Федеральное агентство по образованию

Томский государственный университет систем управления
и радиоэлектроники

Ю.П. Ехлаков

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ И ПРОГРАММНЫЕ ПРОДУКТЫ: РЫНОК, ЭКОНОМИКА, НОРМАТИВНО-ПРАВОВОЕ РЕГУЛИРОВАНИЕ

Учебное пособие

Томск
ТУСУР
2007

УДК 002.6+681.3.06.002](075.8)
ББК 73.05я73+32.97я73
Е 93

Рецензенты:

Силич В.А., д-р техн. наук, профессор,
зав. кафедрой оптимизации систем управления
Томского политехнического университета

Ямпольский С.З., канд. техн. наук, доцент, Генеральный директор
ОАО «Томский международный деловой центр «Технопарк»

Ехлаков Ю.П.

Е 93 Информационные технологии и программные продукты: рынок, экономика, нормативно-правовое регулирование: учеб. пособие / Ю.П. Ехлаков. — Томск: Томск. гос. ун-т систем управления и радиоэлектроники, 2007. — 176 с.

ISBN 978-5-86889-390-2

Раскрывает одно из направлений программной инженерии, связанное с нормативно-правовыми и экономическими механизмами разработки, продвижения на рынок и внедрения программных систем и технологий.

Предназначено для студентов, обучающихся по направлениям 654600 «Информатика и вычислительная техника» и 080700 «Бизнес-информатика», а также может быть рекомендовано студентам, магистрантам, аспирантам, специализирующимся на разработке программных продуктов и информационных технологий.

Разработано и издано в рамках выполнения Инновационной образовательной программы Томского государственного университета систем управления и радиоэлектроники.

УДК 002.6+681.3.06.002](075.8)
ББК 73.05я73+32.97я73

Учебное издание

Ехлаков Юрий Поликарпович

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ И ПРОГРАММНЫЕ ПРОДУКТЫ:
РЫНОК, ЭКОНОМИКА, НОРМАТИВНО-ПРАВОВОЕ РЕГУЛИРОВАНИЕ

Учебное пособие

Редактор Коновалова Н.В.
Корректор Чапля В.В.

Подписано в печать 03.09.07. Формат 60x84/16.
Усл. печ. л. 10,23. Тираж 170 экз. Заказ 1069.

Томский государственный университет систем управления
и радиоэлектроники, 634050, г. Томск, пр. Ленина, 40. Тел. (3822) 53-30-18.

ISBN 978-5-86889-390-2

Оглавление

Введение

Каждый выпускник, окончивший университет, в той или иной мере связан с информационными технологиями. Одни из них становятся пользователями, перед которыми встают проблемы выбора информационных технологий (ИТ) в соответствии с их профессиональными потребностями, другие — разработчиками этих технологий.

Как найти на рынке ИТ нужный программный продукт? Как сформулировать требования к его качеству и процедуре приобретения. Как организовать конкурсные торги, какой тип лицензионного соглашения выбрать? Если на рынке информационных технологий нет продукта, удовлетворяющего пользователей, необходимо сформулировать требования на его разработку, определить трудозатраты и стоимость проекта, правильно выбрать коллектив разработчиков. Это далеко не полный перечень проблем, с которыми проходится сталкиваться пользователям.

Молодые специалисты-разработчики информационных технологий, становясь руководителями проектов, также сталкиваются с рядом проблем: как убедить заказчика не покупать готовые решения, а заказать разработку программного обеспечения (ПО) с учетом специфики своего бизнеса; как определить и обосновать трудозатраты на создание программной системы (ПС); как аргументировано предложить заказчику договорную цену; как грамотно, с учетом зарубежных и отечественных стандартов, организовать процесс разработки; как, учитывая условия сложившегося рынка программных продуктов (ПП), обеспечить требуемый уровень рентабельности своего проекта.

И заказчиков, и разработчиков всегда волнуют проблемы экономической эффективности проекта, его экономика, поиск путей сокращения расходов, увеличения доходов и снижения рисков. Особую актуальность в последнее время приобретает задача правовой защиты интеллектуальной собственности на разрабатываемые программные системы, права и обязанности разработчика и заказчика, нормы и методы защиты интересов каждой из сторон, возможные последствия (наказания) при нарушении законов.

Все эти вопросы в разной степени детализации поднимаются и обсуждаются в данном учебном пособии.

В первом разделе программное обеспечение рассматривается как специфический продукт интеллектуального труда; раскрываются проблемы рынка ПО, с которыми сталкиваются заказчик и разработчик; отражается роль государства как гаранта цивилизованных взаимоотношений всех участников рынка. Приведены достоинства и недостатки двух возможных стратегий заказчика при создании ПО: покупать готовое (тиражное) решение или заказать новую разработку. Приводятся рекомендации заказчику со ссылками на соответствующие стандарты. Анализируются вопросы приобретения и внедрения готовых программных решений пользователями, а также проблемы успешной организации процессов создания и внедрения ПО при его разработке «под заказ».

Во втором разделе описываются наиболее часто используемые зарубежные и отечественные стандарты, регламентирующие процессы жизненного цикла разработки, документирования и оценки качества ПС и информационных технологий. В последнее время эти вопросы все чаще ставятся заказчиком при заключении договоров как на поставку, так и на приобретение, адаптацию и внедрение программных систем. В разделе излагаются основы законодательной базы в области правовой защиты баз данных и программ для ЭВМ: рассматриваются особенности программных продуктов как объектов интеллектуальной собственности; вопросы авторского права, в том числе содержание авторского договора; вопросы регистрации и сертификации программ для ЭВМ, юридической ответственности за правонарушения.

В третьем разделе поднимаются вопросы оценки трудозатрат на создание и внедрение программного обеспечения. В настоящее время для этих целей часто используется экспертная оценка трудозатрат непосредственно в человеко-месяцах, результаты которой не всегда позволяют убедить заказчиков. В пособии приводятся следующие альтернативные методы оценки трудозатрат: прямой метод, основанный на определении размерности ПС в строках исходного кода; методика определения трудозатрат на создание программ для ЭВМ в зависимости от сложности конкретного бизнес-процесса (подсистемы АСУ); метод функциональных точек, позволяющий оценивать трудозатраты, исходя из количественных оценок, реализуемых функциональных возможностей ПС; методика определения трудозатрат, основанная

на определении размерности и сложности баз данных. Приводится распределение трудозатрат по основным этапам жизненного цикла программной системы, структура договорной цены на проект.

В четвертом разделе описываются вопросы экономики создания и внедрения программных систем и информационных технологий. Раскрываются трудно формализуемые проблемы оценки экономической эффективности проектов, приводятся формулы для расчета экономической эффективности внедрения ПС и ИТ как в сфере производства, так и в сфере управления. Для произведения расчетов приводятся статистические оценки влияния информационных технологий на изменение основных факторов производства, описываются показатели и методы их расчета, позволяющие оценить программную систему с точки зрения инновационного проекта, приводится анализ достоинств и недостатков каждого из показателей. Рассматриваются вопросы экономики создания и внедрения ПС: выделяются основные факторы, влияющие на доход, затраты и риски разработчика; приводятся практические рекомендации по оптимизации данных факторов; излагаются вопросы определения рыночных оценок тиражирования ПО; приводятся формулы для расчета объема продаж и рыночной цены на программный продукт, обеспечивающие требуемый уровень его рентабельности.

В заключении кратко освещены вопросы по управлению лицензиями.

В пособии часто встречаются такие выражения как программное обеспечение (ПО), программная система (ПС), программный продукт (ПП), комплекс программ (КП), информационная технология (ИТ). Под информационной технологией понимается упорядоченная совокупность взаимосвязанных этапов технологического процесса обработки информации, а также методы и средства (ПО, ПС, ПП, КП) обработки на каждом из этапов [1]. В этом случае программное обеспечение необходимо рассматривать как средство, обеспечивающее функционирование информационных технологий, а программную систему, программный продукт, комплекс программ — как понятия, раскрывающие конкретное содержание ПО.

1. РЫНОК ПРИКЛАДНОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

1.1. Состояние и проблемы рынка прикладного программного обеспечения

Рынок программного обеспечения и информационных технологий — это наличие товара, цивилизованные взаимоотношения разработчиков (продавцов) и заказчиков (покупателей), гарантии государства по защите как разработчиков, так и заказчиков.[1]

Понятие «рынок» непосредственно связано с понятием «товар», а товаром, как известно, является не просто какая-то вещь, а вещь, предназначенная для продажи. Соответственно, видны и три группы действующих участников рынка: производители (разработчики), продавцы (поставщики) и покупатели (пользователи). С этой точки зрения рынок программного обеспечения (ПО) должен однозначно отождествляться с рынком легального распространения программ, причем очевидно, что в условиях российской действительности этот рынок затрагивает только относительно небольшую часть общей массы программных средств, имеющихся у пользователей компьютеров. Тем не менее, принимая во внимание существование даже среди «нелегалов» каких-то минимальных товарно-денежных отношений, предлагается далее рассматривать **рынок программного обеспечения** как совокупность всех программных продуктов и их пользователей, отдельно выделяя в нем его «легальную» составляющую.

На сегодняшний день основной направленностью абсолютного большинства поставщиков ПО является работа с «крупными заказчиками» (банками, правительством, Думой и пр.). Необходимо отметить, что на отечественном рынке в настоящее время почти полностью отсутствуют мелкие производители программных средств, которые во всем мире заполняют практически все ниши потребительского спроса. При этом суммарные объемы их продаж занимают значительное место.

[1] Здесь и далее ремарки автора

Для анализа рынка ПО представляется целесообразным рассмотреть его структуру с точки зрения собственно программных средств. В этом плане можно воспользоваться следующей классификацией программных продуктов [2]:

- внутрифирменные программные средства;
- коммерческие продукты специального применения;
- коммерческие продукты широкого применения.

Внутрифирменное ПО разрабатывается, как правило, по специальным заказам собственными или сторонними программистами. Именно к данной группе программных продуктов относится часто используемый в последнее время термин «заказное ПО».

Коммерческие продукты специального применения предназначены для использования ограниченным кругом пользователей в определенных предметных областях (издательские системы, научные пакеты и пр.).

Принципиальным отличием между программными продуктами двух последних групп является способ их распространения: ПО широкого применения изначально ориентировано на использование разветвленной сетью потребителей, и в этом плане его можно охарактеризовать как «коробочное» ПО; специальное ПО распространяется, прежде всего, самими разработчиками (для России это особенно характерно), и только самые лучшие его образцы посредством фирм-посредников.

Отсутствие легального рынка, продолжающееся пиратское копирование, а возможно, и расширение его объемов наносят сильный удар по сегменту разработчиков коммерческих продуктов широкого применения. Более того, сейчас можно говорить о том, что российских производителей таких продуктов практически нет. Результатом сложившейся на рынке ПО ситуации совершенно естественными становятся попытки некоторых программистских фирм выйти на международный рынок, минуя российский, несмотря на то, что это задача достаточно сложная в плане разработки программ.

В непростом положении находятся и разработчики сектора специализированных программных продуктов. Эти продукты в меньшей степени подвержены риску пиратского размножения, поскольку будущий пользователь, как правило, заинтересован

в технической поддержке авторов продукта. Но довольно часто встречается другая проблема: пользователь не может купить нужный продукт по той простой причине, что «забыл» заложить в смету своих работ стоимость программных продуктов или не может объяснить своему руководству необходимость покупки программы, когда ее можно просто скопировать. Этот сектор рынка менее всего насыщен контрафактной продукцией. Основным проблемам внутрифирменного (заказного) программного обеспечения посвящен подразд. 1.2.

Импорт зарубежных программных продуктов развит в настоящее время довольно слабо. Причины этой ситуации совершенно ясны: мелким производителям ПО достаточно сложно выйти на российский рынок по экономическим причинам («хлопот очень много и заниматься поставкой отдельных коробок никто не хочет»). Но сейчас часто встречается и другая ситуация, когда разработчик из США просто отказывается продавать свой продукт, узнав, что его покупатель живет в России: доход от отдельной проданной «коробки» небольшой, а потери могут быть весьма значительные, так как пиратское тиражирование просто закроет для него перспективы дальнейшего сотрудничества.

Таким образом, дешевизна программных продуктов на отечественном рынке невыгодна всем участникам рынка: кто-то теряет работу; кто-то переплачивает за заказные программы, вместо того чтобы купить готовые; кто-то не может решить свои прикладные задачи и пр. Причина всего заключается в том, что в основе развития отечественного программного рынка пока еще нет его самого главного стимула развития — нормальных рыночных отношений [3, 4].

Участниками рынка программных продуктов выступают государство как гарант соблюдения законности, заказчики и потребители, производители (разработчики), партнеры, конкуренты (рис. 1.1) [5].

