 4.3):

- разработка, формирование и адаптация профилей ЖЦ ПС для применения в конкретном проекте ИС;
- непосредственное применение требований и рекомендаций каждого адаптированного профиля для регламентирования этапов, работ и документов проекта.

При создании ПС и БД профили развиваются и детализируются параллельно с конкретизацией проекта. Они должны подготавливать 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 ИС и оформление результатов должно опережать, обеспечивать и подготавливать выполнение соответствующих этапов и работ основного ЖЦ программных средств.

Профиль жизненного цикла ПС и БД целесообразно определять на основе подмножества процессов, работ и задач стандарта **ISO 12207** (см. п. 4.1 и рис. 4.1), 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. 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 разработчика проекта ИС или ведомственными нормативными документами.

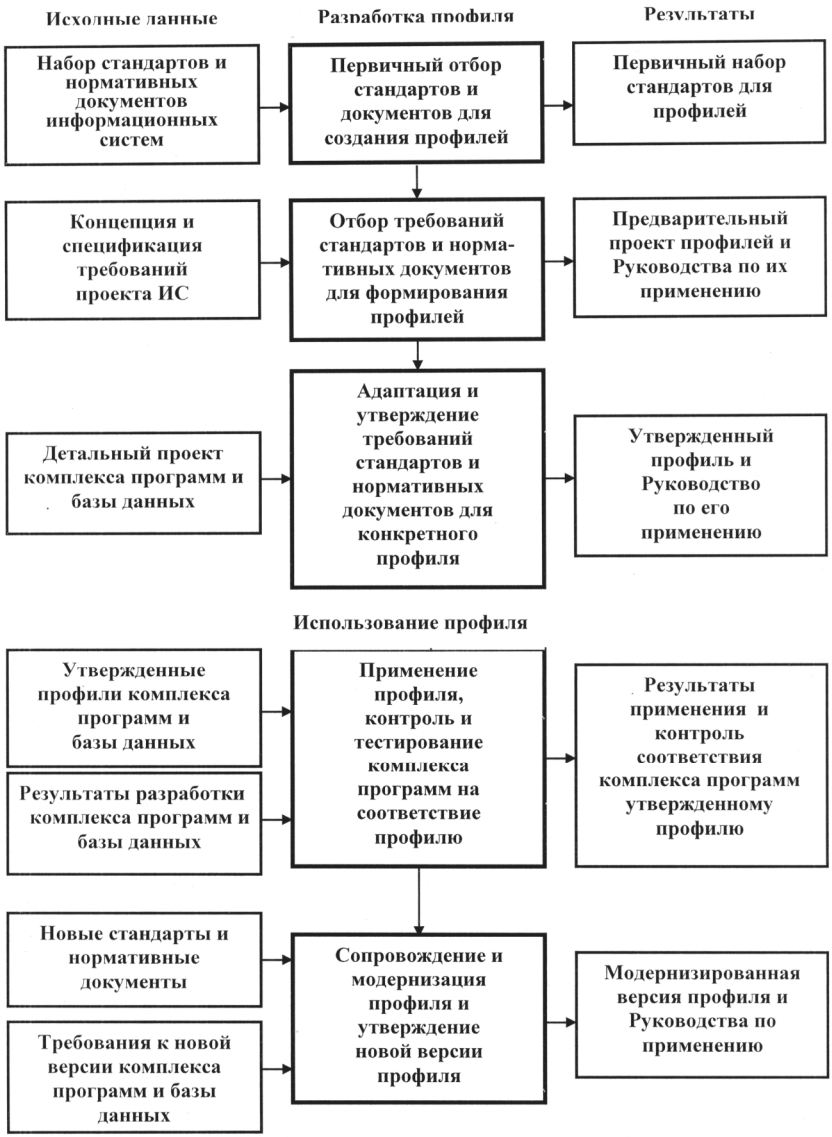

Рис. 4.3

В общем случае созданию профилей сложной информационной 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. 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 жизненного цикла***. Задаются требования к качеству ИС и, соответственно, первичный профиль обеспечения качества комплекса программ и данных, функциональные требования к системе – состав решаемых задач и указываются ссылки на нормативные документы, которые регламентируют правила и процедуры выполнения функций и операций. Этапы разработки профилей, которые определяются разработчиком системы по его усмотрению, должны быть увязаны с этапами жизненного цикла ИС, и выполняться во времени таким образом, чтобы эти разрабатываемые профили могли быть применены тогда, когда это требуется по логике последовательной детализации проекта.

Нормативные документы, регламентирующие жизненный цикл системы и её профилей, либо задаются заказчиком директивно в спецификации требований на создание системы, либо выбираются разработчиком самостоятельно в зависимости от характеристик проекта. При этом надо иметь в виду, итерационный характер формирования и развития профилей конкретной ИС, связанный с итерациями самих процессов разработки, сопровождения и управления конфигурацией системы. Стандарты и нормативные документы, входящие в ***профили жизненного цикла ПС и БД, должны выполнять следующие функции***:

- регламентировать структуру и состав этапов, процессов, работ и документов ЖЦ ПС и БД;
- обеспечивать адаптацию профилей стандартов к характеристикам конкретного проекта или процесса проектирования, внешней и операционной среды;
- формализовать выполнение и документирование конкретных работ при разработке, сопровождении и управлении конфигурацией ПС и БД.

Профиль стандартов ПС и БД (функциональных частей системы) должен определять архитектуру программных комплексов (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). Профиль ПС и БД конкретной системы должен учитывать функциональную ориентацию приложений. Он должен содержать ссылки на стандартизированные интерфейсы между приложениями и внешней средой, которые описываются в профилях среды ИС, защиты информации и инструментальных средств.

Быстро оснащающиеся различными методами и средствами автоматизации этапы системного анализа, моделирования и предварительного проектирования не позволяют стабилизировать основу этих процессов, достаточную для их формализации на уровне международных стандартов. Поэтому для этих этапов могут создаваться и применяться профили ЖЦ ПС и БД как проблемно-ориентирован-ные совокупности нормативных документов и методических руководств, отражающие как наиболее современные методы, так и фрагменты действующих стандартов, в том числе стандартов "де-факто", необходимые для поддержки этапов, работ и объектов жизненного цикла системы определенного класса или функционального назначения.

В стандарте **ISO 12207** изложены ***основы преобразования и адаптации базовой структуры процессов ЖЦ для профиля конкретного проекта ПС и БД***. В нем 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 (см. рис. 4.1).

На этапе ***системного анализа при планировании профиля технологической поддержки разработки ПС и БД*** следует проанализировать набор базовых международных стандартов, связанных с регламентированием особенностей систем и про-

граммных средств (см. рис. 4.3). Для поддержки жизненного цикла разрабатываемых ПС необходимо из них выбрать предварительный набор стандартов, в наибольшей степени относящихся к ПС и БД 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 ИС и комплекса программ.

Одним из преимуществ от разработки и внедрения профиля 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;
- все деловые и технические стратегии, предположения и ограничения проекта;
- 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 ИС;
- уровень компетентности коллектива, разрабатывающего профиль, его знания и пригодность к экспертизе проекта и деятельности предприятия.

На ***этапе проектирования профиля ПС и БД*** уточняется 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 применения с учетом характеристик проекта, методологии и технологии создания ПС,

а также предполагаемых средств автоматизации разработки, сопровождения и управления конфигурацией комплекса программ. Должны быть выбраны, разработаны или приобретены нормативные документы, дополняющие базовые стандарты ЖЦ ПС, с целью полного определения и регламентирования набора стандартов и профилей.

На этом этапе описываются как функциональные, так и технические требования, устанавливаемые в профиле. Полученные результаты могут вызвать изменения в технологии, а также в корпоративной или деловой стратегии. 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 и БД. 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 (см. рис. 4.3). Результатом этого процесса является определение стандартов и выбор интерфейсов, которые удовлетворяют требованиям, предъявляемым к системе в целом. Зачастую, после того как несколько вариантов проекта будут подвергнуты оценке с точки зрения стоимости жизненного цикла или имеющихся рисков, может потребоваться корректировка некоторых требований или проектных решений профилей. 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. Для уверенности в том, что ресурсы будут направлены на решение проблем функциональной совместимо-

сти и переносимости приложений, руководители профилей должны описать эти цели способом, который может быть понят специалистами среднего звена, отвечающими за реализацию технических решений.

Этап разработки ИС связан, прежде всего, с программированием и отладкой компонентов ПС и БД, 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. Аппаратно-программные платформы, на которых выполняются клиентские и серверные части приложений, ***должны соответствовать требованиям профиля среды***.

Как только системные аппаратные и программные средства будут получены от поставщика, их необходимо проверить на соответствие функциональным и эксплуатационным требованиям. 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 и БД 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. Формализация структуры и типового содержания каждого документа должна позволять контролировать соответствие профилю, результаты и качество выполненных работ.

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 и БД***, 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 проверяется соответствие версии ПС и БД данному профилю. Наиболее полная проверка соответствия утвержденному профи-

лю производится в процессе испытаний комплекса программ. В акте по результатам испытаний кроме всех характеристик версии программного продукта должно быть отражено соответствие профилям 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 и БД в процессе создания данной версии комплекса программ (см. рис. 4.3).

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, определяющий стратегию информатизации в целом и позволяющий установить план текущих и долгосрочных мероприятий по развитию информационной инфраструктуры предприятия и внедрению новых информационных систем. Кроме того, в течение времени эксплуатации созданной версии ПС возможно появление новых стандартов де-юре и де-факто, которые целесообразно учесть в данном 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.

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

- правила и процедуры идентификации компонентов и версий профиля;
- 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 ЖЦ ПС и БД.

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. Ряд тестов проверки соответствия применяемых компонентов международным стандартам могут быть использованы готовыми, так как международные стандарты и профили в ряде случаев являются основой при создании международно-признанных аттестационных тестов.

Отдельные компоненты профиля подлежат тестированию, как с точки зрения соответствия необходимым стандартам, так и соответствия требованиям, сформулированным в терминах их характеристик качества. Однако тестирование должно быть ограничено наблюдаемым поведением, которое может быть определено в базовой спецификации. 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. Если эталонная реализация невыполнима, можно использовать другие методы тестирования, такие как тестирование компонентов программного средства, квалификационное тестирование ПС и совместное тестирование аппаратных и программных средств.

На ***этапе внедрения профиля*** важно иметь план по его

применению. Руководители высшего уровня должны установить приоритеты при реализации отдельных частей и требований профиля. Если исполнители будут ориентированы на внедрение профиля в целом без четкого выделения приоритетов и четких временных рамок внедрения различных частей профиля, весьма вероятно, что работы будут малоэффективны. 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 целей управления проектом.

В ряде случаев, производится ***перенос разработанного программного продукта*** с инструментальной платформы разработчика системы на реальную платформу применения ИС. При этом проверяется соответствие реальной платформы требованиям функциональных профилей ИС и функционирование ПС и БД на реальной платформе. Этап внедрения предполагает адаптацию и настройку программного продукта на реальные условия эксплуатации, для которых он создавался. Приемочные испытания ИС должны проводиться в условиях реальной эксплуатации на соответствие спецификациям функциональных требований и требованиям полного профиля ИС, 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. Коллективный подход сокращает субъективное начало при выборе технологии в процессе разработки профилей. Коллективность также гарантирует, что в процессе разработки за основу будут приняты цели и эксплуатационные требования предприятий, которые будут использовать профиль. Можно выделить следующие ***основные группы специалистов, использующие документы профилей*** (см. п. 3.2):

- руководители крупного проекта системы и её основных, функциональных компонентов программного продукта;

- 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 и БД 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 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 и БД [29]. В них должно быть отражено:

- содержание и описание выбранных положений стандартов и нормативных документов профиля с позиции его конкретного пользователя;
- параметры адаптации стандартов профиля и содержание дополнительных нормативных документов;
- методика и сценарии корректного применения всех обязательных и рекомендуемых положений профиля;
- требования к содержанию отчетов о результатах контроля и тестирования компонентов системы на соответствие обязательным положениям профиля в процессе их жизненного цикла.

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.

Общий объем испытаний при сертификации системы и ее компонентов может быть значительно шире, чем проверка только на соответствие профилям.

4.3. Задачи и направления развития концепции

открытых систем

Многие организации понимают необходимость перехода к концепции и стандартам открытых систем, которые нужны ***для обеспечения эффективного сопровождения, функциональной совместимости и переносимости программных средств и информации баз данных*** между системами и платформами. Применение концепции и технологии открытых систем в практической деятельности предприятий обеспечивает независимость от поставщика, гибкость в использовании коммуникаций, технических и программных средств ЭВМ, уменьшает время переобучения персонала и позволяет найти решения ряда сложных проблем проектов ПС и БД. В последнее время технология сопровождения и создания информационных систем путем переноса программных средств и баз данных на другие аппаратные и операционные платформы претерпела качественное изменение и активно развивается на основе концепции и стандартов открытых систем. Цели и понятия открытых систем в имеющихся публикациях [35, 36, 43, 60] формулируются весьма разнообразно. Опубликовано несколько определений этого термина, отражающих его основные аспекты и интересы авторов. Выявлен ряд понятий, методов и функций, которые могут рассматриваться как достаточно полная база и набор компонентов для определения содержания и создания открытых систем. ***Основная цель*** применения методов и стандартов открытых систем состоит в ***повышении эффективности сопровождения, модификации, расширения и переноса сложных программных средств и информации баз данных***.

Понятие открытых систем основывается на выделении в составе информационной системы составных частей или структурных компонентов, каждый из которых объединяет в своем составе совокупность объектов, имеющих близкую физическую природу или родственных между собой по какому-либо признаку. Концепция открытых систем основывается на выделении: среды прикладной платформы, среды прикладных программных средств и внешней среды. Такое выделение облегчается, если указать все интерфейсы между данными средами, включая поддерживаемые ими функции взаимодействия и форматы данных. Для создания среды открытых систем необходимо в первую

очередь заботиться о стандартизации интерфейсов "платформа – приложение" и "платформа – внешняя среда", а также форматов данных.

Открытая система – это исчерпывающая и последовательная совокупность стандартов в области информационных технологий, которая специфицирует интерфейсы, функции и поддерживающие форматы интерфейсов для достижения эффективного взаимодействия и переносимости ПС, данных и персонала пользователей. В результате открытые системы должны обеспечивать:

- сохранение инвестиций, вложенных в реализованные информационные системы, их программные средства и базы данных, в процессе развития архитектуры и ресурсов аппаратных и операционных платформ;
- снижение трудоемкости, стоимости и длительности разработки и сопровождения сложных распределенных информационных систем, программных средств и баз данных;
- возможность эффективного переноса (мобильность) прикладных ПС и БД, разработанных должным образом, с минимальными изменениями на широкий диапазон систем;
- совместную работу (интероперабельность) с другими прикладными ПС и БД на локальных и удаленных платформах;
- взаимодействие с пользователями в стиле, облегчающем им удобный переход от системы к системе (мобильность пользователей).

Основными целями создания и применения концепции, методов и стандартов открытых систем является повышение общей экономической эффективности разработки, сопровождения и функционирования информационных систем, а также логической и технической совместимости их компонентов. Для достижения этих целей развиваются и применяются различные проблемно-ориентирован-ные технологии и комплексы средств автоматизации ЖЦ комплексов программ и баз данных, базирующиеся на повторном использовании апробированных программных компонентов и данных, их эффективном сопровождении и переносе на различные аппаратные и операционные платформы, а также на согласованном взаимодействии в сложных, распределенных информационных системах.

Открытые системы делают доступным пользователям широкий круг готовых прикладных ПС и БД и их компонентов,

позволяют фирмам сэкономить средства на документации и подготовке персонала, а, кроме того, делают их независимыми от конкретного поставщика программного и аппаратного продукта. При этом каждое решение при создании информационной системы должно обеспечивать интероперабельность, переносимость, межплатформенную интеграцию и эффективное использование ресурсов распределенной обработки информации. Построение открытых систем на унифицированных по взаимодействию продуктах стимулирует конкуренцию среди поставщиков как по соотношению цена/производительность, так и по функциональным возможностям ПС и БД. Профессионалы в области открытых систем акцентируют усилия на поиске и создании гибкой, способной к расширению масштаба среды, что базируется на ***трех основных направлениях стандартизации*** в области информационных систем:

- стандартизация аппаратных и операционных платформ;
- стандартизация методов и технологии обеспечения жизненного цикла прикладных программных средств и баз данных;
- стандартизация интерфейсов приложений между собой, с операционной и внешней средой.

Развивающаяся инфраструктура открытых систем базируется на интеграции компонентов для удовлетворения требований сегодняшнего дня, с использованием стандартов на интерфейсы приложений с операционной и внешней средой ***для последующего широкого применения и сопровождения ИС***. В этих условиях особое значение приобретают тестирование компонентов, программных продуктов и сложных информационных систем, гарантирующие их высокое качество и безопасность функционирования. Следует учитывать, что весь процесс создания и развития конкретных открытых систем реализуется путем преодоления постоянных противоречий между требованиями сегодняшнего дня и потенциальными перспективами сопровождения и развития определенного класса ИС по производительности, гибкости и внедрению инноваций.

Для последующего анализа методов и стандартов открытых систем целесообразно представить основные компоненты архитектуры программных средств, участвующих в процессе обработки информации, и особенности их унификации. В современных ЭВМ функционирование прикладных ПС и БД организует-

ся операционными системами (ОС) и рядом типовых системных программ. Структурирование операционных систем состоит, прежде всего, в ***выделение ядра*** – минимальной, функционально полной ее части, служащей основой для модульных и переносимых расширений – рис. 4.4 [22, 43]. Такое минимальное ядро поддерживает только планирование и диспетчеризацию процессов, обработку прерываний и сетевые службы верхнего уровня. Внешними по отношению к ядру модулями становятся файловые системы, сетевые серверы, оконные графические системы. Схема обмена сообщениями между ядрами является удобной основой для организации удаленных вызовов процедур, поскольку ядру безразлично, откуда исходит сообщение.

Ядро компактно изолирует всю машинно-зависимую часть ОС и при переносе на новый процессор все изменения логически сгруппированы. Тем самым разработка нового ядра для новой платформы является относительно небольшой частью локализованных не автоматизируемых работ. Такая организация обеспечивает возможности переноса операционной системы и расширения, опирающиеся на ограниченный набор интерфейсов ядра. Такой набор для ядра может не превышать 20 интерфейсов, что сокращает вероятность ошибок при их использовании системными программистами. Однако для прикладных комплексов программ количество видов интерфейсов на порядок больше, но они остаются вне ядра и не обязательно или только частично являются машинно-зависимыми.

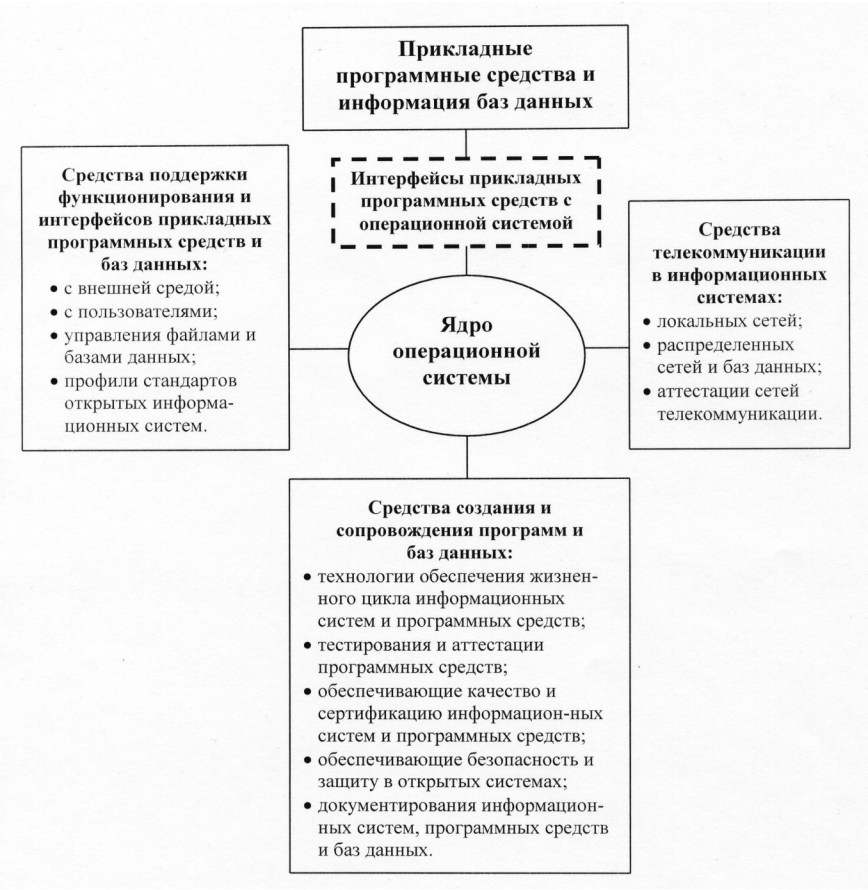

Рис. 4.4

Первоначально концепция открытых систем развивалась при непосредственной поддержке разработчиками операционной системы UNIX с широким набором функций и её модификациями. При последующем изложении предполагается, что ядро операционной системы позволяет использовать стандартизированные интерфейсы с приложениями, а расширения операционной системы выполняют самостоятельные функции и организационно в нее не входят. Таким образом, в понятие операционной системы далее включается ее ядро и некоторые расширения,

непосредственно поддерживающие управление вычислительным процессом и памятью [44, 61, 62].

В составе коммерческих операционных систем или, как достаточно автономные компоненты, обеспечивающие решение сложных прикладных задач, используются следующие крупные ***группы технологических программных средств*** (см. рис. 4.4):

- поддерживающие функционирование и оперативное взаимодействие прикладных программ и баз данных с операционной и внешней средой;
- обеспечивающие телекоммуникацию и обмен данными между территориально распределенными компонентами сложных информационных систем;
- обеспечивающие автоматизацию разработки, сопровождения и всего жизненного цикла мобильных программных средств и баз данных.

Каждое из перечисленных средств содержит широкий набор компонентов, многие из которых можно и целесообразно унифицировать с учетом особенностей проблемно-ориентированных сфер применения ИС. Так как функциональные задачи, реализуемые с использованием прикладных программ и баз данных, непрерывно усложняются и увеличиваются количественно, то унификация их внутреннего содержания весьма ограничена. В то же время необходимо снижать трудоемкость и длительность создания программных продуктов. Это осуществляется путем автоматизации технологии создания ПС и БД, а также стандартизацией окружения операционных систем. Существенное влияние на мобильность программ и данных в сложных ИС оказывает унификация интерфейсов их компонентов. Это обеспечивается стандартизированными средствами взаимодействия программных компонентов между собой, с операционной и внешней средой, как в однородных, так и в гетерогенных аппаратных платформах.

Средства, поддерживающие функционирование и оперативное взаимодействие комплексов программ и баз данных с внешней средой и пользователями, доступны для глубокой стандартизации их концепций, методов и средств реализации базовых функций. Приведенные на рис. 4.4 компоненты, присущи практически любым информационным системам, могут быть локализованы и в значительной степени унифицированы. В этой области достигнуты значительные успехи, которые обес-

печивают унификацию структуры и интерфейсов приложений в распределенных информационных системах. Сохраняется возможность унификации и совершенствования аппаратных средств и логики диалогового общения пользователей с ИС. Разумная глубина стандартизации позволяет консервировать средства непосредственного взаимодействия приложений с операционными системами и сохраняет возможность развития процессов и средств комфортного взаимодействия с пользователями, управления заданиями, а также управления файлами и базами данных.