Рынок прикладного программного обеспечения существует при соблюдении следующих условий:

- наличие реальной потребности у конкретных заказчиков;
- наличие конечных продуктов у производителей, ориентированных на удовлетворение потребностей в прикладном ПО;

- развитая сеть посредников между производителями и потребителями;
- наличие экономических и организационно-правовых механизмов, регламентирующих цивилизованное взаимодействие участников.

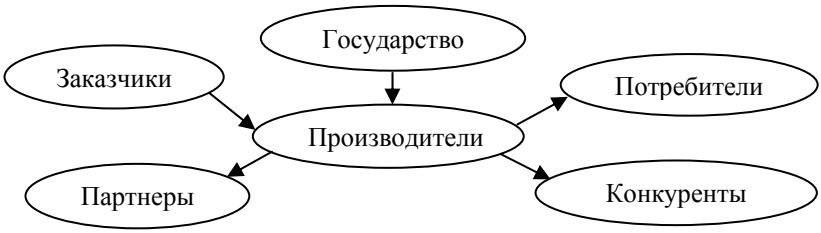

Рис. 1.1. Участники рынка прикладного программного обеспечения

Рассмотрим последовательно существующие проблемы развития рынка с точки зрения каждого из его участников.

Регулирование рынка прикладного программного обеспечения ***со стороны государства*** в настоящее время практически отсутствует. Имеющиеся законы об охране авторских прав, защите интеллектуальной собственности, информации, информатизации и защите информации не работают, так как нет эффективных механизмов их конкретного применения. В этой связи процветает компьютерное пиратство, рынок заполнен нелегальными копиями программных продуктов.

Существующая система нормативных документов (ГОСТов), регламентирующих жизненный цикл проектирования и документирования программных средств, морально устарела и носит рекомендательный характер. Сертификация как институт, обязывающий создавать программные продукты с определенными параметрами качества, существует преимущественно в добровольной форме, не носит масштабного характера, и, как следствие этого на рынке зачастую появляется некачественная, плохо документированная программная продукция. Соответствие нормативным документам и сертификатам должно оговариваться в договорах на разработку, адаптацию или поставку программных систем, о чем пользователь (потребитель) зачастую не информирован. Заказчики и потребители в большинстве своем не знают о

существовании таких документов, а государство никак не регулирует эти процессы.

Крупные потребители сориентированы в основном на иностранные программные продукты либо на удовлетворение своих информационных потребностей через создание собственных структур, занимающихся разработками программных систем. Мелкие и средние предприятия, не имеющие по экономическим соображениям возможности содержать штат высококвалифицированных программистов, сталкиваются при попытке выхода на рынок со следующими проблемами:

- спрос на программное обеспечение, как правило, не сформирован, нет четкого представления о технологии использования программных продуктов в практической деятельности;
- потребители слабо представляют себе рынок предлагаемого программного обеспечения, не способны четко сформулировать требования к приобретаемым программным продуктам, при их выборе по критерию «цена» или «качество» предпочтение отдают первому;
- незнание, а чаще всего игнорирование экономических и нормативно-правовых механизмов цивилизованной работы на рынке;
- скрытое противостояние программистов потребителям рыночных программных продуктов, связанное с потерей собственного имиджа;
- тяжелое финансовое положение, большое несоответствие между высокими ценами на программное обеспечение и сиюминутными «выгодами» от его использования, и как следствие, массовое использование нелегальных копий;
- ментальность отечественного потребителя, не расценивающего факт использования нелегальных копий как хищение собственности производителя.

Положение ***производителя*** на рынке определяется его конкурентоспособностью. Обычно рассматриваются пять факторов, влияющих на конкурентоспособность:

1) цена на программную продукцию или услугу;

2) качество продукции с точки зрения удовлетворения потребностей потребителя;

3) отличительные особенности продукции, побуждающие покупателя приобретать именно данный программный продукт;

4) гибкость производителя, связанная со способностью реагировать на просьбы покупателя по адаптации либо доработке программных систем (ПС);

5) время (сроки) реакции производителя на потребности покупателя (например, время адаптации и внедрения ПС, технология обучения пользователей, сроки гарантийного сопровождения, условия модернизации и поставки новых версий и т. д.).

С этой точки зрения основными факторами, препятствующими развитию рынка, являются:

- ориентация производителей на мелкосерийное производство ПС, как правило, разрабатываемых под конкретный заказ;
- высокая доля фиксированных затрат в структуре издержек, и как следствие, высокие цены на создаваемые ПС (как правило, это цена разработки);
- использование при разработке пиратских инструментальных программных средств, не позволяющее производителю открыто рекламировать свои продукты, участвовать в выставках и т. д.;
- отсутствие начального капитала на развитие фирмы, наработку требуемых заделов, приобретение лицензионного программного обеспечения;
- слабое использование индустриальных методов группового проектирования ПС (как правило, разработчик сам находит заказ, разрабатывает, тестирует и документирует программы);
- слабое представление о существующем рынке конкурирующих ПС;
- отсутствие эффективных программных средств защиты от копирования, а также экономических и юридических механизмов, препятствующих этим процессам;
- неумение представить ПС в виде законченного продукта и организовать маркетинг по его тиражированию; слабая и неэффективная рекламная кампания, отсутствие профессиональных менеджеров по продвижению программных продуктов на рынок ПО;
- незнание или несоблюдение отечественных и международных стандартов на управление жизненным циклом, качеством и документированием ПС.

Партнерами фирмы, специализирующейся на рынке прикладного программного обеспечения, являются соисполнители (подрядчики) по разработке отдельных компонентов ПС и фирмы, специализирующиеся на рекламе, организации выставок, продаже (тиражировании) программных систем.

Взаимодействие производителей ПС с соисполнителями строится, как правило, на договорной основе, при этом все права на дальнейшее использование результатов принадлежат производителям. Это обстоятельство предъявляет определенные требования к форме договора, спецификации продукции, ее оформлению и документированию, передаче прав собственности.

Производить качественные и эффективные рекламные материалы с привлечением профессиональных рекламных компаний могут позволить себе только крупные организации, продукция которых и так известна на рынке программных средств. «Самодельная реклама», как правило, описывает функциональные возможности ПС, излагается на «языке» разработчика и мало ориентирована на потребителя. Отсутствие требуемых финансовых средств на проведение рекламной кампании либо их экономия пагубно влияют на продвижение ПС на рынок. Аналогичная ситуация складывается и при участии производителей в выставочно-ярмарочной деятельности. С одной стороны, выставки посещают преимущественно разработчики ПС, а с другой — цены на участие в таких мероприятиях часто бывают неприемлемыми для мелких и средних производителей. Кроме того, использование при разработках нелицензионного ПО в принципе делает невозможным цивилизованное участие в выставках.

Слабо развита в настоящее время и торговая сеть по распространению ПС. Крупные организации реализуют эти функции через сеть своих филиалов. Реализация же ПС через сеть специализированных магазинов требует их хорошей рекламы, документирования, упаковки и дальнейшего сопровождения.

Обобщая вышеизложенное, следует выделить наиболее перспективные направления развития рынка прикладного ПО:

- усиление роли государства в части нормативно-правового регулирования рынка, финансовой поддержки небольших коллективов в виде получения грантов, льготных кредитов, налоговых льгот на начальных этапах развития фирм;

- повышение профессионального уровня фирм-разработчиков в области индустриального проектирования, менеджмента, управления качеством на всех этапах жизненного цикла программного продукта;
- развитие системы сертификации прикладных программных систем на соответствие отечественным и международным стандартам;
- введение сертификации ПС в качестве обязательного условия при финансировании работ из бюджетов всех уровней;
- повышение качества и эффективности рекламной деятельности, публикация материалов в популярных компьютерных журналах, участие в специализированных выставках-ярмарках, создание электронных каталогов и их рекламу в INTERNET; формирование потребности у пользователей в приобретении качественной, сертифицированной продукции;
- развитие сети оптовой торговли прикладным программным обеспечением через сеть филиалов, работающих с фирмой-производителем на контрактной основе; создание специализированных магазинов, компьютерных фирм, занимающихся поставкой техники, путем включения прикладного программного обеспечения в базовый комплект поставки.

1.2. Стандартный и индивидуальный подходы к созданию программного обеспечения

«Железо» приобретено, системное ПО установлено — пора подумать и о прикладных программах под нужды собственного бизнеса. Заказать ПО либо купить на рынке готовое? — вот в чем вопрос.

Процессы разработки *рыночного* (тиражного) и *заказного* программного обеспечения соответствуют *стандартному* и *индивидуальному* подходам к созданию ПО. Проблема выбора заказчиком между стандартным и индивидуальным ПО при информатизации своих бизнес-процессов стояла и будет стоять еще долгое время.

Безусловно, любой способ создания (приобретения) прикладного ПО таит в себе свои риски. Всегда есть риск купить типовой пакет прикладных программ, долго пытаться его освоить, но в конечном итоге остаться с ним, как с «чемоданом без ручки» (использовать тяжело и отказаться жалко). С другой стороны, имеется риск разработать или заказать индивидуальное прикладное программное обеспечение, работающее с ошибками, непригодное для сопровождения и не соответствующее техническим и отраслевым стандартам. Кроме того, возможен риск затянуть проект или попасть в слишком опасную зависимость от разработчиков (как внешних, так и внутренних). Понятно, что для исключения таких ситуаций необходим качественный анализ собственных потребностей в информатизации, состояния и возможностей рынка поставщиков. Кроме того, при выборе способов создания прикладного ПО очень важно понять особенности своего предприятия и тенденции его развития.

Объективная необходимость заказного программирования существовала всегда. Если раньше это можно было объяснить отсутствием на рынке готовых проектных решений, то сегодня рынок информационных технологий достаточно насыщен, продуктовые линейки вендоров постоянно расширяются, усложняется функциональность программных продуктов. Вместе с тем можно выделить ряд ситуаций, побуждающих заказчиков создавать индивидуальное ПО.

1. Разработка индивидуального программного продукта под собственный бизнес. К этому варианту прибегают в случаях, когда бизнес-задачи предприятия настолько уникальны и масштабны, что использование готового тиражируемого решения оказывается попросту невозможным. Особенно это касается проектов в крупных госструктурах. Понятно, что в таком случае необходимо обращаться к профессионалам — разработчикам и системным интеграторам, обладающим необходимыми для этого ресурсами и опытом.

2. Расширение функциональности используемого ПО и его кастомизация[1] на базе существующей платформы. В этом

[1] Процесс настройки компьютерной системы под нужды пользователя (потребителя); адаптация продукта под требования конкретного клиента.

случае предприятие стремится сохранить уже вложенные средства в создание информационных технологий и обойтись минимальными затратами, модернизируя отдельные участки инфраструктуры информатизации.

3. Создание заказного программного обеспечения на основе готовых модулей (компонентов), объединенных на базе сервисно-ориентированных архитектур. Готовые компоненты или модули позволяют создать некую вертикаль автоматизированных бизнес-функций, критически важных для основного бизнеса компании и позволяющих объединить их на одной платформе.

Сейчас многим стало понятно, что для создания индивидуального ПО гораздо удобнее и дешевле использовать готовую прикладную платформу и готовых специалистов, разбирающихся в ней, но при этом производить ее доработку на протяжении всего срока владения. Естественно, при таком подходе к развитию системы встает вопрос, разрабатывать ПО собственными силами либо привлечь для этих целей внешнюю организацию. В табл. 1.1 представлены преимущества и недостатки вариантов реализации заказного программного продукта собственными силами и с помощью специалистов внешней организации.

При наличии сильной команды собственных разработчиков большие риски могут возникать при длительном сопровождении и развитии системы, сделанной даже на основе добротной прикладной платформы-конструктора. При масштабных, а главное глубоких, затрагивающих функции ядра системы доработках, существует риск остаться без помощи «большого брата» в лице разработчика продукта, т. е. практически без перспектив перехода на новые версии. Отчасти этот риск можно снизить за счет приобретения качественного прикладного ядра и строгого запрета на его изменения в ходе самостоятельного развития системы.

Создание индивидуального ПО силами собственных программистов неизбежно повышает риски зависимости проекта от коллектива разработчиков. Как правило, этот коллектив ориентирован на конкретную программную платформу реализации, производственная текучка и выделенный бюджет не позволяют им изучать новые программные продукты и системы и экспериментировать с ними.