Средства, обеспечивающие автоматизацию разработки, сопровождения и всего жизненного цикла ПС и БД в наибольшей степени допускают унификацию в части языков программирования, а также структуры и состава документов, сопровождающих программы и описания данных. Поэтому исторически наиболее активные и успешные работы по стандартизации еще в 60-е годы сосредоточились на языках программирования и правилах оформления документации. Наименее унифицируемыми в этой группе являются современные CASE-системы, особенно в тех компонентах, которые поддерживают системный анализ и структурное проектирование ПС и БД, что определяет практическое отсутствие стандартов в этой области. Тестирование, испытания и сертификация, а также управление конфигурацией ПС и БД частично стандартизированы методологически и как определенные технологические процессы. Унификация средств автоматизации этих процессов пока вряд ли целесообразна, вследствие широкого спектра характеристик у приложений и технологий их разработки. Средства защиты информации и обеспечения безопасности применения ИС допускают глубокую унификацию концепций и методов защиты, в то же время, стимулируется развитие и совершенствование конкретных средств в соответствии с требованиями к сохранности информации и её конфиденциальности в определенных ПС и БД.

Средства обеспечения телекоммуникации и транспортировки данных имеют большое значение для обеспечения согласованного взаимодействия компонентов в распределенных информационных системах. В этой области успешно проводится глубокая стандартизация не только концепций и методов взаи-

модействия компонентов, но также конкретных протоколов и профилей локальных и распределенных сетей и баз данных. Для гарантий качества реализации интерфейсов и контроля их соответствия стандартам в этой группе средств большое значение придается стандартам аттестации процессов и средств, осуществляющих непосредственную телекоммуникацию.

Идеология открытых систем базируется на соблюдении совокупности протоколов и стандартов де-юре и де-факто. Программные и аппаратные компоненты ИС по этой идеологии должны отвечать ***двум важнейшим требованиям***: переносимости и возможности согласованной, совместной работы с другими удаленными компонентами. Это позволяет обеспечить совместимость различных ИС и средств передачи данных. Компоненты являются открытыми друг для друга в том случае, если, несмотря на их программно-аппаратные различия, они могут взаимодействовать с помощью определенных процедур "***прозрачных***" для пользователя. Задача сводится к максимально возможному повторному использованию разработанных и апробированных программных и информационных компонентов при изменении вычислительных аппаратных платформ, их операционных систем и процессов взаимодействия. В связи с этим сформировался принципиально новый подход к созданию и эволюции распределенных, интегрированных программно-аппаратных систем. Для реализации перечисленных этих целей с начала 80-х годов активно развиваются ***два направления идеологии, концепции и системы стандартов открытых систем*** [22, 36, 60, 62]:

- ***открытых вычислительных информационных*** систем (open computing systems – OCS), обеспечивающих возможность относительно простого и эффективного по трудоемкости переноса апробированных программных средств и информации баз данных на различные типы аппаратных платформ за счет стандартизации процессов и интерфейсов взаимодействия с операционными системами ЭВМ;
- ***взаимосвязи открытых систем*** (open systems interconnection – OSI), унифицирующие структуру, процессы и интерфейсы для обеспечения совместимости методов и средств оперативного обмена данными между разнотипными удаленными ЭВМ, а также поддерживающие возможность предваритель-

ного выбора типов и ресурсов ЭВМ в соответствии с потребностями ИС для решения конкретных прикладных задач.

В ***первом направлении*** основной задачей является транспортировка, модификация и перенос функций и процедур программ обработки информации, а также содержания баз данных между различными платформами, осуществляющими ее обработку. Подобные обмены функциями, процедурами и данными имеют неоперативный характер и могут осуществляться вне реального масштаба времени. Проблемы состоят, преимущественно, в обеспечении целостности и сохранности апробированного функционального ядра текстов программных компонентов и информации баз данных при их изменениях и переносе на иные аппаратные и операционные платформы для снижения трудоемкости создания подобных ИС.

Во ***втором направлении*** основную роль играет оперативная транспортировка данных между компонентами распределенных ИС в реальном масштабе времени. Проблема заключается в обеспечении совместимости различных систем передачи данных и эффективном использовании распределенных вычислительных ресурсов для обработки информации. Основной экономический эффект в этом случае достигается за счет сокращения дополнительных преобразований данных на стыках коммуникационных средств и повышения тем самым степени полезного использования вычислительных и коммуникационных ресурсов.

Важнейшими, объединяющими ***целями развития обоих направлений открытых систем*** являются снижение трудоемкости и длительности создания, а также повышение качества и функциональных возможностей современных ИС. В этих двух направлениях развиваются соответствующие концепции и методы, которые формализуются и детализируются комплексами стандартов. Для каждого направление характерно развитие методов и стандартов, ориентированных преимущественно на его поддержку и реализацию, а также некоторой общей части для обоих направлений. Таким образом, выявились достаточно, автономные части каждого направления и объединяющая их часть методов и стандартов открытых систем.

Разработано свыше четырехсот международных стандартов, в той или иной степени отражающих концепции и методы от-

крытых систем. Они создавались группами специалистов разных подкомитетов **ISO** и попали в разные категории, которые не всегда адекватно отражают их сущность и функциональное назначение. Поэтому рядом специалистов созданы модели, призванные сгруппировать методы и стандарты открытых систем и представить их взаимодействие в обобщенной и наглядной форме. Однако обычно внимание акцентируется либо на стандартах, поддерживающих мобильность прикладных комплексов программ, либо на стандартах, обеспечивающих коммуникацию данных.

Наиболее известной является ***семиуровневая базовая эталонная модель взаимосвязи открытых систем*** (BOC - OSI) [10, 30, 60], которая ориентирована, прежде всего, на регламентирование телекоммуникации в ИС (второе направление). Модель описывает процессы взаимодействия сообщениями и данными между прикладными информационными системами в вычислительных сетях. Она является наиболее проработанной с функциональной точки зрения, полноты набора стандартов и определения их совместимости друг с другом. Модель основана на разбиении среды на семь уровней, взаимодействие между которыми описывается соответствующими стандартами. Это обеспечивает практически полную "прозрачность" взаимодействия через эти уровни вне зависимости от того, каким образом построен любой из уровней в каждой конкретной реализации. Моделью задается открытая коммуникационная среда, полностью независимая от того, как и на какой аппаратной и программной основе реализован каждый уровень. Эта модель относится к области коммуникационных взаимодействий и не рассматривает взаимодействия составных компонентов процессов в отдельной вычислительной машине, на основе анализа которых возможно обеспечение мобильности прикладных программ и информации баз данных.

Рабочей группой POSIX P1003.0 (см. п. 4.4) Института инженеров по электронике и электротехнике США – IEEE предложена Референсная Модель Среды Открытых Систем (OSE/RM) – ***Эталонная модель функциональной среды открытых систем*** [44, 60]. Данная модель предусматривает разбиение среды на три составных части: прикладные программные средства; прикладная платформа; внешняя среда. Под первыми понимаются собственно комплексы программ, информация баз

данных, а также документация и средства обучения пользователей. Прикладная платформа состоит из аппаратной платформы и системного программного обеспечения. Сюда входят: операционная система, компиляторы, СУБД, графические системы, составляющие операционную среду для прикладных программных средств и баз данных. К внешней среде относятся все системные компоненты, которые являются внешними по отношению к прикладной платформе и прикладным ПС и БД. Это утилиты и подсистемы, реализуемые на других (удаленных) платформах, а также периферийные устройства.

В данной книге изложение базируется на модели, объединяющей два, приведенные выше направления развития методов и стандартов открытых систем [22] – рис. 4.5. Ее левая часть соответствует направлению и концепции Открытых вычислительных систем – OCS, а правая – Открытых систем коммуникации – OSI. Их объединяет группа методов и стандартов совместимости с операционной и внешней средой, которая является общей для обоих направлений. Для каждого из них выделена концептуальная часть, отражающая наиболее существенную специфику каждого направления открытых систем, и часть методов и стандартов, детализирующих реализацию конкретных процедур соответствующего направления.

Цель первого направления – создание методов и стандартов, обеспечивающих эффективное по трудоемкости и качеству ***сопровождение и перенос вне реального времени*** программных средств обработки информации и баз данных на различные аппаратные и операционные платформы. Эти методы можно разделить на ***три части*** (см. рис. 4.5):

- общая концепция, методы и стандарты непосредственного обеспечения мобильности прикладных программных средств и баз данных в процессе разработки информационных систем за счет унификации интерфейсов с операционной системой;
- методы и стандарты, поддерживающие мобильность прикладных программ и данных в распределенных информационных системах и совместимость их взаимодействия с внешней средой и с пользователями;
- методы и стандарты, обеспечивающие создание текстов программных средств, баз данных и их компонентов на стандартизированных языках программирования высокого уровня, а

также сопровождение, мобильность и потенциальную возможность их переноса на различные аппаратные и операционные платформы.

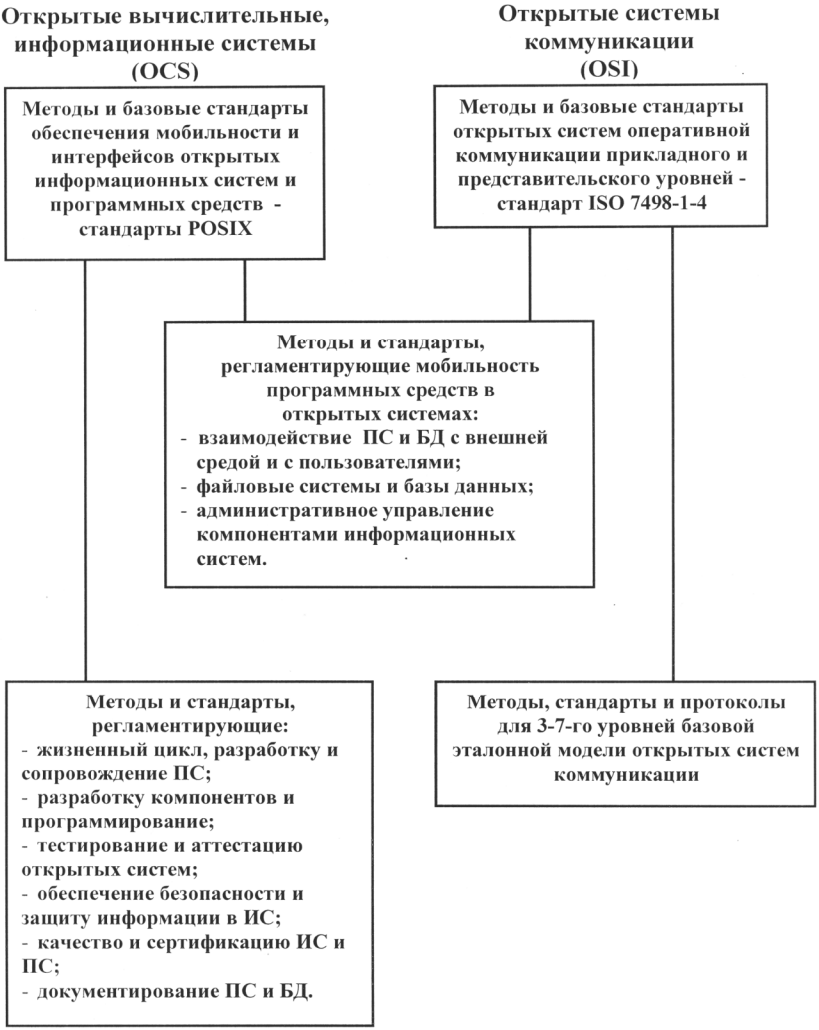

Рис. 4.5

Первая часть методов и стандартов создавалась с ограниченной и определенной целью локализовать и унифицировать

интерфейсы компонентов и прикладных комплексов программ между собой, с заранее выделенными операционными системами и внешней средой. Интерфейсные стандарты являются базой для обеспечения расширения и свободного перемещения ПС и БД в различные по архитектуре и функциям операционные окружения и аппаратные платформы. Эти методы позволяют разделить функциональную часть программных компонентов 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 информационных системах и обеспечивают свободу разработчикам при выборе технологии и методов проектирования и языков программирования для создания комплексов программ и информации баз данных. Эта часть стандартов представлены достаточно подробно ниже при описании архитектуры и содержания стандартов POSIX (см. п. 4.4), в которой определены концепция и функции интерфейсов переносимых операционных систем, команды управления и сервисные программы, а также расширение для переносимых операционных систем реального времени.

Вторая часть методов и стандартов имеет целью поддержать мобильность приложений в открытых информационных системах путем унификации их интерфейсов с внешней средой. Они обеспечивают унификацию и совместимость обмена данными в различных файловых системах и базах данных, унификацию административного управления и взаимодействия с пользователями функциональных приложений, а также методов защиты информации. Таким образом, создается стандартизированная по интерфейсам внешняя среда на различных гетерогенных аппаратных и операционных платформах, в которую могут эффективно погружаться различные приложения, согласованные по интерфейсам с этими стандартами. Стандарты 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 файловых систем и баз данных, обеспечивающих

их совместимость и унификацию в сложных информационных системах;

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

Эта часть стандартов, как и поддерживаемые ими методы, относятся к обоим направлениям стандартов открытых систем. Они первоначально разрабатывались, в основном, как компоненты эталонной модели ВОС и использованы при формировании стандартов POSIX. Поэтому при общей характеристике стандартов второго направления они обозначены на рис. 4.5, как связующее звено между стандартами POSIX и OSI.

Третья часть методов и стандартов создавалась в значительной степени независимо от первой и второй и на много раньше. Методы преследуют цель унифицировать тексты программ и описания данных, создаваемых для различных аппаратных платформ при любой их архитектуре, независимо от операционной и внешней среды. Первоначально большое число языков программирования третьего поколения (3GL) (свыше 200), каждый из которых имел несколько диалектов, в результате сократилось до 4 – 6 языков, ограниченных стандартами, не допускающих диалектов. Создание современных ПС и БД поддерживается методами и средствами тестирования и аттестации программ и их интерфейсов, а также унифицированным составом и содержанием документации на прикладные программы и базы данных (см. п. 2.3). Эти методы и средства обеспечивают необходимое качество сложных ПС и БД и его сохранение при их модификации и переносе. В результате обеспечена ***мобильность функциональной части программ***, однако они могут требовать доработок для сопряжения с новой средой аппаратного и операционного окружения информационных систем.

Представление этих стандартов совместно с методами и стандартами открытых систем является необходимым расширением для более полного отражения процессов и методов сопровождения и переноса программ и данных (см. рис. 4.1). Процесс поддерживается широким набором стандартов, основная часть которых создавалась независимо от концепции открытых систем. Основные стандарты этой группы регламентируют:

- жизненный цикл, разработку и сопровождение современных ПС и БД;

- разработку компонентов на языках программирования;
- тестирование и аттестацию программных средств;
- обеспечение безопасности и защиту информации ИС;
- качество и сертификацию ИС и ПС;
- документирование программных средств и баз данных.

Цель второго направления – обеспечение совместимости оперативного обмена данными между информационными системами, базирующимися на гетерогенных и распределенных аппаратных платформах и включает свыше ста стандартов открытых систем коммуникации. Для этого потребовалось создание концепции, методов и стандартов унификации транспортировки данных между компонентами сложных информационных систем. Эти методы и стандарты, также как и для первого направления можно разделить на ***три части*** (см. рис. 4.5):

- общая концепция и методы организации совместимой коммуникации данных между компонентами сложных информационных систем, размещенных на различных аппаратных и операционных платформах и группа базовых стандартов, поддерживающих, конкретизирующих и развивающих семиуровневую эталонную модель взаимосвязи открытых систем (ВОС), а также ее компоненты и общие функции на прикладном и, частично, представительском уровнях;
- методы обеспечения непосредственной совместимости интерфейса данных между компонентами информационных систем и группа стандартов средних и нижних (3 – 7) уровней эталонной модели взаимосвязи открытых систем, унифицирующих и непосредственно реализующих конкретные функции коммуникации данных между компонентами локальных и распределенных информационных систем;
- методы и стандарты поддержки локальных функций коммуникации и совместимости при взаимодействии приложений с внешней средой и обеспечение их унификации, которая представлена выше в первом направлении открытых систем.

Первая часть методов и стандартов создавалась как концептуальная база взаимосвязи открытых коммуникационных систем. Цель состояла в создании системных основ и общей архитектуры телекоммуникационных интерфейсов и протоколов локальных и территориально распределенных информационных систем. Эта цель реализована в совокупности стандартов – ***Ба-***

зовой эталонной модели взаимосвязи открытых систем, а также, в группе стандартов, поддерживающих и развивающих семиуровневую Базовую модель, на ее верхних уровнях.

Вторая часть методов регламентирует процессы, состав и протоколы обмена информацией на средних и нижних уровнях Базовой эталонной модели взаимосвязи открытых систем. Это обеспечивает унификацию обмена информацией и конкретизацию взаимодействия в сложных системах телекоммуникации с последовательной детализацией логики функционирования и процедур обмена данными между компонентами систем. Особое внимание уделяется методам обеспечения, проверки корректности и аттестации реализации протоколов взаимосвязи систем, а также достоверности транспортировки информации. На мобильность программ и данных эти методы влияют только в той их части, которая непосредственно отражает процессы коммуникации.

Первой и второй группам стандартов посвящалось традиционно основное внимание в публикациях по открытым системам [35, 43, 60, 61]. В них можно найти изложение стандартов эталонной модели ВОС и ее компонентов, поддерживающих коммуникацию. Поэтому далее основное внимание сосредоточено на тех стандартах ВОС, которые регламентируют взаимодействие с внешней средой, поддерживают сопровождение и переносимость программ и данных.

Третья часть методов в концепции взаимосвязи открытых систем обеспечивает унификацию основных системных функций и интерфейсов с операционной и внешней средой, а также качество их функционирования. Эти методы используются в комплексной стандартизации коммуникации между сервисными и прикладными компонентами информационных систем и также непосредственно влияют на переносимость прикладных программ и данных. Унификация интерфейсов наиболее общих компонентов внешней среды: взаимодействия с пользователями; административного управления, организации хранения и обмена данными и их защита, одновременно влияет как на оперативную транспортировку данных между компонентами информационных систем, так и на возможность переноса приложений, с которыми взаимодействуют эти компоненты. Таким образом, эта часть методов является общей как для концепции открытых

вычислительных систем, так и для открытых систем коммуникации.

Следует подчеркнуть, что выделенные направления и группы методов и стандартов не всегда имеют четкие границы с другими группами. Это, в частности, отражается на небольших дублированиях и на взаимных ссылках при их аннотировании. В подлинных текстах стандартов обычно содержатся ссылки на наиболее важные положения других международных стандартов. Реальные функциональные связи между стандартами открытых систем особенно сильно проявляются при формировании профилей для проблемно-ориентированных информационных систем. Несколько особняком от приведенных направлений и групп стоят ***методы и стандарты формирования профилей*** (см. п. 4.2), а также, поддерживающие сертификацию приложений в открытых системах. Эти методы и стандарты выполняют некоторые обобщающие функции при реализации систем. В той или иной степени они используют выборки требований из базовых стандартов открытых систем и на этой основе могут оформляться в самостоятельные стандарты.

В профилях сосредоточиваются наборы базовых стандартов из перечисленных ранее групп, предназначенные для регламентирования конкретных прикладных функций открытых систем. При этом могут использоваться не все положения, требования и параметры отобранных для профиля базовых стандартов, а также могут дополнительно применяться некоторые стандарты де-факто и формализованные спецификации, не отраженные в международных стандартах. При сертификации приложений также могут использоваться, применявшиеся базовые стандарты, профили и документы, в соответствии с назначением и функциями прикладной открытой системы. Эти две группы методов и стандартов, также как и предшествующие, направлены на ***обеспечение качества и переносимости прикладных программ и информации баз данных***, а также на повышение технико-экономических показателей при создании сложных ИС.

Следует подчеркнуть, что применение стандартов открытых систем должно начинаться ***при создании исходных, мобильных ПС и БД***, а далее неукоснительно использоваться при всех процессах сопровождения и переноса на различные платформы. При переносе унаследованных ПС и БД, созданных вне

концепции открытых систем, может оказаться рентабельным одновременно с переходом на новую технологию ввести применение стандартов открытых систем, что обеспечит возможность удобного, последующего функционального расширения перенесенной системы. Во многих случаях создание современных сложных ИС целесообразно вести с использованием совокупности международных стандартов, значительная часть которых обеспечивает сопровождение и мобильность программных средств и баз данных.

4.4. Стандарты, регламентирующие структуру и интерфейсы программных средств с операционной и внешней средой

Существующие международные стандарты неравномерно поддерживают жизненный цикл ПС и БД, а также не всегда ориентированы на создание мобильных программ и данных. В данном разделе собраны и аннотированы стандарты, которые непосредственно и в наибольшей степени могут быть полезны при создании и сопровождении мобильных ПС и БД. Эти стандарты можно разделить на две крупные группы. К первой группе относятся стандарты, регламентирующие объекты – фрагменты мобильных прикладных ПС и БД. Вторую группу составляют стандарты, поддерживающие процессы жизненного цикла программ и данных. При этом делается акцент на создание потенциально мобильных программ и данных, в предположении, что процессы переноса таких ПС и БД являются более простыми и почти не требуют дополнительной стандартизации. При ***первичной разработке мобильных ПС целесообразно использовать совокупность стандартов***, регламентирующих и поддерживающих современные методы, средства автоматизации и технологические процессы создания сложных прикладных программ и баз данных. К ним также относятся стандарты, которые существенно влияют на качество ПС и БД (см. Приложение и рис. 4.1):

Опыт создания мобильных программных средств привел к необходимости разработки концепции и комплекса стандартов, обеспечивающих ***эффективную по трудоемкости переносимость прикладных программ*** между различными аппаратными и операционными платформами. Ядром стала группа стандартов, созданная специалистами США под эгидой IEEE под об-

щим названием – Интерфейсы переносимых операционных систем (Portable operating system interfaces – POSIX). Проблему переноса программ сосредоточили на унификации интерфейсов операционных систем ЭВМ с различными прикладными программами, а также с окружающей средой: с пользователями, базами данных, средствами коммуникации. Эти стандарты не ориентированы на определенную, конкретную архитектуру ЭВМ, однако предполагают использование современной операционной среды и прежде всего UNIX, как стандарта де-факто, а также международных стандартов на языки программирования и стандартов верхних уровней взаимосвязи открытых систем (ВОС – **ISO 7498**) [10, 30, 43, 60]. В совокупности они образуют нормативную базу открытых компьютерных систем – OCS, обеспечивающих разработку переносимых программных средств и баз данных.

При формировании ***концепции стандартов POSIX были поставлены следующие задачи*** [60, 62]:

- содействовать облегчению переноса кода прикладных программ на иные платформы;
- способствовать определению и унификации интерфейсов заранее при проектировании, а не только в процессе их реализации;
- сохранить по возможности и учитывать все главные, созданные ранее, унаследованные и используемые прикладные программы;
- определять необходимый минимум интерфейсов прикладных программ, для ускорения создания, одобрения и утверждения документов;
- развивать стандарты в направлении обеспечения коммуникационных сетей, распределенной обработки данных и защиты информации;
- рекомендовать ограничивать использование объектного кода для приложений в простых системах.