Таблица 1.1

Сравнительный анализ вариантов организации «заказного» программирования

Вариант	Характеристика варианта	
	Преимущества	Недостатки
1. Реализация проекта полностью собственными силами	1. Меньшие финансовые затраты. 2. Знание бизнес-процессов. 3. Независимость на этапе эксплуатации	1. Требуются специалисты с хорошим знанием программного продукта. 2. Требуются дополнительные штаты программистов. 3. Требуется разработка методологии управления проектом и строгое ее выполнение. 4. Необходимость решения вопроса дальнейшей занятости сотрудников, выделенных (или нанятых) для реализации проекта
2. Реализация проекта (или его этапов) «под ключ» силами внешней компании	1. Разработанная и обкатанная методология внедрения. 2. Опыт внедрения системы на нескольких предприятиях. 3. «Новый взгляд» на задачи предприятия. 4. Способность оказания услуг в области оптимизации системы управления, владение современными методами построения систем управления. 5. Знание программного продукта. 6. Штат опытных программистов	1. Большие финансовые затраты. 2. Сторонние специалисты не знают особенностей конкретного предприятия, и им требуется время на их изучение. 3. Проблема поддержки системы на этапе эксплуатации

Высокий уровень заработной платы программистов и текучесть вынуждают руководителей фирм постоянно увеличивать бюджет (затраты) на модернизацию и сопровождение ПО. Сама модернизация зачастую сводится к улучшению пользовательских интерфейсов и расширению перечня статистических отчетов. Кроме того, коллектив разработчиков, являясь монополистом, для сохранения своего имиджа часто прибегает к необоснованной критике других проектов, что естественно исключает рыночное обоснование истинной стоимости проекта.

Преимущества индивидуального (заказного) ПО обусловлены следующими факторами:

- требования индивидуализации ПО усиливаются растущей изменчивостью внешней и внутренней бизнес-среды предприятий;
- динамика изменений условий и целей предприятий приводит к тому, что вовремя и с приемлемой стоимостью модифицировать и развивать ПО во многих случаях могут именно собственные IT-службы компаний-пользователей;
- в использовании типового («стандартного») прикладного ПО во многих случаях возникает немало проблем: слишком высокая стоимость приобретения, внедрения и эксплуатации; большие, иногда непредсказуемо долгие сроки внедрения; высокая сложность или невозможность реализации многих индивидуальных требований заказчика;
- оценивание затрат, качества и рисков при создании и развитии индивидуального (заказного) прикладного ПО в последнее время радикально улучшилось за счет возможностей нового поколения ПО, так называемых прикладных платформ и систем (конструкторов приложений).

Основные проблемы эффективного использования готового тиражного ПО заключаются в следующем:

1) разнообразие стоящих перед компаниями задач столь велико, что не позволяет продолжать унификацию программных систем;

2) гибкость программных решений приобретает все большее значение, между тем как унифицированные решения практически не бывают гибкими;

3) настройка бизнес-процессов компаний на бизнес-процессы, заложенные в готовом программном решении, затратна, дорогостояща и редко заканчивается успехом;

4) потребности бизнеса все чаще вынуждают предприятия отстаивать индивидуальность своих программных решений;

5) готовые решения позволяют предприятию создать «кусочную автоматизацию», но не позволяют построить полноценную архитектуру программных систем.

Таким образом, выбирая пути построения и развития ПО надо учитывать множество взаимосвязанных условий и вероятных последствий — политических, экономических, кадровых и технических. Не предлагая конкретных рекомендаций в пользу рыночного либо заказного ПО, рассмотрим особенности разработок для каждого подхода по следующим критериям:

- денежные и временные затраты;
- функциональные возможности;
- качество, сервисы и поддержка пользователей;
- безопасность инвестиций;
- затраты на сопровождение.

1. Денежные и временные затраты

Одним из главных аргументов в пользу стандартного ПО при разработке информационных систем является стоимость проекта в целом, когда рыночная цена программного продукта в несколько раз меньше стоимости разработки. Вместе с тем при приобретении лицензий на стандартное программное обеспечение необходимо помнить следующее:

- стандартное ПО, как правило, не покрывает все бизнес-процессы компании и возникает необходимость его доработки;
- лицензия обычно продается на одно рабочее место;
- часто лицензионные модели компаний не позволяют клиенту приобретать только тот функционал, который ему нужен, и приходится платить за продукт в целом;
- ежегодные отчисления за поддержку лицензий доходят до 25 % от первоначальной цены;
- рыночная стоимость специалистов по внедрению постоянно растет, это ведет к росту затрат на стандартное ПО.

Что касается временных затрат, то использование стандартного ПО конечно же значительно сокращает жизненный цикл разработки. Вместе с тем следует отметить, что использование при индивидуальном проектировании индустриальных методов разработки и множества готовых программных компонентов

позволяют разработчику также существенно сократить жизненный цикл разработки индивидуального ПО.

2. Функциональные возможности

Традиционное преимущество стандартного ПО — широкий спектр функций, позволяющий закрывать любые потребности бизнеса. Вместе с тем это достоинство может превратиться в недостаток. Зачем заказчику избыточные возможности ПО, за которые он должен платить, но никогда не будет их использовать? Индивидуальное ПО, наоборот, создается для конкретного заказчика и в соответствии с его требованиями, и оно обладает именно теми возможностями, которые необходимы пользователю.

3. Качество, сервисы и поддержка пользователей

Несомненным преимуществом использования стандартного ПО является качество разработки, которое гарантировано массовым его внедрением в различных предметных областях. Все используемые сервисы, как правило, стандартизированы, что обеспечивает сопровождаемость ПО со стороны производителя, поддержку пользователей осуществляют множество посреднических фирм. При индивидуальном проектировании все эти проблемы возлагаются на фирму-разработчика. В этом случае заказчику необходимо тщательно продумать технологию размещения торгов на разработку индивидуального ПО и правила (критерии) выбора фирмы-победителя.

4. Безопасность инвестиций

Стандартное ПО приобретается в крупной компании, которая существует на рынке много лет, у нее множество клиентов и большие объемы продаж. В этом случае инвестиции заказчика защищены положительной историей успеха фирмы-поставщика.

При индивидуальном проектировании существует вероятность банкротства фирмы-разработчика. В этом случае целесообразно вести поэтапное проектирование, документирование и внедрение системы с выделением очередности приемки-сдачи каждого этапа работ.

5. Затраты на сопровождение

При сопровождении стандартного ПО заказчик связан с вендорами и их представителями, и следовательно, со многими не всегда контролируемыми соглашениями о качестве предоставляемых услуг. Понятно, что при использовании индивидуального

ПО риски зависимости от разработчика также возрастают. Но, во-первых, они могут быть снижены за счет создания базы знаний сопровождения продукта, а во-вторых, они взаимоувязаны с условием договора. Другой проблемой сопровождения ПО является конфликт интересов между разработчиком (продавцом) и заказчиком (покупателем). Первый заинтересован в увеличении количества продаж, что крайне отрицательно сказывается на качестве конкретных проектов. Второй же заинтересован в качестве проекта и его постоянном развитии, что, в свою очередь, крайне отрицательно сказывается на проектном бюджете. В этом случае при малом объеме продаж данный вид услуги становится разработчику экономически не выгоден, и он теряет интерес к заказчику. Выходом из создавшегося положения является поставка заказчику вместе с системой группы молодых специалистов по ее сопровождению.

Развитие компании предполагает изменение состава и содержания ее бизнес-процессов, что влечет изменение функциональной структуры программной системы. Вместе с тем трудоемкость работ по изменению (сопровождению) программного обеспечения и, соответственно, их стоимость меньше стоимости базового варианта поставки ПО. Постоянное изменение бизнес-процессов компании объективно требует и изменения функциональности тиражного ПО. Выходом из создавшегося положения является переход от трехзвенной распределенной архитектуры к ***прикладным платформенным архитектурам.***

Под ***прикладной платформой*** понимается программное обеспечение, предназначенное для проектирования, разработки, выполнения и модернизации программных компонентов, автоматизирующих деятельность различных конкретных предприятий. Иными словами, прикладная платформа позволяет собрать программную систему «под себя». Причем с помощью прикладных платформ можно создавать системы, которые поддаются изменениям в соответствии с переменами в бизнесе компании. Прикладная платформа неразрывно связана со всеми этапами жизненного цикла созданного программного обеспечения (в том смысле, что без нее программное обеспечение не может эксплуатироваться) и обычно сама состоит из разнообразных взаимосвязанных компонентов. Это качество обеспечивает широкие

возможности по развитию созданного на ее основе ПО. Прикладная платформа позволяет добавлять программные компоненты, переписывать и развивать их, что обеспечивает эффективную структуру совокупной стоимости владения, когда на первом этапе можно ограничиться относительно небольшими затратами на закупку собственно прикладной платформы, разработку и внедрение ядра системы, а затем развивать базовое решение.

Таким образом, сегодня любая тиражируемая программная система, если она претендует на роль индустриального продукта, обязана существовать в двух ипостасях — как готовый продукт и как прикладная платформа для создания программных систем поддержки основных бизнес-процессов организации. При этом выбрать прикладную платформу оказывается еще труднее, чем определиться с тиражируемым программным решением. Дело в том, что здесь появляется еще одна степень свободы, связанная со специфическими свойствами прикладных платформ, описанными выше, и прежде всего с их зрелостью.

1.3. Рекомендации по управлению жизненным циклом заказного и тиражного программного обеспечения

1.3.1. Разработка программного обеспечения «под заказ»

Создание ПО — трудно формализуемый процесс, каждый программист индивидуален, но сила — в команде.
Планировать нужно все, что не поддается планированию

Описанные ниже рекомендации предложены и использованы Дж. Маккартни при разработке приложений в среде MS + 4.0 для продукта Microsoft Solution Framework, созданного фирмой Microsoft. Рекомендации объединены в три раздела правил: «Выпустить», «Лучший проект», «Выпустить точно в срок» [6].

Раздел 1. «ВЫПУСТИТЬ»

1. ***Вы не знаете того, чего вы не знаете.*** В любом проекте всегда существуют неопределенности, но человеку свойственно

отвергать свое незнание, подменяя истинное знание предположениями. Неизвестно, что является ошибкой, поэтому старайтесь быть критичными к себе. Найдите для своей команды мотивацию к тому, чтобы она постаралась обнаружить и понять максимум «белых пятен». Не пытайтесь обмануть себя предположениями.

2. ***Старайтесь иметь четкое представление о состоянии проекта.*** В проекте должны участвовать люди, способные объективно оценить готовность системы. Разработчики всегда более оптимистичны, чем тестировщики. Поощряйте плохие новости, и тогда снизится риск неожиданного провала в самом конце проекта. Самый лучший способ — полагаться на объективные данные, метрики (степень покрытия функциональности тестами, число активных дефектов, динамика их исправления и т. д.). Состояние осведомленности обязательно должно включать знание обо всех составляющих проекта и содержать точную информацию о статусе проекта в определенный период времени.

3. ***Помните о треугольнике приоритетов.*** Существуют три взаимосвязанных компонента любого проекта: ресурсы (люди и деньги), функциональность и сроки. Изменение одного из них всегда влияет на остальные. Можно зафиксировать значения только двух из этих показателей: например, зафиксировать характеристики времени и команды, чтобы получить требуемую функциональность; зафиксировать характеристики функциональности и команды, чтобы получить нужное время; зафиксировать характеристики времени и функциональности, чтобы получить требуемую численность команды разработки. Не забывайте, что не все задачи можно выполнять параллельно.

4. ***Старайтесь быть на виду***. При планировании надо разбивать задачи на более мелкие, на выполнение которых требуется полдня или день. Тогда о том, что вы начинаете отставать, вы узнаете достаточно рано и сможете предпринять корректирующие шаги. Неделя для проекта разработки — это вечность. Каждый проект, который отстает на месяц, вначале отставал на один день. Узнайте об отставании прежде, чем наступит момент, когда исправить что-то будет уже невозможно. Несомненно, подобный стиль управления весьма опасен и периодически вас будут в нем обвинять. Но если вам удается довести до участников

5. ***Используйте контрольные точки с отсутствием дефектов***. В каждом приложении можно выделить 20 % функциональности, которая будет использоваться 80 % времени; в свою очередь, в оставшихся 80 % функциональности можно опять выделить 20 % и т. д. Реализуйте сначала первые 20 % функций, но полностью, а также протестируйте и соберите программу установки с прототипом документации. И только когда все это будет готово, двигайтесь дальше. Этот метод основан на идее выделения контрольных точек, в которых продукт должен содержать некое, не большое, но четко определенное количество дефектов. Их промежуточный контроль позволяет быстро понять, какие части проекта могут стать причиной проблем, и получить полную картину о проекте, а также дают возможность сфокусироваться на достижении целей каждой контрольной точки.

6. ***Бойтесь разработчиков, сидящих в башнях из слоновой кости***. Разработчики должны уметь работать в команде, демонстрировать свой прогресс, делиться опытом и проверять то, что уже сделано. Избегайте «примадонн», которые считают себя умнее всех. Несомненно, проект лишь выиграет, если в команде появится пара гениев или просто отличных специалистов, но еще лучше будет, если их талант признает и команда, и все лица, заинтересованные в результатах проекта. Разработка программ осуществляется силами команды, для определения состава которой может пригодиться правило, принятое в Microsoft, — шесть разработчиков, три инженера по качеству, один менеджер, два технических писателя. Это правило — результат многочисленных проб и ошибок, через которые прошла корпорация при разработке и масштабных платформ, и совсем небольших утилит.

7. ***Плохая дата — не просто плохая дата***. Обычно вы заранее знаете, что опоздаете, знают это и все вокруг. Вы настаиваете на смене даты, но с каждой отсрочкой теряете доверие клиента. Вот хорошее правило: перенос даты должен происходить только в тех случаях, когда точно известны все составляющие и причины задержки. Изменение сроков требует ресурсов, поэто-

му следующая контрольная точка должна быть реалистичной. В противном случае подобный «проект» никогда не закончится.

8. ***Сдвинув сроки, не проваливайте их***. Перенос срока — это симптом, который свидетельствует о выявлении «белых пятен». Такое случается весьма часто, поскольку разработка ИТ-решений является экспериментальным видом деятельности, в котором используются новые технологии. Все это ведет к неопределенности по поводу сроков проекта. Если сдвиг сроков выполнения проекта стал неожиданностью, значит используемые методы общения с сотрудниками и управления командой проекта надо менять. В то же время задержка дает возможность переоценить ресурсы и возможности продукта; в итоге это приносит положительный эффект. Поэтому, сдвигая сроки, внимательно анализируйте ошибки и старайтесь их больше не допускать.

9. ***Чем проще — тем лучше***. Лучше обещать меньше, но сделать больше, чем пообещать больше и не выполнить. Для заказчика важнее результат, а не обещания. А для успеха проекта стоит выбирать максимально простые, но надежные решения.

10. ***Время для проектирования — это время для проектирования***. Оценивая способы проектирования, всегда учитывайте временной фактор и выбирайте из альтернативных решений наименее рискованное и оптимальное по времени реализации. Часто в погоне за красотой реализации, проектировщик склонен выбрать вариант, совершенно не укладывающийся в сроки проекта. Помните, что время — весьма ограниченный ресурс, который дан для того, чтобы достичь успеха, а не удивить.

11. ***Если вы не можете что-то собрать, значит, вы не сможете это выпустить.*** Продукт должен постоянно собираться (компилироваться вместе со всеми компонентами системы, документацией и программой установки). Это необходимо потому, что с самого начала нужно определить все составляющие будущего продукта и адекватно оценить степень их готовности. В противном случае вы обязательно что-нибудь забудете, и это станет весьма неприятной неожиданностью, особенно перед окончанием работ. Пусть на начальном этапе это будут скромные зачатки будущей системы — придет время, и из них вырастет отличный продукт. С другой стороны, промежуточная

12. ***Мысли о многоплатформенности.*** Старайтесь «не перебарщивать» с числом платформ, на которых будет работать система. Держите в голове тот факт, что разработка для каждой из платформ зачастую представляет собой переработку большей части функциональности, а также ее тщательное тестирование и последующее исправление ошибок. Соответственно, это увеличивает как стоимость системы, так и затраты на будущую поддержку. Помните, что пользователям чаще всего нужен продукт, а не его удивительная гибкость.

Раздел 2. « ЛУЧШИЙ ПРОДУКТ»

13. ***Заказчик — это ваше все.*** Старайтесь в следующих версиях учитывать даже весьма странные, нечетко выраженные пожелания клиентов. Часто в этих требованиях скрывается то, что делает продукт уникальным и неотличимым от других, добавляет ему маркетинговой привлекательности и заставляет пользователей использовать его долгие годы. Помните, что заказчик лучше вас знает, что ему нужно.

14. ***Самое главное — единство и интеграция.*** Единство причины и единство исполнения должны стать девизами команды разработчиков.

15. ***Двигайтесь правильным курсом.*** Цель — основная идея вашей разработки. Все оценки продукта основываются именно на ней, поэтому она должна быть очень четкой. Старайтесь как можно раньше наметить цель и сохранить веру в нее вплоть до конца проекта.

16. ***Будьте гибким.*** Часто по ходу проекта требования к системе могут изменяться — будьте готовы к этому. Старайтесь постоянно проверять, насколько мнение пользователя соответствует поставленной цели. Используйте для этого промежуточные версии продукта, вовлекая заказчика в процесс работы с системой как можно раньше. Однако, собираясь менять курс, помните: цель должна остаться прежней.

17. ***Соблюдайте баланс.*** Правильно расставьте акценты на разных составляющих проекта. Ни в коем случае не увлекайтесь наращиванием свойств какой-либо из возможностей продукта,

это станет причиной дискомфорта пользователей и может привести к серьезным проблемам со сроками реализации продукта.

18. ***Развивайте продукт постепенно.*** Правильное развитие выглядит так: ранние стадии разработки определяют более поздние, ошибки не повторяются, а результат отвечает потребностям конечного пользователя. Плавное развитие вселяет в вас ощущение предсказуемости и стабильности процесса разработки.

19. ***Продукт — это иерархия компонентов.*** Следуя этому принципу, элементам проекта уделяют внимание пропорционально их важности, что обеспечивает стабильность и сбалансированное развитие. Иерархию очень удобно использовать как скелет для постепенного расширения системы, сначала вы реализуете основу, а затем наполняете ее будущим содержимым; новые компоненты опираются на уже разработанные.

20. ***Все должны разделять общее видение продукта.*** Все члены команды должны знать, какие цели должны быть достигнуты, как продукт должен выглядеть, какова стратегия его разработки. Если в команде появляются противники текущей цели, постарайтесь дать им слово и аргументировать свое видение, может быть, оно лишь улучшит будущую систему. Все противоречия должны быть разрешены, а видение предмета приведено к единству.

Раздел 3. «ВЫПУСТИТЬ ТОЧНО В СРОК»

21. ***Ваша главная задача — выпустить продукт.*** Помните:

- команда обязана поставить продукт в срок, а все члены команды должны верить в то, что это возможно;
- каждый должен понимать, что от него для этого требуется, а менеджер должен сделать все от него зависящее, чтобы иметь все шансы сделать то, что требуется;
- каждый должен не просто хотеть, а гореть желанием достичь цели.

Выпуск продукта должен стать целью каждого члена команды, самым ожидаемым событием для всех!

1.3.2. Приобретение готового программного обеспечения

Готовое ПО лучше приобретать по правилам.
Лучшие правила для заказчика (покупателя) содержатся в международных и отечественных стандартах.
Корректное использование стандартов — залог успеха в этом «абсолютно безнадежном деле».

Жизненный цикл ПО на процессы поставки и приобретения готовых компонентов и программных средств, а также организационного взаимодействия поставщиков и потребителей рассматривается в международном стандарте **ISO 12207: 1995**. **Процессы жизненного цикла программных средств**, содержание которого описывается в п. 2.1.5 учебного пособия. При этом в разделе 5.1 данного стандарта внимание акцентируется на организации экономического и технического взаимодействия покупателя и продавца готового программного продукта или сервиса в течение всего жизненного цикла программного обеспечения. Раздел 5.2 стандарта содержит основные требования заказчика к организации работ поставщика в течение всего жизненного цикла ПО и его компонентов, которые детализируются и реализуются всеми последующими разделами стандарта.

Остановимся кратко на описании содержания каждого из приведенных разделов стандарта [7].

Заказчик (покупатель) начинает процесс приобретения, описывая концепцию или потребность в приобретении, разработке или развитии системы, программного продукта или обслуживания программного обеспечения. При этом он анализирует следующие требования к системе: коммерческую направленность проекта; организацию и требования надежности, безопасности, защиты информации; другие принципиальные требования.

Покупатель рассматривает варианты приобретения программного продукта, начиная с анализа затрат и получаемой выгоды от каждого варианта до учета возможного риска. Различные варианты могут предоставлять разнообразные возможности:

а) покупку готового программного продукта, который удовлетворяет всем требованиям;

б) разработку программного продукта собственными силами и приобретение отдельных сервисных программ, расширяющих его функциональные возможности;

в) разработку программного продукта и приобретение сервисов программного обеспечения через контракт с внешней организацией;

г) комбинацию *а*, *б*, *в*;

д) модернизацию существующего программного продукта.

При покупке готового программного изделия покупатель должен получить гарантии в том, что будут удовлетворены следующие условия:

- выполнены все требования к программному продукту;
- в комплект поставки входит документация;
- соблюдены права собственности, использования, лицензирования и гарантийного обслуживания;
- обеспечена будущая поддержка программного продукта.

Покупатель должен подготовить, документировать и выполнить план приобретения, который включает:

а) требования к системе;

б) варианты и степень использования системы;

в) тип используемого контракта;

г) обязательства внешних организаций;

д) поддержку ПС;

е) учтенные риски и методы управления ими.

Вне зависимости от типа приобретаемого ПС покупатель должен задокументировать требования на приобретение (например, в виде заявки на приобретение). Документация на приобретение должна включать *требования к системе, область действия, инструкции для участников торгов, список программных продуктов, сроки и условия приобретения, контрольные сроки выполнения этапов субподрядного договора, технические ограничения.*

Процедура организации и подведения итогов торгов оговаривается соответствующими нормативными актами.

Требования на приобретение должны быть предоставлены организации, выбранной для выполнения деятельности по приобретению.

Далее покупатель должен осуществить процедуры выбора поставщика, включая критерии оценки предложений и соответствия продукта сформулированным им требованиям.

После выбора поставщика покупатель готовит и обсуждает условия контракта с поставщиком, которому адресует требования к приобретаемой ПС, включая стоимость и спецификацию программного продукта или сервиса, который должен быть поставлен. Контракт фиксирует права собственности, использования, лицензирования и гарантийные обязательства поставщика, связанные с имеющимися в наличии программными продуктами многократного использования.

Если контракт уже заключен, то покупатель последовательно контролирует изменения путем переговоров с поставщиком. Изменения в контракте используются покупателем для воздействия на проектные планы, затраты, выгоды, качество.

Процесс приобретения заканчивается приемкой программного обеспечения на основе определенных критериев и приемно-сдаточной стратегии, включающей подготовку контрольных тестовых примеров, данных, процедур и среды эксплуатации.

Покупатель проводит приемочные испытания поставляемого программного продукта или сервиса и принимает их от поставщика, когда удовлетворены все условия приемки. После принятия ПО покупатель должен взять на себя ответственность за дальнейшее использование поставляемого программного продукта.

Стандартизация процесса поставки заключается в регламентации действий и задач поставщика. Процесс может быть начат решением о подготовке предложения для ответа на запрос о приобретении или заключением и подписанием контракта с покупателем на приобретение системы, программного продукта или предоставленного сервиса.

Первоначально поставщик должен провести обзор требований в запросе о приобретении, принимая во внимание организационную стратегию покупателя, предлагаемую цену и другие правила, подготовить предложение в ответ на запрос о приобретении, включая рекомендуемые положения данного международного стандарта.

После заключения контракта поставщик должен провести оценку требований к программной системе, чтобы определить структуру управления проектом, включая гарантии выполнения проекта и гарантии качества поставляемого программного продукта или сервиса.

Кроме того, поставщик определяет или выбирает модель жизненного цикла программного обеспечения, соответствующую области применения, параметры важности и сложности проекта, если это не было отражено в контракте. Все процессы, действия и задачи этого международного стандарта должны быть выбраны и отображены в модели жизненного цикла ПС.

Поставщик должен установить требования к планам исполнения проекта, обеспечения качества поставляемого программного продукта или сервиса. Требования к планам должны включать потребности в ресурсах и возможные ограничения для пользователя.

Если требования к планированию установлены, поставщик должен рассмотреть варианты разработки программного продукта или предоставления сервиса, сопоставив их с анализом рисков, связанных с каждым из вариантов. Предложенные варианты могут содержать следующие возможности:

а) разработку программного продукта или предоставление сервиса, используя внутренние ресурсы;

б) разработку программного продукта или предоставление сервиса, заключая субподрядный договор;

в) получение готовых программных продуктов от внутренних или внешних источников;

г) комбинации *а*, *б*, *в*.

В плане управления проектом должны быть отражены следующие моменты:

1) проектирование организационной структуры, полномочий, ответственности (обязательств) каждой организационной единицы, включая внешние организации;

2) проектирование окружения (для разработки, функционирования или сопровождения), включая испытательное оборудование, библиотеки, средства, стандарты, процедуры и инструментарий;

3) разработка структуры прерывания процессов жизненного цикла и действий, включая программные продукты, персонал, материальные ресурсы;

4) разработка планов управления качественными характеристиками программного продукта или сервиса;

5) проектирование механизмов обеспечения безопасности, защиты и других критических требований к программным продуктам;

6) разработка процедур управления субподрядчиками, включая выбор субподрядчика, и решения финансовых затруднений между субподрядчиком и покупателем, если они возникают;

7) процедуры верификации, сертификации и аттестации, включая варианты связи с аттестационным агентом, если это определено в контракте.

8) варианты снятия разногласий с покупателем путем совместных оценок, проверок, неофициальных встреч, сообщений, обсуждений, модификаций и изменений;

9) управление риском, т. е. управление областями проекта, которые включают технический потенциал, стоимость и планирование рисков;

10) политика безопасности, включающая правила обязательного доступа к информации на каждом проектном уровне организации;

11) утверждение проекта, обеспечиваемое такими средствами, как процедуры закрепления необходимых прав собственности, лицензионных прав, гарантий;

12) обучение персонала.