Все стандарты POSIX имеют рекомендательный характер. Они не должны служить препятствием для переноса объектного кода, ограничивать или ухудшать исполнение приложений при стандартизированных интерфейсах и ограничивать формирование новых унифицированных интерфейсов по мере необходи-

мости. По функциональному назначению ***документы POSIX можно разделить на четыре группы стандартов*** – рис. 4.6:

- ***базовые стандарты***, определяющие общие принципы построения, директивы, основы реализации и методологию тестирования интерфейсов мобильных приложений, а также общее административное управление программами и данными:
  - **IEEE 1003.0** – Руководство по POSIX окружению открытых систем. Набор POSIX стандартов;
  - **ISO 9945-1:1990 (IEEE 1003.1)** – Информационная технология. Интерфейсы переносимых операционных систем. Часть 1: Интерфейсы систем прикладных программ (язык Си);
  - **ISO 9945-2:1992 (IEEE 1003.2)** – Информационная технология. Интерфейсы переносимых операционных систем. Часть 2. Команды управления и сервисные программы (Shell and utilities);
  - **IEEE 1003.3** – Методы тестирования для определения соответствия стандартам POSIX;
  - **IEEE 1003.4** – Расширение к IEEE 1003.1 для переносимых операционных систем и прикладных программ реального времени;
  - **IEEE 1003.7** – Расширения, связанные с системным администрированием (управлением);
- ***конкретизирующие интерфейсы*** с операционной платформой прикладных программ, разрабатываемых на трех базовых языках программирования – Си, Фортран и Ада:
  - **IEEE 1003.5** – Интерфейсы переносимых приложений на языке Ада с привязкой к системным средствам POSIX 1003.1;
  - **IEEE 1003.9** – Интерфейсы переносимых приложений на языке Фортран;
  - **IEEE 1003.16** – Развитие интерфейсов для переносимых приложений на языке Си;
  - **IEEE 1003.19** – Развитие интерфейсов для переносимых приложений на языке Фортран;
  - **IEEE 1003.20** – Развитие интерфейсов для переносимых приложений на языке Ада;

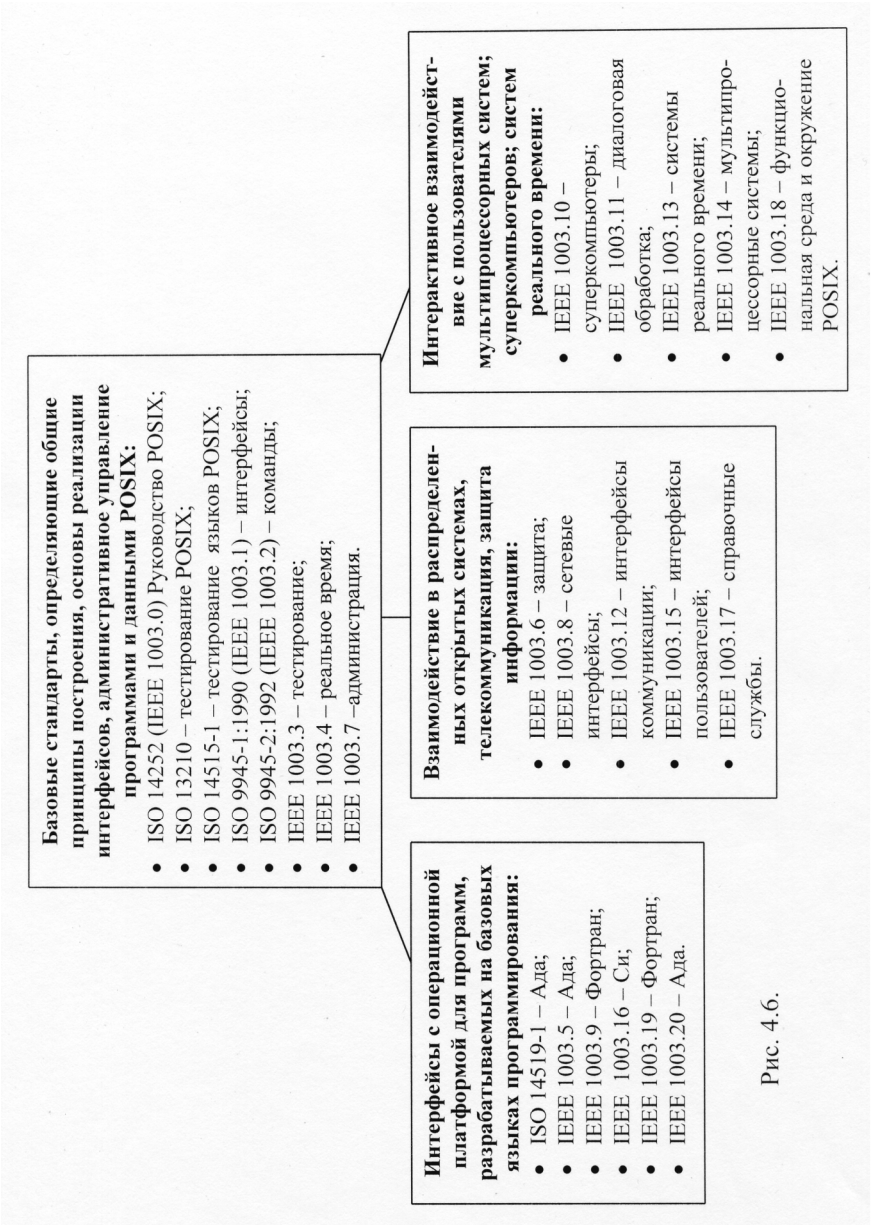

Рис. 4.6.

- ***определяющие взаимодействие*** в распределенных открытых системах, телекоммуникацию в сетях, а также защиту информации:

* **IEEE 1003.6** – Расширения для обеспечения защиты информации и безопасности переносимых прикладных программ;
* **IEEE 1003.8** – Сети передачи данных и сетевые интерфейсы для прозрачного доступа к файлам;
* **IEEE 1003.12** – Протокольно независимые интерфейсы для коммуникации;
* **IEEE 1003.15** – Расширение для очередей пакетов прикладных программ, определяющие интерфейсы пользователей и администраторов при пакетной обработке;
* **IEEE 1003.17** – Справочные службы, обслуживание каталогов (директорий) и пространство имен;

- регламентирующие процессы создания, основные компоненты и структуру профилей прикладного окружения для: ***интерактивного взаимодействия*** с пользователями; мультипроцессорных систем; супер-компьютеров; систем реального времени:

* **IEEE 1003.10** – Профиль прикладного окружения для суперкомпьюторов;
* **IEEE 1003.11** – Профиль прикладного окружения для распределенной диалоговой обработки транзакций;
* **IEEE 1003.13** – Профиль прикладного окружения систем реального времени;
* **IEEE 1003.14** – Профиль прикладного окружения мультипроцессорных систем;
* **IEEE 1003.18** – Общие характеристики функциональной среды и профиль окружения прикладной платформы, соответствующей стандартам POSIX.

Параллельно с подготовкой новых стандартов группы POSIX из представленного списка, разработаны международные стандарты, углубляющие возможности создания мобильных приложений:

- **ISO 13210** – Методы тестирования для измерения соответствия стандартам POSIX.
- **ISO 14252** (развитие **IEEE 1003.0**) – Руководство по функциональной среде открытых систем POSIX;
- **ISO 14515-1** – Языки программирования, их функциональная среда и системные интерфейсы программного обеспе-

чения POSIX. Методы тестирования для проверки соответствия с POSIX. Часть 1. Системные интерфейсы;

- **ISO 14519-1** – Интерфейсы языка Ада, соответствующие POSIX. Часть 1. Связка для системного интерфейса прикладных программ;
- **ISO 15068-1** – Системная администрация POSIX. Часть 1. Администрация программных средств.

Идеология и модель создания мобильных программных средств изложена в документе **ISO 14252** (развитие **IEEE 1003.0**) – Руководство по POSIX окружению открытых систем (OSE), которое детализирует для пользователей модель комплекса стандартов POSIX, а также взаимодействующих с ними стандартов де-юре и 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. Предполагается, что прикладные программы непосредственно не взаимодействуют с внешним окружением, а связаны с ним только через операционную систему.

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. Эти интерфейсы и услуги более детально должны быть формализованы соответствующими частными стандартами POSIX. Документ включает также общие принципы и руководство по формированию и описанию сложных согласованных профилей и по обеспечению непротиворечивости их интерфейсов с внешним окружением. ***Понятие профиль*** в соответствии с **ISO 10000** и **IEEE 1003.0**, определяет документ, который предписывает состав, комплектацию, взаимосвязь и применение нескольких стандартов де-юре и де-факто с указанием параметров и директив, необходимых для построе-

ния сложных информационных систем или при реализации специальных функций информационных технологий.

Детализация интерфейсов начинается в стандарте **IEEE 1003.1** – Интерфейсы систем прикладных программ (язык Си), в котором уточняется концепция переносимости и принципы ее обеспечения путем унификации интерфейсов прикладных программ с операционными системами. Доработанная версия концептуального стандарта утверждена Международной организацией по стандартизации (**ISO 9945-1**) и аннотирована ниже. Основные команды управления и сервисные программы, обеспечивающие переносимость, изложены в стандарте **IEEE 1003.2** (Shell and utilities), который послужил базой одноименного стандарта **ISO 9945-2**. Таким образом, эти два стандарта являются основными в нормативной базе, поддерживающей перенос программ в операционной среде [22, 36, 43].

Развитие и конкретизация методов и средств, обеспечивающих мобильность прикладных программ, излагается в совокупности из 18, поддерживающих стандартов группы POSIX **IEEE 1003.3.20**, содержание которых изложена ниже. Фазу разработки и утверждения этими организациями каждого из них в данный момент трудно установить и реально действуют как международные стандарты только отмеченные выше. Остальные стандарты аннотированы далее в утвердительной форме как завершенные разработки, однако некоторые из них не вышли из стадии первоначальных проектов IEEE. По всем стандартам возможны и пока проводятся дополнения и изменения.

В стандартах POSIX (**IEEE 1003.5; -9; -16; -19; -20**) конкретизируются интерфейсы прикладных программ, разрабатываемых на стандартизированных языках программирования: Си (**ISO 9899**), Ада (**ISO 8652**), Фортран (**ISO 1359, 7846**). В базовых стандартах POSIX упоминаются некоторые особенности интерфейсов приложений на языках программирования Паскаль (**ISO 7185**), Кобол (**ISO 1989**). Рекомендуется применение языков в виде современных модернизированных и дополненных версий и использование для трансляции программ компиляторов, аттестованных авторитетными организациями.

В части взаимодействия мобильных прикладных программ и операционных систем с внешней средой стандарты POSIX, ориентируются на международные ***стандарты взаимосвязи***

открытых систем – ВОС (OSI), в которые входят группы стандартов [43]:

- 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.

Основы профилей представлены в документе **IEEE 1003.18**, в котором изложены профили интерфейсов приложений в интерактивных, распределенных, многопользовательских прикладных платформах. Стандарты **IEEE 1003.11; -13; –14** формализуют профили взаимодействия соответственно для диалоговой обработки заданий, для информационных систем реального времени и для мультипроцессорных вычислительных систем. NIST разработал комплект аттестационных тестов и обеспечивает тестирование в аккредитованных на соответствие POSIX испытательных лабораториях, а также выдает сертификаты проверки. Каталог проверенных программных средств публикуется в специальном издании.

Базовый стандарт **ISO 9945-1** – 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 современные, версии UNIX, а также на другие операционные системы, совместимые по интерфейсам с вер-

сиями UNIX. Считаются недопустимыми дополнительные интерфейсы разработчиков, не входящие в состав определенных стандартами POSIX. Разработчики стандарта стремились предусмотреть все потенциальные функции взаимодействия прикладных программ с ядром ОС на различных аппаратных платформах, в автономных, сетевых и распределенных информационных системах. Предусмотрены минимальные изменения и доработки при переносе прикладных программ и указываются их элементы, где возможны такие изменения.

Языково-ориентированные услуги и интерфейсы для языка Си в этом 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.

В описании стандарта выделены пять функциональных частей:

- предисловие, введение, общий взгляд на стандарт и нормативные рекомендации (раздел 1);
- терминология, общие требования и концепция стандартизации интерфейсов прикладных программ (раздел 2);
- процессы, средства и описания компонент интерфейсов изложены в семи разделах:
  * компоненты процессов взаимодействия прикладных программ (раздел 3);
  * процессы взаимодействия с окружением, с терминалами и пользователями (раздел 4);
  * файлы и директории – характеристики, создание, изменение, перемещение (раздел 5);
  * примитивы ввода и вывода – манипуляции, передача и контроль операций с файлами (раздел 6);
  * взаимосвязь с внешними устройствами – описание и контроль интерфейсов терминалов (раздел 7);
  * языково-ориентированные услуги и интерфейсы для языка программирования Си (раздел 8);
  * системные базы данных – функции и интерфейсы доступа (раздел 9);

- форматы архивации и обмена файлами данных (раздел 10).

Подчеркивается целесообразность применения для взаимодействия с внешней средой концепции и совокупности стандартов, являющихся развитием базовой эталонной модели взаимосвязи открытых систем (ВОС) – **ISO 7498**.

В ***стандарте*** **ISO 9945-2**, который является развитием стандарта **IEEE 1003.2**, изложено конкретное представление команд операционной системы и утилит, обеспечивающих унифицированное взаимодействие с мобильными прикладными программами. Первичная цель этого 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. Факультативные утилиты расширяют возможности для пользователей мобильных приложений по связи с внешним окружением. В терминах языка Си (стандарт **ISO 9899:1990**) описаны языковонезависимые 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. Язык программирования оболочки позволяет создавать мобильные программы для выполнения действий, которые позволяют комбинировать функции отдельных утилит прямо из прикладных программ или вызывать из базовых директив. Реализация команд и функций утилит возможна в операционных системах конкретных поставщиков, которые соответствуют данной спецификации.

Стандарт содержит 7 разделов и 7 приложений (3 из них обязательные). В двух первых разделах представлены общие сведения об интерфейсах прикладных программ, нормативные ссылки, структурные требования, методы тестирования, терминология. Далее детально описаны:

- язык управления (shell) (раздел 3);

- утилиты среды взаимодействия программ (раздел 4);
- управляющие утилиты, поддерживающие мобильность программ и
- связь внешних пользователей с асинхронными терминалами (раздел 5);
- утилиты для взаимодействия комплексов программ (раздел 6);
- языково-независимые системные услуги для прикладных программ на нескольких языках программирования высокого уровня (раздел 7).

Всего описано более 130 директив. Описание предназначено для разработчиков прикладных программ, создателей и покупателей информационных систем и конечных пользователей.

В трех обязательных приложениях (A, B, C) конкретизируются детали утилит интерфейсов для использования в прикладных программах на языках Си и Фортран.

Стандарт **IEEE 1003.3** 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 9945-2**. 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 [9, 62]. Приводятся рекомендации по:

- классификации тестовых сценариев и утверждений (раздел 5);
- подготовке описаний детерминированных тестов и тестов для проверки структурных конструкций (раздел 6);
- кодированию и представлению результатов тестирования во взаимосвязи с исходными тестами (раздел 7);
- формированию удостоверения о соответствии интерфейсов прикладных программ стандартам POSIX (раздел 8).

В информационном приложении А отражены ссылки на 38 стандартов ВОС ISO и языки программирования. Тестирование интерфейсов приложений на соответствие стандартам POSIX более полно представлено в [9, 22, 62].

Стандарт **IEEE 1003.4** дополняет и конкретизирует интерфейсы программ, которые для мобильных приложений, функционирующих в реальном времени, требуют изменений и расширений административного управления относительно синтаксических и семантических положений документа **IEEE 1003.1**. Стандарт охватывает три основных направления:

- файловые системы реального времени;
- согласование функционирования многокомпонентных архитектур ЭВМ;
- распределение использования памяти посредством семафоров и сигналов.

Вводятся понятия, детально рассмотрены применение и интерфейсы: таймеров, приоритетных очередей, управление списками, файлов реального времени, семафоров, синхронного и асинхронного ввода-вывода, блокировки памяти. Выделены разделы, посвященные обеспечению обмена сообщениями с внешней средой, организации цепных списков, их управлению и корректировке, слежению за исполнительным временем процессора, управлению прерываниями и взаимодействию с внешними устройствами. Функциональные спецификации подверглись модификациям в направлении расширения стандартизуемых функций во взаимодействии со стандартом на профили интерфейсов приложений реального времени **IEEE 1003.13**. Оба стандарта, несомненно, очень важны для создания многих сложных мобильных ИС, которые функционируют в реальном времени.

В ***стандарте*** **IEEE 1003.5** конкретизируются описания связей прикладных программ, представленные в **IEEE 1003.1** и **1003.2** и необходимые для комплексирования с операционной системой мобильных приложений, разработанных на языке Ада [2, 36].

Дополнения **IEEE 1003.6** для обеспечения безопасности и защиты информации интерфейсов прикладных программ расширяют положения стандартов POSIX для обеспечения четырех уровней защиты критических информационных систем. Кроме того, представлены 10 классов критериев защиты. Описаны осо-

бенности услуг дискретного и мандатного контроля, проверки безопасности интерфейсов и защиты от несанкционированного доступа. Введены функции привилегий, которые предназначены для регламентирования доступа пользователей разных рангов, а также функции ревизий (аудит) для регистрационных механизмов доступа, записей и контроля. Отмечаются особенности тестирования защиты информации и взаимодействие со стандартами **IEEE 1003.7** и **-8**. Стандарт ориентирован на активное применение методов и средств обеспечения безопасности функционирования прикладных программ, изложенных в группе стандартов ISO, посвященных защите в открытых системах и криптографии [7, 27, 38, 43]. Одной из самых сложных и недостаточно проработанных проблем остается аттестационное тестирование и удостоверение соответствия разработанных прикладных программ данному стандарту. Более полно обеспечение безопасности ИС представлено в стандартах **ISO 15408:1-3 , ISO 13335:1-5.**

Стандарт **IEEE 1003.7** содержит системные интерфейсы и сервисные программы для управления при инсталляции, конфигурировании и сопровождении версий прикладных ПС и БД окружения операционных систем. Рассматривается административное управление со стороны пользователей системными файлами, сопровождением и развитием версий прикладных программ, услугами коммуникаций, а также формирование и печать документов в соответствии со стандартом **ISO 10175**. Стандарт учитывает требования по управлению **IEEE 1003.15** и по защите информации **IEEE 1003.6**, а также отражает управление в системе стандартов де-факто X/Open и протоколы управления в распределенных гетерогенных сетях в соответствии со стандартами OSI и концепцией UNIX.

Стандарт **IEEE 1003.8** ориентирует разработчиков и пользователей на применение базовой эталонной модели ISO взаимосвязи открытых систем (ВОС) коммуникации и ее развитие в стандартах и протоколах верхних уровней ВОС, на стандарты обмена в файловых системах (FTAM, NFS, RFS). Изложена концепция стандарта **ISO 8571** - передача, доступ и управление файлами (FTAM), который регламентирует услуги по всем функциям манипулирования файлами, а также отражены процедуры управления файлами в распределенных информационных системах [43].

Стандарт **IEEE 1003.9** специфицирует языковонезависимые связи мобильных прикладных программ, разработанных на языке Фортран-77 (ISO) с операционной системой UNIX. Детализируются утилиты интерфейсов, представленные в приложении "С" стандарта **ISO 9945-2**.

Стандарт **IEEE 1003.10** определяет профили прикладного окружения для суперкомпьютеров, которые отличаются от традиционных моделей распределения времени в UNIX в части приоритетного обслуживания, наличия рестарта и эффективного ввода-вывода. Особенностями этих систем являются также большой объем прикладных программ, интенсивное использование процессора и памяти. Профили используют сценарии заданий, сетевые интерфейсы и графику. Кроме того, могут привлекаться базовые стандарты POSIX, стандарты на языковые интерфейсы и некоторые стандарты OSI на файловые системы и коммуникацию.

В ***стандарте*** **IEEE 1003.11** представлены профили прикладного окружения для распределенной диалоговой обработки заданий. Документ опирается на стандарты ВОС **ISO 10026-1, -6**, на базовые стандарты POSIX и на стандарты де-факто X/Open и их детализацию при обработке удаленных заданий. Поддерживается высоконадежный ввод-вывод, использование эффективных структур файлов, интерфейсов со стандартизированными базами данных, ориентированными на распределенные диалоговые системы ВОС и модель X/Open.

В ***стандарте*** **IEEE 1003.12** регламентированы схемы реализации интерфейсов прикладных программ для взаимодействия с протоколами коммуникации верхних уровней базовой эталонной модели ВОС, вплоть до выхода на транспортный уровень ВОС [43].

Документ **IEEE 1003.13** отражает профили прикладного окружения систем реального времени. Он является развитием стандарта **IEEE 1003.4** и тесно с ним связан. Он поддерживает реализацию функций взаимодействия встроенных систем реального времени, а также взаимодействия окружения с однопроцессорными файловыми системами. Для жесткого регламента времени он покрывает экстенсивный ввод-вывод; логику управления; нормализацию и обработку сигналов прерывания; временные вызовы; циклические операции.

***Стандарт* IEEE 1003.14** формализует профили прикладного окружения мультипроцессорных систем, поддерживает только частный вид платформ, представленных типовыми мультипроцессорными системами и не предназначен быть основой для пользователей при разработке других профилей. По структуре он подобен стандарту **IEEE 1003.18**.

***Стандарт* IEEE 1003.15** является развитием документа **IEEE 1003.10** (профили прикладного окружения для суперкомпьютеров) и дополняет процессы административного управления системами (**IEEE 1003.7**) для очередей пакетов прикладных программ. Он регламентирует такие особенности работ, как очень длительное исполнение программ и восстановление окружения после прерывания.

***Стандарты* IEEE 1003.16; -19; -20** развивают и формализуют интерфейсы для мобильных приложений, созданных на языках программирования Си, Фортран и Ада соответственно. Документ **IEEE 1003.19** детализирует интерфейсы программ, разработанных на языке Фортран-90 и их взаимодействие с интерфейсами на языке Си (**IEEE 1003.16**) и далее с языковонезависимым представлением интерфейсов. Стандарт **IEEE 1003.20** расширяет первичные интерфейсы программ на языке Ада (**IEEE 1003.5**) в направлении обеспечения систем реального времени и полного применения функций языка Ада.

***Стандарт* IEEE 1003.17** регламентирует обслуживающие каталоги (директории) интерфейсов переносимых приложений. Каталоги услуг предназначены для применения пользователями детальных сетевых интерфейсов (DNI) в комбинации с сетевыми интерфейсами OSI и стандартами X.500 для распределенных POSIX систем.

***Профили окружения платформы POSIX* – IEEE 1003.18** являются развитием концепций стандарта **IEEE 1003.1**. Эти профили определяют окружение, взаимодействующее с пользователями в традиционных распределенных, интерактивных, многопользовательских системах. Устанавливается взаимодействие средств для кросс-компиляции программ с одной платформы для использования на другой аппаратной платформе. Различается платформа разработки приложений, где программы компилируются или интерпретируются, от платформы исполнения приложений, где программы функционируют и решают задачи. Тем самым программы могут компилироваться на много-

пользовательской системе с большими ресурсами для исполнения на однопользовательской персональной ЭВМ. Системы могут соединяться сетевыми средствами, коммутируемой линией или взаимодействовать через переносимые магнитные ленты или диски.