В процессе реализации проекта поставщик должен постоянно осуществлять мониторинг и контролировать развитие и качество программного продукта или сервиса в течение всего жизненного цикла, обеспечивая при этом проблемную идентификацию отклонений от плана, запись, анализ и решение.

Кроме того, поставщик должен добиться удовлетворения всех предусмотренных контрактом требований, необходимых для гарантии того, что программный продукт или сервис, доставленный покупателю, разработан или выполнен согласно требованиям основного контракта. При этом для решения этой задачи по-

ставщик может использовать проверки, аттестацию или испытательных агентов, как определено в контракте и проектных планах. Для окончательной оценки и проверки качества проекта поставщик должен:

- координировать действия по оценке контракта и связям с покупателем;
- проводить и поддерживать неформальные встречи, приемную оценку, приемные испытания, совместные оценки, проверки с покупателем;
- выполнять верификацию и аттестацию программных продуктов или сервиса, а также демонстрацию их соответствия требованиям контракта;
- предоставлять доступ покупателю к докладам об оценке, проверке, тестировании и решении возникших проблем;
- обеспечивать покупателю доступ к оборудованию поставщиков и субподрядчиков для обзора программных продуктов или сервиса.

После поставки и оформления соответствующих документов поставщик должен обеспечить помощь покупателю в поддержке поставленного программного продукта или сервиса на условиях, предусмотренных в контракте.

В заключение отметим, что технология создания и применения мобильных прикладных программ и баз данных с использованием готовых компонентов быстро совершенствуется и в ближайшие годы станет доминирующей при создании сложных информационных систем. Это позволит повысить качество и конкурентоспособность готовых программных средств и баз данных и резко сократить трудоемкость, стоимость и период их создания.

Контрольные вопросы

1. Что надо предпринять небольшой фирме для успешной цивилизованной работы на рынке информационных технологий?
2. Как потребителю не допустить ошибок при приобретении готового программного продукта и выборе поставщика?
3. Каким образом государство должно защищать интересы разработчиков и заказчиков?

4. Как правильно выстроить процессы управления по созданию программного продукта, включая вопросы взаимоотношения с заказчиком?

5. Рынок программного обеспечения — это «рынок» или «базар»? Как активизировать разработчиков и обеспечить интересы заказчиков?

6. Преимущества и проблемы программирования «под заказ». Как убедить заказчика перейти на индивидуальное ПО?

2. НОРМАТИВНО-ПРАВОВОЕ РЕГУЛИРОВАНИЕ В ОБЛАСТИ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

2.1. Стандартизация основных этапов жизненного цикла создания программных систем и их документирования

2.1.1. Цели стандартизации

Стандарты — это благо или помехи?

Непрерывный рост масштабов проектов по созданию ПС и их необозримости для отдельного специалиста привели к необходимости совершенствования процессов проектирования, документирования и организации работ коллективов специалистов при обеспечении длительного жизненного цикла создания программных систем. Накопленный мировой опыт сосредоточен и обобщен в международных и национальных стандартах, которые почти неизвестны отечественным специалистам. Пренебрежение стандартами резко снижает конкурентоспособность отечественных программных продуктов и информационных систем, созданных даже на современной программно-аппаратной платформе.

Основными целями применения стандартов при создании программных систем являются:

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

Применение стандартов при системном проектировании ПС позволяет ориентироваться на построение систем из крупных функциональных узлов, применять достаточно отработанные и проверенные проектные решения, стандартизировать интерфейсы и протоколы взаимодействия компонентов таким образом, что разработчику системы, как правило, не требуется вдаваться в детали внутреннего устройства этих компонентов. Стандарты, относящиеся к прикладным программным комплексам (функциональным частям) ПС, облегчают повторное использование в проектируемой системе уже разработанных и проверенных прикладных программ. Таким образом, проектирование ПС в значительной степени может сводиться к ее компоновке из стандартизированных узлов.

Основу отечественной нормативной базы в области создания и документирования программных средств составляют следующие комплексы стандартов:

- «Единая система программной документации (ЕСПД)» — ГОСТ 19;
- «Информационные технологии. Автоматизированные системы» — ГОСТ 34;
- «Государственные стандарты РФ» (ГОСТ Р).

Следует отметить, что вышеперечисленные стандарты носят рекомендательный характер и, в соответствии с Законом РФ «О стандартизации», становятся обязательными ***на контрактной основе***, т. е. при ссылке на них в договоре на разработку (поставку) программных средств и автоматизированных систем.

Большая часть стандартов морально устарела, однако заказчик и разработчик могут по взаимному согласию выбрать необходимое подмножество стандартов, отвечающих специфике конкретного проекта, дополнить выбранные документы нужными разделами из других документов, исключив при этом ненужные.

2.1.2. Стандарты комплекса ГОСТ 19

Рациональное использование ГОСТ 19 — это:

- *разумный компромисс с заказчиком по этапам жизненного цикла создания ПС;*
- *оптимальное документирование*

Основная и большая часть комплекса ЕСПД была разработана в 1970–1980-е гг. В настоящее время этот комплекс представляет собой систему межгосударственных стандартов стран СНГ (ГОСТ 19), действующих на территории Российской Федерации на основе межгосударственного соглашения по стандартизации. Стандарты ЕСПД в основном охватывают ту часть документации, которая создается в процессе разработки программных средств, и связаны, по большей части, с документированием функциональных характеристик ПС. В состав ЕСПД входят:

1) ГОСТ 19.001-77 ЕСПД. Общие положения;

2) ГОСТ 19.101-77 ЕСПД. Виды программ и программных документов;

3) ГОСТ 19.102-77 ЕСПД. Стадии разработки;

4) ГОСТ 19.103-77 ЕСПД. Обозначение программ и программных документов;

5) ГОСТ 19.104-78 ЕСПД. Основные надписи;

6) ГОСТ 19.105-78 ЕСПД. Общие требования к программным документам;

7) ГОСТ 19.106-78 ЕСПД. Требования к программным документам, выполненным печатным способом;

8) ГОСТ 19.201-78 ЕСПД. Техническое задание. Требования к содержанию и оформлению;

9) ГОСТ 19.202-78 ЕСПД. Спецификация. Требования к содержанию и оформлению;

10) ГОСТ 19.301-79 ЕСПД. Порядок и методика испытаний;

11) ГОСТ 19.401-78 ЕСПД. Текст программы. Требования к содержанию и оформлению;

12) ГОСТ 19.402-78 ЕСПД. Описание программы;

13) ГОСТ 19.404-79 ЕСПД. Пояснительная записка. Требования к содержанию и оформлению;

14) ГОСТ 19.501-78 ЕСПД. Формуляр. Требования к содержанию и оформлению;

15) ГОСТ 19.502-78 ЕСПД. Описание применения. Требования к содержанию и оформлению;

16) ГОСТ 19.503-79 ЕСПД. Руководство системного программиста. Требования к содержанию и оформлению;

17) ГОСТ 19.504-79 ЕСПД. Руководство программиста;

18) ГОСТ 19.505-79 ЕСПД. Руководство оператора;

19) ГОСТ 19.506-79 ЕСПД. Описание языка;

20) ГОСТ 19.508-79 ЕСПД. Руководство по техническому обслуживанию. Требования к содержанию и оформлению;

21) ГОСТ 19.604-78 ЕСПД. Правила внесения изменений в программные документы, выполняемые печатным способом;

22) ГОСТ 19.701-90 ЕСПД. Схемы алгоритмов, программ, данных и систем. Условные обозначения и правила выполнения;

23) ГОСТ 19.781-90 Обеспечение систем обработки информации программное. Термины и определения.

Из всех стандартов ЕСПД остановимся только на тех, которые могут чаще всего использоваться на практике [1, 8].

ГОСТ (СТ СЭВ) 19.201-78 (1626-79) ЕСПД. Техническое задание. Требования к содержанию и оформлению. Техническое задание (ТЗ) содержит совокупность требований к ПС и используется в дальнейшем в качестве критерия при сдаче-приемке разработанной системы в эксплуатацию. Поэтому достаточно полно составленное (с учетом возможности внесения дополнительных разделов) и принятое заказчиком и разработчиком ТЗ является одним из основополагающих документов проекта ПС.

Техническое задание должно содержать следующие разделы:

- введение;
- основания для разработки;
- назначение разработки;
- требования к программе или программному изделию;
- требования к программной документации;
- технико-экономические показатели;
- стадии и этапы разработки;
- порядок контроля и приемки.

В ТЗ допускается включение приложений. В зависимости от особенностей программы или программного изделия допускается уточнять содержание разделов, вводить новые разделы или объединять отдельные из них.

ГОСТ (СТ СЭВ) 19.101-77 (1626-79) ЕСПД. Виды программ и программных документов. Устанавливает виды программ и программных документов для вычислительных машин, комплексов и систем независимо от их назначения и области применения. В качестве основных видов программ стандартом определяются:

компоненты — программы, рассматриваемые как единое целое, выполняющие законченную функцию и применяемые самостоятельно или в составе комплекса;

комплексы — программы, состоящие из двух или более компонентов и выполняющие взаимосвязанные функции, и применяемые самостоятельно или в составе другого комплекса.

Виды программных документов и их краткое содержание представлены в стандарте описаниями, приведенными в табл. 2.1.

Таблица 2.1

Виды программных документов

Вид документа	Содержание документа
Спецификация	Состав программы и документация на нее
Ведомость держателей подлинников	Перечень предприятий, на которых хранятся подлинники программных документов
Текст программы	Запись программы с необходимыми комментариями
Описание программы	Сведения о логической структуре и функционировании программы
Программа и методика испытаний	Требования, подлежащие проверке при испытании программы, а также порядок и методы их контроля
Техническое задание	Назначение и область применения программы; технические, технико-экономические и специальные требования, предъявляемые к программе; необходимые стадии и сроки разработки; виды испытаний
Пояснительная записка	Схема алгоритма, общее описание алгоритма и (или) функционирования программы, а также обоснование принятых технических и технико-экономических решений
Эксплуатационные документы	Сведения для обеспечения функционирования и эксплуатации программы

Перечень эксплуатационных документов, рекомендуемых ЕСПД, представлен в табл. 2.2.

Таблица 2.2

Виды эксплуатационных документов

Вид документа	Содержание документа
Ведомость эксплуатационных документов	Перечень эксплуатационных документов на программу
Формуляр	Основные характеристики программы, комплектность и сведения об эксплуатации программы
Описание применения	Сведения о назначении программы, области применения, применяемых методах, классе решаемых задач, ограничениях для применения, минимальной конфигурации технических средств
Руководство системного программиста	Сведения для проверки, обеспечения функционирования и настройки программы на условия конкретного применения
Руководство программиста	Сведения для эксплуатации программы
Руководство оператора (пользователя)	Сведения для обеспечения процедуры общения оператора с вычислительной системой в процессе выполнения программы
Описание языка	Описание синтаксиса и семантики языка
Руководство по техническому обслуживанию	Сведения для применения тестовых и диагностических программ при обслуживании технических средств

Допускается объединение отдельных видов эксплуатационных документов (за исключением ведомости эксплуатационных документов и формуляра), необходимость объединения указывается в техническом задании. Объединенному документу присваивают наименование и обозначение одного из объединяемых документов. В объединенных документах должны быть приведены сведения, которые необходимо включать в каждый объединяемый документ.

ГОСТ 19.102-77 ЕСПД. Стадии разработки. Устанавливает стадии разработки программ и программной документации для вычислительных машин, комплексов и систем независимо от их назначения и области применения (табл. 2.3).

Таблица 2.3

Стадии разработки, этапы и содержание работ

Стадии разработки	Этапы работ	Содержание работ
I ТЕХНИ-ЧЕСКОЕ ЗАДАНИЕ	Обоснование необходимости разработки программы	Постановка задачи. Сбор исходных материалов. Выбор и обоснование критериев эффективности и качества разрабатываемой программы. Обоснование необходимости проведения научно-исследовательских работ
	Научно-исследо-вательские работы	Определение структуры входных и выходных данных. Предварительный выбор методов решения задач. Обоснование целесообразности применения ранее разработанных программ. Определение требований к техническим средствам. Обоснование принципиальной возможности решения поставленной задачи
	Разработка и утверждение технического задания	Определение требований к программе. Разработка технико-экономического обоснования разработки программы. Определение стадий, этапов и сроков разработки программы и документации на нее. Выбор языков программирования. Определение необходимости проведения научно-исследовательских работ на последующих стадиях. Согласование и утверждение технического задания

Продолжение табл. 2.3

Стадии разработки	Этапы работ	Содержание работ
II ЭСКИЗНЫЙ ПРОЕКТ	Разработка эскизного проекта	Предварительная разработка структуры входных и выходных данных. Уточнение методов решения задачи. Разработка общего описания алгоритма решения задачи. Разработка технико-экономического обоснования
	Утверждение эскизного проекта	Разработка пояснительной записки. Согласование и утверждение эскизного проекта
III ТЕХНИ-ЧЕСКИЙ ПРОЕКТ	Разработка технического проекта	Уточнение структуры входных и выходных данных. Разработка алгоритма решения задачи. Определение формы представления входных и выходных данных. Определение семантики и синтаксиса языка. Разработка структуры программы. Окончательное определение конфигурации технических средств
	Утверждение технического проекта	Подготовка плана мероприятий по разработке и внедрению программ. Разработка пояснительной записки. Согласование и утверждение технического проекта

Окончание табл. 2.3

IV РАБОЧИЙ ПРОЕКТ	Разработка программы	Программирование и отладка программы.
	Разработка программной документации	Разработка программных документов в соответствии с требованиями ГОСТ 19.101-77
	Испытания программы	Разработка, согласование и утверждение программы и методики испытаний. Проведение предварительных государственных, межведомственных, приемо-сдаточных и других видов испытаний. Корректировка программы и программной документации по результатам испытаний
V ВНЕДРЕНИЕ	Подготовка и передача программы	Подготовка и передача программы и программной документации для сопровождения и (или) изготовления. Оформление и утверждение акта о передаче программы на сопровождение и (или) изготовление. Передача программы в фонд алгоритмов и программ

Примечание: допускается исключение 2-й стадии разработки, объединение 3-й и 4-й стадий, введение других этапов работ по согласованию с заказчиком. Необходимость проведения стадий разработки указывается в ТЗ.

2.1.3. Стандарты комплекса ГОСТ 34

Информационная технология — это совокупность программ, информационных ресурсов, средств вычислительной техники, персонала и сотрудников. Соблюдение ГОСТ 34 — гарантия комплексного и эффективного взаимодействия этих составляющих.

Полный жизненный цикл на создание, внедрение и сопровождение автоматизированных систем и информационных технологий, включающий в себя, несомненно, программные продукты и информационные системы, регламентирован стандартами комплекса ГОСТ 34.

ГОСТ 34 в конце 1980-х годов предполагалось создавать как всеобъемлющий комплекс взаимосвязанных межотраслевых документов. Объектами стандартизации являются автоматизированные системы (АС) различных видов и все виды их компонентов, а не только ПО и БД. Комплекс рассчитан на взаимодействие заказчика и разработчика, при этом предусмотрено, что заказчик может разрабатывать АС для себя самостоятельно, если создаст для этого специализированное подразделение. Поскольку ГОСТ 34 в основном уделяет внимание содержанию проектных документов, распределение действий между сторонами обычно осуществляется в соответствии с их содержанием.

ГОСТ 34 способствует более полной и качественной стыковке действительно разных систем, что особенно важно в условиях, когда разрабатывается все больше сложных комплексных АС, например типа CAD/CAM-систем[1], которые включают в свой

[1] CAD/CAM-системами на западе называют то, что в странах бывшего СССР принято было называть системой автоматизированного проектирования (САПР). CAD — Computer Aided Design, или Computer Aided Drafting (проектирование и конструирование с помощью ЭВМ, или черчение с помощью ЭВМ). Впервые термин CAD прозвучал в конце 50-х гг. прошлого века в Массачусетском технологическом институте в США. Распространение эта аббревиатура получила уже в 1970-х гг. как международное обозначение технологии конструкторских работ. С началом применения вычислительной техники под словом CAD подразумевалась обработка данных средствами машинной графики. CAM — Computer Aided Manufacturing. Программирование устройств ЧПУ станков с помощью CAD-систем отождествляют с понятием CAM (так называемые CAD/CAM-системы). В иных случаях под CAM понимают применение ЭВМ в управлении производством и движением материалов.

состав АСУТП (автоматизированные системы управления технологическими процессами), АСУП (автоматизированная система управления предприятием), САПР-конструктора, САПР-технолога, АСНИ (автоматизированная система научных исследований) и другие системы. Введение единой, определенной терминологии, наличие разумной классификации работ, документов, видов обеспечения безусловно полезно. Определено несколько важных положений, отражающих особенности АС как объекта стандартизации, например: «В общем случае АС состоит из программно-технических комплексов (ПТК), программно-методических комплексов (ПМК) и отдельных компонентов организационного, технического, программного и информационного обеспечения».

В отличие от ПТК, АС есть не «ИС с БД», а система, соответствующая следующим определениям:

- организационно-техническая система, обеспечивающая выработку решений на основе автоматизации информационных процессов в различных сферах деятельности (управление, проектирование, производство и т. д.) или их сочетаниях, что особенно актуально в аспектах бизнес-реинжиниринга;
- система, состоящая из персонала и комплекса средств автоматизации его деятельности, реализующая информационную технологию выполнения установленных функций.

Приведенные определения показывают, что АС, в первую очередь, характеризуется персоналом, функциями которого является принятие решений и выполнение других управляющих действий, поддерживаемых организационно-экономическими механизмами и программно-техническими средствами.

В составе ГОСТ 34 в настоящее время действуют следующие стандарты:

1) ГОСТ 34.601–90. Информационная технология. Автоматизированные системы. Стадии создания;

2) ГОСТ 34.201–89. Информационная технология. Виды, комплектность и обозначение документов при создании автоматизированных систем;

3) ГОСТ 34.602–89. Информационная технология. Техническое задание на создание автоматизированных систем;

4) ГОСТ 34.603–92. Информационная технология. Виды испытаний автоматизированных систем;

5) РД 50–34.698–90. Методические указания. Информационная технология. Автоматизированные системы. Требования к содержанию документов.

ГОСТ 34.601–90 Информационная технология. Автоматизированные системы. Стадии создания. Настоящий стандарт распространяется на автоматизированные системы, используемые в различных видах деятельности (исследование, проектирование, управление и т. п.), включая их сочетания, создаваемые в организациях, объединениях и на предприятиях (далее организациях). Стандарт устанавливает стадии и этапы создания АС, а также содержание работ на каждом этапе.

Процесс создания АС представляет собой совокупность упорядоченных во времени, взаимосвязанных, объединенных в стадии и этапы работ, выполнение которых необходимо и достаточно для создания АС, соответствующей заданным требованиям (табл. 2.4).

Допускается исключение стадии «Эскизный проект» и отдельных этапов работ на всех стадиях, объединение стадий «Технический проект» и «Рабочая документация» в одну стадию «Техно-рабочий проект». В зависимости от специфики создаваемых АС и условий их создания допускается выполнение отдельных этапов работ до завершения предшествующих стадий, параллельное выполнение этапов работ, включение новых этапов работ.

ГОСТ 34.602–89. Информационная технология. Техническое задание на создание автоматизированных систем. Настоящий стандарт устанавливает требования на структуру и содержание технического задания на систему.

Техническое задание является ключевым исходным документом, определяющим состав и основные функции функциональных и обеспечивающих подсистем, процесса создания и приемки АС, взаимоотношения разработчика и заказчика на всех этапах жизненного цикла АС. При этом ТЗ разрабатывает организация-разработчик (по ГОСТ 34.602-89), но формально ТЗ разработчику выдает заказчик (по РД 50-680-88).

Таблица 2.4

Основные этапы и стадии создания АС

Наименование этапа	Содержание этапа
1. Формирование требований к АС	Обследование объекта и обоснование необходимости создания АС. Формирование требований пользователя АС. Оформление отчета о выполненной работе и заявки на разработку АС (тактико-технического задания)
2. Разработка концепции АС	Изучение объекта. Проведение необходимых научно-исследовательских работ. Разработка вариантов концепции АС и выбор варианта концепции АС, удовлетворяющего требованиям пользователя. Оформление отчета о выполненной работе
3. Техническое задание	Разработка и утверждение технического задания на создание АС
4. Эскизный проект	Разработка предварительных проектных решений по системе в целом и ее частям. Разработка документации на АС и ее части
5. Технический проект	Разработка проектных решений по системе и ее частям. Разработка документации на АС и ее части. Разработка и оформление документации на поставку изделий для комплектования АС и/или технических требований (технических заданий) на их разработку. Разработка заданий на проектирование в смежных частях проекта объекта автоматизации
6. Рабочая документация	Разработка рабочей документации на систему и ее части. Разработка или адаптация программ

Окончание табл. 2.4

Наименование этапа	Содержание этапа
7. Ввод в действие	Подготовка объекта автоматизации к вводу АС в действие. Подготовка персонала. Комплектация АС поставляемыми изделиями (программными и техническими средствами, программно-техническими комплексами, информационными изделиями). Строительно-монтажные работы. Пусконаладочные работы. Проведение предварительных испытаний. Проведение опытной эксплуатации. Проведение приемочных испытаний
8. Сопровождение АС	Выполнение работ в соответствии с принятыми гарантийными обязательствами. Послегарантийное обслуживание

2.1.4. Государственные стандарты РФ (ГОСТ Р) и международные стандарты ИСО

Все течет, все изменяется: единая Европа —
это и единые стандарты на информационные технологии.

В настоящее время в РФ действует ряд государственных стандартов РФ, разработанных на основе прямого применения международных стандартов и дополняющих ГОСТы 19 и 34.

ГОСТ Р ИСО/МЭК 9294-93. Информационная технология. Руководство по управлению документированием программного обеспечения. Стандарт полностью соответствует международному стандарту ИСО/МЭК ТО 9294:1990 и устанавливает рекомендации по эффективному управлению документированием ПС для руководителей, отвечающих за их создание. Целью стандарта является оказание помощи в определении стратегии документирования ПС, выборе стандартов по документированию, выборе процедур документирования, определении необходимых ресурсов, составлении планов документирования.

ГОСТ Р ИСО 9127-94. Системы обработки информации. Документация пользователя и информация на упаковке для потребительских программных пакетов. Стандарт полностью соответствует международному стандарту ИСО 9127:1989. В контексте настоящего стандарта под потребительским программным пакетом (ПП) понимается «программная продукция, спроектированная и продаваемая для выполнения определенных функций; программа и соответствующая ей документация, упакованные для продажи как единое целое». Под документацией пользователя понимается документация, которая обеспечивает конечного пользователя информацией по установке и эксплуатации ПП. Под информацией на упаковке понимают информацию, воспроизводимую на внешней упаковке программного пакета. Ее целью является предоставление потенциальным покупателям первичных сведений о ПП.

ГОСТ Р ИСО/МЭК 8631-94. Информационная технология. Содержит программные конструктивы и условные обозначения для их представления. Описывает представление процедурных алгоритмов.

В дополнение к отечественным стандартам ***целесообразно использовать зарубежные стандарты***. Наибольшие достижения в регламентировании требований к объектам и процессам ЖЦ ПС и их реализации сосредоточены в стандартах Министерства обороны США, которые должны обеспечивать высокое качество и безопасность функционирования критических военных систем. Стандарт **МО США MIL-STD-498** представляет собой комплект из трех документов:

1) собственно стандарт MIL-STD-498. Разработка и документирование программного обеспечения (Software Development and Documention);

2) руководство «Обзор и адаптация» (подготовка к применению) (Over-view and Tailoring);

3) руководство «Применение и рекомендации» (Application and Reference).

Создание ПС рассматривается как часть полного процесса разработки специальных информационных систем военного назначения. Отмечается, что этот стандарт базируется на процессах и документах, представленных в международном стандарте **ISO/IEC 12207:1995. Процессы жизненного цикла программных средств**, а также в **стандарте по обеспечению качества продукции ISO 9001** и предшествовавших военных стандартах. Начальные этапы проектирования и заключительные этапы испытаний и сдачи заказчику объединены в совместный анализ программных и аппаратных средств целостной информационной системы, полностью решающей требуемые потребителем функциональные задачи. Стандарт унифицирует требования к проектированию, разработке, модификации и документированию ПС. Проведено разделение требований к объектам и процессам жизненного цикла ПС.

Отдельная группа стандартов посвящена регламентации жизненного цикла на создание и сопровождение программных систем. Одним из них является **стандарт ISO 12207:1995. Процесс жизненного цикла программных средств**, определяющий архитектуру, процессы, разделы и подразделы жизненного цикла ПС, а также перечень базовых работ, детализирует содержание каждой из них.

ISO 12207 — базовый стандарт процессов жизненного цикла ПО, ориентированный на различные (любые!) виды ПО и типы проектов АС, куда ПО входит как часть. Стандарт определяет стратегию и общий порядок в создании и эксплуатации ПО, он охватывает жизненный цикл ПО от концептуализации идей до завершения жизненного цикла.