В большинстве стандартов 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 более чем на порядок. В конкретных прикладных разработках каждый раз необходима только некоторая, относительно небольшая часть из них, которую целесообразно выделять в соответствующий проблемно-ориентиро-ванный профиль (см. п. 4.2). Однако неравномерность разработки и формализации стандартов приводит к тому, что в профилях образуются не стандартизированные пробелы, в которых разработчики вынуждены принимать специфические, самостоятельные решения. Неоднозначность конкретных проектных решений усугубляется параллельным, активным развитием стандартов де-факто с подобными функциями, позволяющих выбирать из них наиболее подходящие для конкретной ситуации.

Рядом зарубежных организаций и промышленных фирм ведется разработка новых версий стандартов открытых систем **POSIX 200x** [36]. К 2000-му году выполнена большая работа по пересмотру и реорганизации базовых спецификаций POSIX и подготовлен проект фундаментального международного стандарта **IEEE 1003.1 – 200x**, объемом свыше трех тысяч страниц. Цель документа – поддержка 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 тома включают: базовые определения; оболочки и утилиты; системные интерфейсы. Кроме того, имеется том обоснования выбранных решений. Важными свойствами разработанных программных интерфейсов являются модульность их построения и параметризуемость. Основной акцент делается на функции и средства, используемые в коммерческих приложениях.

Рекомендательный характер всей совокупности зарубежных стандартов предполагает возможность творческого выбора на-

боров, адекватных особенностям проектируемых информационных систем, а также аппаратных и операционных платформ. Несмотря на пробелы и неопределенности в стандартизации обеспечения переносимости прикладных программ, созданная и развивающаяся группа стандартов POSIX 200x решает основные задачи в проблеме обеспечения мобильности программ.

4.5. Особенности профилей открытых информационных систем, программных средств и баз данных

Профиль открытых информационных систем представляет собой специально подобранную совокупность стандартов, которая, для определенного класса или области приложений, определяет на языке функциональных требований, протоколов, форматов данных, поведение ПС и БД при их взаимодействии между собой, с операционной и аппаратной внешней средой. Профиль документирует основные технические требования стандартов, в которых в формализованном виде представлены потребности в информационных системах.

Известен ***ряд методов разработки профилей открытых систем***, предложенные поставщиками ИС, промышленными ассоциациями, а также методы независящие от конкретных поставщиков. Требования к профилю, выраженные в терминах требований к системе и функций информационной системы, необходимых для их поддержки, разрабатывают способом ***сверху-вниз***. Напротив, выделение функций, необходимых для обеспечения ИС, выраженных в терминах стандартов и продуктов, проводят итеративным способом ***снизу-вверх***. Это требует наличия опытных профессионалов в области ИС, умеющих обнаружить несоответствия между требованиями практической деятельности и действующим стандартом или продуктом, имеющимся на рынке. В свою очередь, наличие нового технологического стандарта или продукта может привести к изменениям в деловой стратегии в целом и, следовательно, к изменениям профиля открытой системы используемого предприятием или проектом.

В международной, функциональной стандартизации ИС принята ***жесткая трактовка понятия профиля***. Считается, что основой профиля могут быть только международные и национальные, утвержденные стандарты (не допускается исполь-

зование стандартов де-факто и нормативных документов фирм). Подобное понятие профиля активно используется в гамме международных стандартов, конкретизирующих и регламентирующих основные процессы и объекты взаимосвязи открытых систем (ВОС), в которых возможна и целесообразна жесткая формализация профилей (функциональные стандарты **ISO 10607 - 10613** и соответствующие им ГОСТ Р). Однако при таком подходе невозможны унификация, регламентирование и параметризация множества конкретных функций и характеристик сложных объектов современных ИС. Более гибким является ***прагматический подход к разработке и применению профилей ИС***, который состоит в использовании совокупности адаптированных и параметризованных базовых международных и национальных стандартов и открытых спецификаций, отвечающих стандартам "де-факто" и рекомендациям международных консорциумов.

В качестве методологической базы построения и применения профилей сложных, распределенных ИС целесообразно использовать технический отчет **ISO TR 10000**. Части 1 и 2 этого документа введены в России в качестве ГОСТ Р. Часть 3, определяющую основы и таксономию профилей среды открытых систем, предлагается использовать при построении и применении профилей ИС как документ прямого применения. Эталонная модель среды открытых систем (OSE/RM см.п.4.3) определяет разделение любой информационной системы на приложения (прикладные программы и программные продукты) и среду, в которой эти приложения функционируют. Между приложениями и средой определяются стандартизированные интерфейсы. Эти интерфейсы являются необходимой частью профилей любой открытой системы. Кроме того, в профилях ИС могут быть определены унифицированные интерфейсы взаимодействия прикладных программ (функциональных частей) между собой и интерфейсы взаимодействия между компонентами внешней среды ИС. В соответствии с определениями профиля и базовых стандартов, входящих в профиль, спецификации выполняемых функций и интерфейсов взаимодействия могут быть оформлены как профиль каждого компонента системы. Таким образом, ***профили ИС***, как сложной открытой системы с иерархической структурой, ***могут включать***:

- стандартизированные описания функций, выполняемых данной системой, и взаимодействия с внешней для нее средой;
- стандартизированные интерфейсы между приложениями и средой ИС;
- профили отдельных функциональных компонентов, входящих в систему, в программные средства и в информацию баз данных;
- существующие и потенциальные требования, которые используются при построении профиля открытой системы:
  * стратегия основной деятельности предприятия;
  * практическая реализация (функциональная архитектура) стратегии основной деятельности;
  * стратегия управления предприятием;
  * стратегия информатизации предприятия;
  * требования конечного пользователя ИС;
- стандарты, которые предположительно этим требованиям удовлетворяют:
  * действующие стандарты, принятые и утвержденные в установленном порядке;
  * создаваемые стандарты (проекты);
  * фактические стандарты (стандарты де-факто);
  * спецификации и документы, которые удовлетворяют требованиям профиля открытой системы;
  * планы изменения требований пользователя к ИС и необходимых ему услуг;
  * стратегия замены одних информационных систем другими.

Наиболее актуальными в настоящее время представляются открытые распределенные системы с архитектурой "клиент-сервер". Профиль среды ИС должен определять её архитектуру в соответствии с выбранной моделью распределенной обработки данных. Стандарты интерфейсов ПС и БД со средой ИС должны быть определены по функциональным областям профилей. Декомпозиция структуры среды функционирования ИС на составные части, выполняемая на этапе проектирования, позволяет детализировать ***профиль среды*** ИС по функциональным областям эталонной модели OSE/RM:

- графического пользовательского интерфейса;
- реляционных или объектно-ориентированных СУБД ;

- операционных систем с учетом сетевых функций, выполняемых на уровне операционной системы;
- телекоммуникационной среды в части услуг и сервисов прикладного уровня: электронной почты, доступа к удаленным базам данных, передачи файлов, доступа к файлам и управления файлами.

Выбор аппаратных платформ ИС связан с определением требуемых их параметров: вычислительной мощности серверов и рабочих станций в соответствии с проектными решениями по разделению функций между клиентами и серверами; степени масштабируемости аппаратных платформ; их надежности. Профиль внешней среды ИС должен содержать стандарты, определяющие параметры технических средств и способы их измерения (например, стандартные тесты измерения производительности).

Состав инструментальных средств определяется на основании решений и нормативных документов об организации сопровождения и развития открытой системы. При этом должны быть учтены правила и порядок, регламентирующие внесение изменений в действующие компоненты и системы. Функциональная область профиля инструментальных средств охватывает функции централизованного ***управления и администрирования***, связанные с:

- контролем корректности функционирования и производительности открытой системы в целом;
- управлением конфигурацией прикладных программных средств и тиражированием версий;
- управлением доступом пользователей к ресурсам открытой системы и конфигурацией ресурсов;
- настройкой пользовательских интерфейсов (генерация экранных форм и отчетов);
- ведением баз данных открытой системы;
- восстановлением работоспособности системы после сбоев и аварий.

Ссылки на соответствующие стандарты, входящие в ***профиль среды***, должны быть указаны в профиле инструментальных средств. В этом профиле должны быть также предусмотрены ссылки на требования к средствам тестирования, которые необходимы для процессов сопровождения и развития открытой

системы. В состав ***ИС для тестирования профиля открытой системы должны входить средства, обеспечивающие***:

- функциональное тестирование профилей ПС и БД;
- тестирование корректности интерфейсов пользователей и внешней среды;
- системное квалификационное тестирование профиля открытой системы.

После выбора аппаратных платформ, типа СУБД и других компонентов внешней среды ИС создаются прототипы приложений, рассчитанные на схему "клиент-сервер". Выполняя одни и те же тесты на разных прототипах компонентов, проектировщик может уточнить и оптимизировать архитектуру проектируемой открытой системы за счет рационального распределения функций между её компонентами. В результате окончательно определяется и оформляется профиль среды ИС, который в дальнейшем применяется при разработке ПС и БД, комплексировании и испытаниях открытой системы, а также при модернизации и расширении системы, связанных с модификацией её компонентов.

ЛИТЕРАТУРА

1. Андон Ф.И., Коваль Г.И., Коротун Т.М., Суслов В.Ю. Основы инженерии качества программных систем. – К.: Академпериодика. 2002.
2. Бар Р. Язык Ада в проектировании систем: Пер. с англ. /Под ред. Е.К. Масловского. – М.: Мир, 1988.
3. Бойченко А.В., Кондратьев В.К., Филинов Е.Н. Основы открытых информационных систем. – М. МГУЭСИ. 2004.
4. Боэм Б.У. Инженерное проектирование программного обеспечения. Пер. с англ. /Под ред. А.А. Красилова. – М.: Радио и связь, 1985.
5. Вигерс К.И. Разработка требований к программному обеспечению. Пер. с англ. – М.: Русская редакция. 2004.
6. Вендров А.М. Проектирование программного обеспечения экономических информационных систем. Учебник. – М.: Финансы и статистика. 2002
7. Герасименко В.А. Защита информации в автоматизированных системах обработки данных. Книга 1 и 2. – М.: Энергоатомиздат. 1994.
8. Гласс Р., Нуазо Р. Сопровождение программного обеспечения: Пер. с англ. /Под ред. Ю.А. Чернышева. – М.: Мир, 1983.
9. Дастин Э., Рэшка Д., Пол Д. Автоматизированное тестирование программного обеспечения. Внедрение, управление и эксплуатация. Пер. с англ. – М. ЛОРИ. 2003.
10. Зайцев С.С., Кравцунов М.И., Ротанов С.В. Сервис открытых информационно - вычислительных сетей: Справочник. – М.: Радио и связь, 1990.

11. Калбертсон Р., Браун К., Кобб Г. Быстрое тестирование. Пер. с англ. – М.: Вильямс.2002.
12. Калянов Г.Н. Теория и практика реорганизации бизнес-процессов. – М.: СИНТЕГ. 2000.
13. Кантор М. Управление программными проектами. Практическое руководство по разработке успешного программного обеспечения. Пер. с англ. – М.: Вильямс. 2002.
14. Когаловский М.Р. Энциклопедия технологий баз данных. – М.: Финансы и статистика. 2002.
15. Колташев А. Особенности переноса бортовых программ. – М. Открытые системы. №4. 2004.
16. Крайер Э. Успешная сертификация на соответствие нормам ИСО серии 9000. Пер. с нем. – М.: ИЗДАТ. 1999.
17. Кузнецов С.Д. Основы баз данных. Курс лекций. Учебное пособие. – М. ИНТУИТ. 2005.
18. Леман М.М. Программы, жизненные циклы и законы эволюции программного обеспечения //ТИИЭР. Техника программного обеспечения: Пер. с англ. – М.:Мир. 1980. - Т.68. - N 9. - С.26-45.
19. Леффингуэлл Д., Уидриг Д. Принципы работы с требованиями к программному обеспечению. Унифицированный подход. Пер. с англ. – М.: Вильямс. 2002.
20. Липаев В.В., Потапов А.И. Оценка затрат на разработку программных средств. – М.: Финансы и статистика. 1988.
21. Липаев В.В. Отладка сложных программ. – М.: Энергоатомиздат. 1993.
22. Липаев В.В., Филинов Е.Н. Мобильность программ и данных в открытых информационных системах. – М.: РФФИ. 1997.
23. Липаев В.В. Выбор и оценивание характеристик качества программных средств. – М.: СИНТЕГ. 2001.
24. Липаев В.В. Системное проектирование сложных программных средств для информационных систем. Изд. второе переработанное и дополненное. – М.: СИНТЕГ. 2002.
25. Липаев В.В. Методы обеспечение качества крупномасштабных программных средств. – М.: РФФИ. СИНТЕГ. 2003.
26. Липаев В.В. Технико-экономическое обоснование проектов сложных программных средств. – М.: СИНТЕГ. 2004.