В основу стандарта положены следующие базовые понятия:

система — это объединение одного или более процессов, аппаратных средств, программного обеспечения, оборудования и специалистов для обеспечения возможности удовлетворения определенных потребностей или целей;

модель жизненного цикла — структура, содержащая процессы, действия и задачи, которые осуществляются в ходе разработки, функционирования и сопровождения программного продукта в течение всей жизни системы, от определения требований до завершения ее использования. Множество процессов и задач сконструировано так, что возможна их адаптация в соответствии с проектами ПО. Процесс адаптации — это процесс исключения видов деятельности и задач, не применимых в конкретном проекте;

требование квалификации — набор критериев или условий (квалификационные требования), которые должны быть удовлетворены, чтобы квалифицировать программный продукт как подчиняющийся (удовлетворяющий условиям) его спецификации и готовый для использования в целевой окружающей среде.

Стандарт не предписывает конкретную модель жизненного цикла или метод разработки программного обеспечения, но определяет, что стороны-участники использования стандарта ответственны за выбор модели жизненного цикла для проекта ПО, за адаптацию процессов и задач стандарта к этой модели, за выбор и применение методов разработки ПО, за выполнение действий и задач, подходящих для проекта ПО. Стандарт ISO12207 равносильно ориентирован на организацию действий каждой из двух сторон: поставщика (разработчика) и покупателя (пользователя). Стандарт может быть в равной степени применен, когда обе стороны входят в одну организацию.

Каждый процесс жизненного цикла разделен на набор действий, каждое действие — на набор задач. Очень важное отличие

ISO: каждый процесс, действие или задача инициируются и выполняются другим процессом по мере необходимости, причем нет заранее определенных последовательностей (естественно, при сохранении логики связей по исходным сведениям задач и т. п.).

Стандарт ISO12207 содержит:

пять основных процессов жизненного цикла ПО:

1) *процесс приобретения*, определяющий действия предприятия-покупателя, которое приобретает АС, программный продукт или сервис ПО;

2) *процесс поставки*, определяющий действия предприятия-поставщика, снабжающего покупателя системой, программным продуктом или сервисом программного обеспечения;

3) *процесс разработки*, определяющий действия предприятия-разработчика, которое разрабатывает принцип построения программного изделия и программный продукт;

4) *процесс функционирования*, определяющий действия предприятия-оператора, которое обеспечивает обслуживание системы (а не только ПО) в процессе ее функционирования в интересах пользователей. В отличие от действий, которые определяются разработчиком в инструкциях по эксплуатации (эта деятельность разработчика предусмотрена во всех трех стандартах), конкретизируются действия оператора (консультирование пользователей, получение обратной связи и др.), которые планируются им самостоятельно и в дальнейшем выступают как соответствующие, принятые на себя, обязанности;

5) *процесс сопровождения*, определяющий действия персонала сопровождения, который обеспечивает сопровождение программного продукта, включающее управление модификациями программного продукта, поддержку его текущего состояния и функциональной пригодности, инсталляцию и удаление программного изделия на вычислительной системе;

четыре вспомогательных процесса, которые поддерживают реализацию другого процесса, будучи неотъемлемой частью всего жизненного цикла программного изделия, и обеспечивают должное качество проекта ПО:

1) *решение проблем*;

2) *документирование*;

3) *управление конфигурацией*;

4) *гарантирование качества,* использующего результаты остальных процессов группы обеспечения качества, а именно: *верификации*, *аттестации, совместной оценки*, *аудита*;

четыре организационных процесса:

1) *управление;*

2) *создание инфраструктуры;*

3) *усовершенствование;*

4) *обучение.*

Под *процессом усовершенствования* понимается не усовершенствование АС или ПО, а улучшение самих процессов приобретения, разработки, гарантирования качества и других, реально осуществляемых в организации процессов.

К вышеуказанным процессам добавляется особый *процесс адаптации*, определяющий основные действия, необходимые для адаптации стандарта к условиям конкретного проекта.

«Динамический» характер стандарта определяется способом определения последовательности выполнения процессов и задач, при котором один процесс при необходимости вызывает другой или его часть.

Например, выполнение *процесса приобретения* в части анализа и фиксации требований к системе или ПО может вызывать исполнение соответствующих задач *процесса разработки*. В *процессе поставки* поставщик должен управлять субподрядчиками согласно *процессу приобретения* и выполнять верификацию и аттестацию по соответствующим процессам. *Процесс сопровождения* может требовать развития системы и ПО. В этом случае он приобретает характер *процесса разработки*.

Такой характер позволяет реализовывать любую модель жизненного цикла. Анализ требований к ПО проводится в соответствии с 11 классами характеристик качества, используемыми позже при гарантировании качества. При этом разработчик должен установить и документировать следующие требования к программному обеспечению:

1) функциональные и возможные спецификации, включая исполнение, физические характеристики и условия среды эксплуатации, при которых единица программного обеспечения должна быть выполнена;

2) внешние связи (интерфейсы) с единицей программного обеспечения;

3) требования квалификации;

4) спецификации надежности, включая спецификации, связанные с методами функционирования и сопровождения, воздействия окружающей среды и вероятностью травмы персонала;

5) спецификации защищенности;

6) человеческие факторы спецификаций по инженерной психологии (эргономике), включая связанные с ручным управлением, взаимодействием человека и оборудования, ограничениями на персонал и с областями, нуждающимися в концентрированном человеческом внимании, которые являются чувствительными к ошибкам человека и обучению;

7) определение данных и требований базы данных;

8) установочные и приемочные требования поставляемого программного продукта в местах функционирования и сопровождения (эксплуатации);

9) документация пользователя;

10) работа пользователя и требования выполнения;

11) требования сервиса пользователя.

Интересно и важно, что эти и аналогичные характеристики хорошо корреспондируются с характеристиками АС, предусматриваемыми в ГОСТ 34 по видам обеспечения системы.

2.1.5. Некоторые рекомендации по взаимодействию разработчика и заказчика при создании программного обеспечения по ГОСТ 19

Заказчик не всегда представляет, чего он хочет, но он всегда прав.

Деятельность разработчиков и заказчиков, направленная на снижение рисков, связанных с ошибками в оценке длительности и стоимости разработки заказного ПО, может осуществляться по двум сценариям:

1) взаимоотношения заказчика и разработчика строятся на взаимном доверии, просчеты в оценке проекта берет на себя в основном разработчик — *мягкое внедрение*;

2) взаимоотношения заказчика и разработчика строго регламентированы и обязательны для исполнения обеими сторонами, спорные моменты часто могут приводить к конфликтам — *жесткое внедрение.*

Первый сценарий (мягкое внедрение) ориентирован на проекты с небольшой продолжительностью (до 3-х месяцев) либо проекты, которые можно разбить на отдельные этапы: постановочный, уточняющий, стабилизирующий, внедрение. Планирование трудоемкости и оценка стоимости проводится по каждому этапу отдельно.

Постановочный этап

Данный этап проводится по договору о консалтинге, ввиду невозможности спланировать заранее стоимость проекта. Оценка затрат производится по суммарной трудоемкости в человекоднях. Результаты выполнения этапа оформляются в виде документа «Техническое задание» (ТЗ). Данный документ должен определять цель проекта и включать в себя описание проекта и список ключевых требований без подробной расшифровки. Несмотря на отсутствие подробного описания, состав работ должен поддаваться статистической оценке трудоемкости со стандартным отклонением (риском) в разумных рамках. Кроме того, в данном документе необходимо представить предварительную оценку экономической эффективности от внедрения проекта.

При расчете трудоемкости целесообразно использовать статистику трудоемкости (эффективности) аналогичных проектов. При отсутствии данной статистики неизбежны ошибки в оценках, в этом случае следует попробовать получить статистику, опираясь на результаты разработки прототипов. Для оценки рисков рекомендуется разработать как минимум 2 простейших прототипа:

1) «***интерфейсный прототип***», имитирующий 1–2 важнейших диалога программы и ориентированный на изучение рисков, связанных с модификацией требований будущих пользователей;

2) «***архитектурный прототип***», отображающий наиболее критические места будущей архитектуры и предназначенный для оценки технологических рисков.

В дальнейшем данные прототипы не должны использоваться при разработке системы. Это связано с тем, что прототипы служат

для нахождения оптимальных решений, но не являются сами оптимальными решениями.

Оценку рисков требуется выразить в виде возможного превышения трудоемкости (пессимистическая оценка). Именно из данной оценки следует исходить при определении общей стоимости проекта.

Условием завершения этапа является подписание сторонами технического задания.

Уточняющий этап

На данном этапе производится создание серии рабочих прототипов будущей системы, проводится согласование ее функциональности с ключевыми пользователями. Стоимость этапа составляет примерно 30 % от общей стоимости разработки.

Одновременно в виде пошаговых сценариев создается «Руководство пользователя», разрабатываются отдельные пункты документа «Описание применения». Сначала формируется общее описание системы, затем готовятся должностные инструкции для пользователей. Пользователи оценивают прототип и документацию одновременно.

На данном этапе «Руководство пользователя» фактически заменяет классическое ТЗ. Такой подход имеет ряд преимуществ:

1) включение пользователя в анализ своей рабочей документации непосредственно на первых этапах разработки программы;

2) отсутствие необходимости в одновременной правке технического задания и «Руководства пользователя»;

3) достижение соответствия создаваемой документации текущему состоянию будущего проекта.

Условием завершения этапа является подписание письменного соглашения заказчика и разработчика о принятии системы при наличии ее соответствия последней согласованной версии «Руководства пользователя», стабильности архитектуры и требований к системе (допустимые изменения в ходе следующего этапа составляют не более 20 %).

В случае если не удается достигнуть согласия ключевых пользователей с прототипом или описанием в документации, заказчик должен принять волевое решение на уровне топ-менеджера и определиться с требованиями. Если этого не удается достичь

или требования выходят за рамки ТЗ с учетом надбавок на риск, рекомендуется пересмотреть трудоемкость/цену проекта или прекратить его разработку. Указанная возможность прекращения проекта должна быть отражена в договоре.

Стабилизирующий этап

На данном этапе устраняются недостатки в прототипах и документации и выпускается «Релиз системы». Стоимость этапа составляет примерно 50 % от общей стоимости разработки.

В начале этапа составляется и согласовывается документ «Программа и методика испытаний». Данный документ содержит описание тестов, успешное выполнение которых является необходимым и достаточным условием приемки системы. Иными словами, документ описывает минимально гарантированное качество реализации. После тестирования отдельных компонентов системы по инициативе тестера и согласию заказчика «Руководство пользователя» может быть изменено.

После исправления дефектов, выявленных тестерами, выпускается первая версия системы, проходит ее окончательное тестирование и в случае успешного тестирования объявляется готовой к сдаче в опытную эксплуатацию.

Внедрение

На данном этапе заказчик выявляет дефекты программы, обнаруженные в процессе опытной эксплуатации, а разработчик их устраняет. Ошибка это или доработка решается на основании «Руководства пользователя». Стоимость этапа составляет примерно 10 % от разработки.

После доработки разработчик изменяет «Руководство пользователя», устанавливает систему на рабочей станции заказчика, проводит обучение пользователей. Пользователи должны изучить свои должностные инструкции и подтвердить возможность эксплуатации системы в соответствии с данными инструкциями. Эту процедуру можно проводить в виде аттестации рабочего места пользователя. Если все ошибки, выявленные в процессе опытной эксплуатации, исправлены, и окончательная версия программы соответствует «Руководству пользователя», то по согласованию с заказчиком разрабатываются остальные документы согласно табл. 2.2 и принимается решение о приемке системы в промышленную эксплуатацию.

Второй сценарий (**жесткое внедрение**) предполагает соблюдение разработчиком и заказчиком одного из отечественных либо зарубежных стандартов по жизненному циклу создания программных продуктов (предпроектное обследование, проектирование, разработка, документирование, тестирование, опытная эксплуатация, сдача в промышленную эксплуатацию) и документирование каждого из его этапов. На начальном этапе заказчик представляет будущий ПП, удовлетворяющий некоторому набору требований. Эти требования формулируются, как правило, нечетко (расплывчато). Поэтому первая версия системы это обычно не готовый программный продукт, а некоторый прототип. Отличие же готового продукта от прототипа заключается в следующем: функционирование системы не требует участия разработчика, функции системы соответствуют технической документации, отсутствуют ошибки в программах.

Причина некачественной первой версии заключается, в первую очередь, в принципиальной нечеткости требований, а не в ошибках проектирования и программирования. Заказчик считает, что его требования правильны, причем это может происходить как по причине его консервативности, так и по причине нежелания признавать свои ошибки и просчеты. В этом случае разработчик может взять ответственность за нечеткость требований на себя, но потребовать увеличения сроков и стоимости проекта.

В противном случае налицо конфликтная ситуация. Во избежание этого сформулируем ряд рекомендаций по формализации отношений между заказчиком и разработчиком:

1) документы, формализующие взаимоотношения, могут быть двух типов: планы работ, определяющие, что надо сделать; функциональные и информационные модели, описывающие бизнес-процессы заказчика (как надо делать);

2) все требования и планы работ должны оформляться в документальном виде с указанием сроков и исполнителей и утверждаться первым руководителем организации;

3) требования к системе и планы работ должны детализироваться до простейших задач, имеющих однозначную трактовку;

4) необходимо заранее согласовать с заказчиком контрольные тесты (примеры) и договориться о том, что именно они являются критерием корректности работы системы.