27. Липаев В.В. Функциональная безопасность программных средств. – М.: СИНТЕГ. 2004.
28. Липаев В.В. Анализ и сокращение рисков проектов сложных программных средств. – М.: СИНТЕГ. 2004.
29. Липаев В.В. Документирование сложных программных средств. М.: СИНТЕГ. 2005.
30. Мизин И.А.-ред. Справочник. Протоколы информационно-вычислительных сетей. – М.: Радио и связь, 1990.
31. Мильман К., Мильман С. CMMI – шаг в будущее. Открытые системы. №5-6 и №7-8. 2005.
32. Мобильность программного обеспечения: Пер. с англ. /Под ред. Д.Б. Подшивалова. – М.:Мир, 1980.
33. Оценка и аттестация зрелости процессов создания и сопровождения программных средств и информационных систем (ISO/IEC TR 15504 – CMM). – М.: Книга и бизнес. 2001.
34. Прангишвили И.В. Системный подход и общесистемные закономерности. – М.: СИНТЕГ. 2000.
35. Руководство по проектированию профилей среды открытых систем. Пер. с англ. – М. Янус-К. 2002.
36. Сухомлин В.А. Введение в анализ информационных технологий. Учебник для вузов. – М. Горячая линия – Телеком. 2003.
37. Трахтенгерц Э.А. Субъективность в компьютерной поддержке управленческих решений. – М.: СИНТЕГ. 2001.
38. Трубачев А.П., Долинин М.Ю., Кобзарь М.Т. и др. Оценка безопасности информационных технологий. Общие критерии. Под ред. В.А. Галатенко. – М.: СИП РИА, 2001.
39. Тэллес М., Хсих Ю. Наука отладки. – М.: Кудиц-образ. 2003.
40. Уайт Б.А. Управление конфигурацией программных средств. Практическое руководство по Rational ClearCase. Пер. с англ. – М. ДМК Пресс. 2002.
41. Фатрелл Р.Т., Шафер Д.Ф., Шафер Л.И. Управление программными проектами: достижение оптимального качества при минимальных затратах. Пер. с англ. – М.: Вильямс. 2003.
42. Шаракшанэ А.С., Шахин В.П., Халецкий А.К. Испытания программ сложных автоматизированных систем. – М.: Высшая школа. 1982.

43. Щербо В.К., Козлов В.А. Функциональные стандарты в открытых системах. Ч. 1. Концепция открытых систем. Ч.2. Международные функциональные стандарты. – М.: Изд. МЦНТИ. 1997.
44. APP - Application Portability Profile. The U.S. Government open system environment profile OSE/1 Version 2.0.NIST. Washington. 1993.
45. Boehm B.W. et al. Software cost estimation with COCOMO II. Prentice Hall PTR. New Jersey. 2000.
46. Beizer B. Software testing techniques. N.Y.: Van Nostrand Reinhold. 1990.
47. Buckle J.K. Software configuration management. - London: Macmillan Press. 1982.
48. Charett R. Software engineering risk analysis and management. N.Y.: McGraw – Hill. 1989.
49. Davis A. Software requirements: Objects, functions and states. – Englewood Cliffs. NY. Prentice-Hall. 1993.
50. Grady R. Practical software metrics for project management and process improvement. – Englewood Cliffs. NY. Prentice-Hall. 1992.
51. Encyclopedia of Software Engineering. Vol.1 A-N; Vol.2 O-Z. Editor – In – Chief John J. Marciniak. John Wiley & Sons. Inc. 1995.
52. Jones C. Applied software measurement, assuring productivity and quality. McGraw-Hill. NY. 1996.
53. Higuera R., Haimes Y. Software risk management. Pittsburg. Software engineering institute, Cornegie Mellon University. – 1996.
54. Karolak D. W. Software engineering risk management. IEEE Computer Society Press. Washington. 1996.
55. Krishnamurthy P., Balanchander A. Practical Reusable Unix Software. John Wiley & Sons. 1995.
56. Londeix B. Cost estimation for software development. Cornwall: Addison-Wesley. 1987.
57. Martin J., McClure C. Software maintenance, the problems and its solutions. - N.Y.: Prentice-Hall.1983.
58. Musa J.D., Iannino A., Okumoto K. Software Reliability: Measurement, Prediction, Application.N.Y. McGraw Hill. 1987.

59. Ng P.A., Yeh R.T. ed. Modern software engineering. Foundations and current perspectives. - N.Y.: Van Nostrand Reinhold. 1990.
60. Open systems handbook: A guide to building open systems. Digital Equipment Corparation. USA, 1991.
61. Perley D.R. Migrating to open systems: taming the tiger. McGraw-Hill. UK, 1993.
62. Quarterman J.S., Wilhelm S. Unix, Posix and open systems: The open standards puzzle. N.Y., Addison - Wesley. 1993.
63. Schindler M.J. Computer- aided software design. Build quality software with CASE. - N.Y. John Wiley & Sons, 1990.
64. Sommerville I. Software engineering. Lancaster University. Addison-Wes-ley.2000.

Приложение

ПЕРЕЧЕНЬ ОСНОВНЫХ СТАНДАРТОВ, РЕГЛАМЕНТИРУЮЩИХ СОПРОВОЖДЕНИЕ И УПРАВЛЕНИЕ КОНФИГУРАЦИЕЙ ПРОГРАММНЫХ СРЕДСТВ

1. **ISO 9000:2000.** (ГОСТ Р – 2001). Система менеджмента (административного управления) качества. Основы и словарь.
2. **ISO 9001:2000.** (ГОСТ Р – 2001). Система менеджмента (административного управления) качества. Требования.
3. **ISO 9004:2000.** (ГОСТ Р – 2001). Система менеджмента (административного управления) качества. Руководство по улучшению деятельности.
4. **ISO 90003:2004** – Руководство по организации применения стандарта **ISO 9001:2000** для программных средств.
5. **ISO 10005: 1995** – Административное управление качеством. Руководящие указания по программам качества.
6. **ISO 10006: 1997** – Руководство по качеству при управлении проектом.
7. **ISO 10007: 1995** – Административное управление качеством. Руководящие указания при управлении конфигурацией.
8. **ISO 10011-1-3: 1990.** – Руководящие положения по проверке систем качества. Ч.1. Проверка. Ч.2. Квалификационные

критерии для инспекторов-аудиторов систем качества. Ч.3. Управление программами проверок.
9. **ISO 10013: 1995.** – Руководящие указания по разработке руководств по качеству.
10. **ISO 14252:1996** (IEEE 1003.0). – Руководство по функциональной среде открытых систем POSIX.
11. **ISO 9945-1:1990** (IEEE 1003.1). – ИТ. Интерфейсы переносимых операционных систем. Ч.1. Интерфейсы систем прикладных программ (язык Си).
12. **ISO 9945-2:1992** (IEEE 1003.2). – ИТ. Интерфейсы переносимых операционных систем. Часть 2. Команды управления и сервисные программы.
13. **ISO 13210:1994.** – ИТ. Методы тестирования для измерения соответствия стандартам POSIX.
14. **ISO 15288:2002.** – Системная инженерия. Процессы жизненного цикла систем.
15. **ISO 19760:2003.** – Системная инженерия. Руководство по применению стандарта **ISO 15288.**
16. **ISO 12207:1995.** (ГОСТ Р – 1999). – ИТ. Процессы жизненного цикла программных средств.
17. **ISO 15271:1998.** (ГОСТ Р – 2002). – ИТ. Руководство по применению ISO 12207.
18. **ISO 16326:1999.** (ГОСТ Р – 2002). – ИТ. Руководство по применению ISO 12207 при административном управлении проектами.
19. **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.
20. **ГОСТ Р 51904 – 2002**. – Программное обеспечение встроенных систем. Общие требования к разработке и документированию.
21. **ISO 12182:1998.** (ГОСТ Р– 2002). – ИТ. Классификация программных средств.

22. **ISO 9126:1991.** (ГОСТ – 1993). – ИТ. Оценка программного продукта. Характеристики качества и руководство по их применению.
23. **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.
24. **ISO 9126:1-4:2002-2004**. ТО. – Качество программных средств: Ч.1. Модель качества. Ч.2. Внешние метрики. Ч. 3. Внутренние метрики. Ч. 4. Метрики качества в использовании.
25. **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**.
26. **ANSI/IEEE 829 - 1983**. – Документация при тестировании программ.
27. **ANSI/IEEE 1008 - 1986**. – Тестирование программных модулей и компонентов ПС.
28. **ANSI/IEEE 1012 - 1986**. – Планирование верификации и подтверждения достоверности качества (валидации) программных средств.
29. **ISO 12119:1994.** (ГОСТ Р – 2000 г). ИТ. – Требования к качеству и тестирование.
30. **ISO 14756: 1999.** ИТ. – Измерение и оценивание производительности программных средств компьютерных вычислительных систем.
31. **IEC 61508:1-6:1998-2000.** – Функциональная безопасность электрических / электронных и программируемых электронных систем. Часть 3. Требования к программному обеспечению. Часть 6. Руководство по применению стандартов IEC 61508-2 и IEC 61508-3.
32. **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.
33. **ISO 13335:1-5:1996-1998.** ТО.– Руководство по управлению безопасностью. Ч. 1. Концепция и модели обеспечения безопасности информационных технологий. Ч.2. Планиро-

вание и управление безопасностью информационных технологий. Ч.3. Техника управления безопасностью ИТ. Ч.4. Селекция (выбор) средств обеспечения безопасности. Ч.5. Безопасность внешних связей.
34. **IEC 60880:1998.** – Ч. 1.1986. Программное обеспечение компьютеров в системах безопасности атомных электростанций. Ч.2. 2000. Программные аспекты защиты от отказов по общим причинам, использование программных инструментов и ранее разработанного программного обеспечения.
35. **ISO 16085: 2004.** – Характеристики процессов управление рисками при разработке, применении и сопровождении программных средств.
36. **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.
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 6592:2000.** ОИ. – Руководство по документации для вычислительных систем.
40. **ISO 9294:1990.** (ГОСТ–1993 г). ТО. ИТ. – Руководство по управлению документированием программного обеспечения.
41. **ISO 15910:1999.** (ГОСТ Р – 2002) ИТ. – Пользовательская документация программных средств.
42. **ISO 18019:2004.** ИТ. – Руководство по разработке пользовательской документации на прикладные программные средства для офисов, бизнеса и профессиональных применений.
43. **РД 50-34.698-90.** – Методические указания. Информационная технология. Автоматизированные системы. Требования к содержанию документов.
44. **ГОСТ Р 51901-2002.** – Управление надежностью. Анализ риска технологических систем.

45. **DO-178 B -1995.** – Соглашение по сертификации бортовых систем и оборудования в части программного обеспечения.
46. **ISO 14102:1995.** – Оценка и выбор CASE- средств.
47. **ISO 14471:1995.** – Руководство по адаптации CASE-средств.
48. **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.