2.1.6. Стандартизированные показатели качества программных систем и баз данных

Доказать преимущество авторского программного продукта можно путем сравнения его с аналогами по объективным общепринятым критериям.

Формализации показателей качества программных средств посвящена также целая серия стандартов. В базовом международном **стандарте ISO/МЭК 9126:1991. *Оценка программного продукта. Характеристики качества и руководство по их применению.*** При отборе минимума стандартизируемых показателей, описываемых в стандарте разработчиками, выдвигались и учитывались следующие принципы:

- ясность и измеряемость значений;
- независимость между используемыми показателями;
- соответствие установившимся понятиям и терминологии;
- возможность последующего уточнения и детализации.

Выделенные в стандарте характеристики позволяют оценивать ПС ***с позиции пользователя, разработчика и управляющего проектом***.

Пользователи в основном проявляют заинтересованность к возможностям применения программного обеспечения, его производительности и результатах использования и оценивают ПО без изучения его внутренних аспектов или условий создания программного обеспечения.

Пользователя могут интересовать следующие вопросы:

- наличие требуемых функций в программном обеспечении;
- надежность программного обеспечения;
- эффективность программного обеспечения;
- удобство при использовании ПО;
- простота переноса ПО в другую среду.

Процесс создания требует от пользователя и разработчика использования одних и тех же характеристик качества программного обеспечения, так как они применяются для установления требований и приемки. Когда разрабатывается программное обеспечение для продажи, в требованиях должны быть отражены предполагаемые потребности.

Так как разработчики отвечают за создание программного обеспечения, которое должно удовлетворять требованиям качества, они заинтересованы в качестве промежуточной продукции так же, как и в качестве конечной продукции.

Представление пользователя также должно включать представление о характеристиках качества, требуемое тем, кто сопровождает программное обеспечение.

Руководитель проекта может быть больше заинтересован в общем качестве, чем в конкретной характеристике качества, и по этой причине будет нуждаться в определении важности значений, отражающих коммерческие требования для индивидуальных характеристик.

Руководителю также может потребоваться сопоставление повышения качества с критериями управляемости, такими как плановая задержка или перерасход стоимости, потому что он желает оптимизировать качество в пределах ограниченной стоимости, трудовых ресурсов и установленного времени.

В стандарте рекомендуется использовать шесть основных характеристик качества ПС, каждая из которых детализируется еще несколькими субпоказателями. В настоящее время подготовлен проект модернизации стандарта, в котором предполагается выделить четыре части:

1) модель показателей качества и их развития в жизненном цикле ПС;

2) внешние метрики качества, которые отражают наиболее общие характеристики программного средства;

3) внутренние метрики качества, характеризующие структуру и конструктивные особенности ПС;

4) метрики качества ПС с позиции пользователя:

– эффективность применения ПС пользователем,

– производительность при решении основных функциональных задач,

– степень защищенности от внешних угроз,

– удовлетворенность пользователей качеством решения задач.

Близкими к описанному стандарту по идеологии, структуре и содержанию являются три отечественных стандарта:

ГОСТ 28806-90. Качество программных средств. Термины и определения;

ГОСТ 28195-89. Оценка качества программных средств. Общие положения;

ГОСТ Р ИСО/МЭК 9126-93. Информационная технология. Оценка программной продукции.

В вышеуказанных документах для оценки качества ПС выделяются шесть характеристик: функциональные возможности, надежность, практичность, эффективность, сопровождаемость, мобильность.

Функциональные возможности — набор атрибутов, относящихся к сути набора реализуемых в программном продукте функций и их конкретным свойствам (установленные или предполагаемые потребности).

Надежность — набор атрибутов, относящихся к способности программного обеспечения сохранять свой уровень качества функционирования при установленных условиях за установленный период времени.

Практичность — набор атрибутов, относящихся к объему работ, требуемых для использования и индивидуальной оценки такого использования определенным и предполагаемым кругом пользователей.

Эффективность — набор атрибутов, относящихся к соотношению между уровнем качества функционирования программного обеспечения и объемом используемых ресурсов при установленных условиях.

Сопровождаемость — набор атрибутов, относящихся к объему работ, требуемых для проведения конкретных изменений (модификаций).

Мобильность — набор атрибутов, относящихся к способности ПО быть перенесенным из одного окружения в другое.

В случае необходимости относительная важность каждой характеристики может изменяться в зависимости от специфики программного продукта и области применения. Например, надежность наиболее важна для программного обеспечения боевых критичных систем, эффективность наиболее важна для ПО критичных по времени систем реального времени, а практичность — для ПО диалога конечного пользователя.

В дальнейшем каждая характеристика должна быть детализирована на подхарактеристики. В стандарте ISO/МЭК 9126:1991 предлагается набор подхарактеристик, представленный на рис. 2.1.

Рис. 2.1. Характеристики качества ПС

Детализация каждой характеристики на подхарактеристики является необходимым этапом построения качественной модели оценивания программного продукта. В настоящее время в литературе имеется описание ряда подобных моделей, однако сте-

пень их завершенности пока не позволяет создавать какой-либо единый стандарт.

Набор вышеуказанного стандарта подхарактеристик включает:

функциональные возможности:

– *пригодность* — атрибут ПО, относящийся к наличию и соответствию набора функций конкретных задач;

– *правильность* — атрибуты ПО, относящиеся к обеспечению правильности или соответствия результатов или эффектов;

– *способность к взаимодействию* — атрибуты программного обеспечения, относящиеся к способности его взаимодействовать с конкретными системами;

– *согласованность* — атрибуты ПО, заставляющие программу придерживаться соответствующих стандартов или соглашений, или положений законов, или подобных рекомендаций;

– *защищенность* — атрибуты ПО, относящиеся к его способности предотвращать несанкционированный доступ, случайный или преднамеренный, к программам и данным;

надежность:

– *стабильность* — атрибуты ПО, относящиеся к частоте отказов при ошибках в программном обеспечении;

– *устойчивость к ошибке* — атрибуты программного обеспечения, относящиеся к его способности поддерживать определенный уровень качества функционирования в случаях программных ошибок или нарушения определенного интерфейса;

– *восстанавливаемость* — атрибуты программного обеспечения, относящиеся к его возможности восстанавливать уровень качества функционирования и данные, непосредственно поврежденные в случае отказа, а также к времени и усилиям, необходимым для этого;

практичность:

– *понятность* — атрибуты программного обеспечения, относящиеся к усилиям пользователя по пониманию общей логической концепции и ее применимости;

– *обучаемость* — атрибуты программного обеспечения, относящиеся к усилиям пользователя по обучению его применению (например, оперативному управлению, вводу, выводу);

– *простота использования* — атрибуты ПО, относящиеся к усилиям пользователя по эксплуатации и оперативному управлению;

эффективность:

– *временная экономичность* — атрибуты программного обеспечения, относящиеся к временам отклика и обработки и к скоростям выполнения его функций;

– *ресурсная экономичность* — атрибуты ПО, относящиеся к объему используемых ресурсов и продолжительности такого использования при выполнении функции;

сопровождаемость:

– *анализируемость* — атрибуты программного обеспечения, относящиеся к усилиям, необходимым для диагностики недостатков или случаев отказов или определения составных частей для модернизации;

– *изменяемость* — атрибуты программного обеспечения, относящиеся к усилиям, необходимым для модификации, устранению отказа или для изменения условий эксплуатации;

– *устойчивость* — атрибуты ПО, относящиеся к риску от непредвиденных эффектов модификации;

– *тестируемость* — атрибуты ПО, относящиеся к усилиям, необходимым для проверки модифицированного программного обеспечения;

мобильность:

– *адаптируемость* — атрибуты ПО, относящиеся к удобству его адаптации к различным конкретным условиям эксплуатации без применения других действий или способов, кроме тех, что предназначены для этого в рассматриваемом ПО;

– *простота внедрения* — атрибуты программного обеспечения, относящиеся к усилиям, необходимым для внедрения программного обеспечения в конкретное окружение;

– *соответствие* — атрибуты программного обеспечения, которые заставляют программу подчиняться стандартам или соглашениям, относящимся к мобильности;

– *взаимозаменяемость* — атрибуты ПО, относящиеся к простоте и трудоемкости его применения вместо другого конкретного программного средства в среде этого средства.

Немаловажной проблемой при оценке характеристик и подхарактеристик является выбор соответствующих метрик (показателей) их описания. В литературе существует несколько общепринятых метрик для характеристик (подхарактеристик), описанных выше. Примеры описания характеристики практичности представлены в табл. 2.5.

Таблица 2.5

Метрика характеристики «Практичность»

Характеристики качества	Мера	Шкала
Понятность		
четкость концепции	Порядковая	Отл., хор.
демонстрационные возможности		Удовл., неудовл.
наглядность и полнота документации		
Простота использования		
простота управления функциями	Порядковая	Отл., хор.
комфортность эксплуатации		Удовл., неудовл.
среднее время ввода заданий	Секунды	1–1000
среднее время отклика на задание	Секунды	1–1000
Обучаемость		
трудоемкость изучения применения ПС	Чел.-ч	1–1000
продолжительность изучения	ч	1–1000
объем эксплуатационной документации	Страницы	1–1000

Участники процесса оценивания могут разрабатывать и собственные метрики.

Вышеописанные стандарты рекомендуется применять для установления требований к качеству ПО и оценивания (измерения, ранжирования и оценки) программных продуктов после завершения их разработки в следующих случаях:

- определение требований к качеству программной продукции;
- оценивание технических требований к ПО при контроле за выполнением требования к качеству в процессе разработки;
- описание признаков и свойств (атрибутов) внедренного ПО (например, в руководствах пользователя);

- оценивание разработанного ПО перед его поставкой;
- оценивание программного обеспечения перед приемкой.

Для оценки баз данных (БД) пока отсутствуют какие-либо стандарты, регламентирующие их показатели качества. Поэтому ниже представлен набор характеристик, который наиболее часто используется на практике при решении этой проблемы. В системах баз данных доминирующее значение приобретают сами данные, их хранение и обработка. Поэтому при анализе качества БД целесообразно выделить два компонента [8]:

1) программные средства системы управления базой данных, независимые от сферы их применения и смыслового содержания накапливаемых и обрабатываемых данных;

2) информация БД, доступная для обработки и использования в конкретной проблемно-ориентированной сфере применения.

Практически весь набор показателей качества ПС, изложенный выше, в той или иной степени может использоваться при анализе и оценке качества СУБД. Особенности состоят в изменении акцентов при выборе и упорядочении этих показателей качества. Почти во всех случаях важнейшими показателями качества СУБД являются функциональные характеристики процессов формирования и изменения информационного наполнения БД администраторами, а также доступа к данным и представления результатов пользователям БД.

Качество интерфейса специалистов с БД, обеспечиваемого средствами СУБД, оценивается в значительной степени субъективно, однако имеется ряд характеристик, которые можно оценивать достаточно корректно.

Вторым компонентом БД является собственно накапливаемая и обрабатываемая информация в базе данных. Выделяемые показатели качества должны иметь практический интерес для пользователей БД и быть упорядочены в соответствии с приоритетами практического применения. Кроме того, каждый выделяемый для проверки показатель должен быть пригоден для достаточно достоверного измерения и сравнения с требуемым значением при испытаниях и сертификации.

Функциональными показателями качества информации БД являются:

- полнота накопленных описаний объектов — относительное число объектов или документов, имеющихся в БД, к общему числу объектов по данной тематике или к числу объектов в аналогичных БД по той же тематике;
- достоверность — степень соответствия данных об объектах в БД реальным объектам вне ЭВМ в данный момент времени, определяющаяся изменениями самих объектов, некорректностью записей об их состоянии или некорректностью расчетов их характеристик;
- идентичность данных — относительное число описаний объектов, не содержащих ошибки, к общему числу документов об объектах в БД;
- актуальность данных — относительное число морально устаревших данных об объектах в БД к общему числу накопленных и обрабатываемых данных.

К конструктивным показателям качества информации в БД относятся в основном объемно-временные характеристики сохраняемых и обрабатываемых данных:

- объем базы данных — число записей описаний объектов или документов в БД, доступных для хранения и обработки;
- оперативность — степень соответствия изменений данных в процессе сбора и обработки состояниям реальных объектов или величина запаздывания между появлением или изменением характеристик реального объекта и его отражением в БД;
- периодичность — промежуток времени между поставками двух последовательных, достаточно различающихся информацией версий БД;
- глубина ретроспективы — интервал времени от даты выпуска и/или записи в базу данных самого раннего документа до настоящего времени;
- динамичность — относительное число изменяемых описаний объектов к общему числу записей в БД за некоторый интервал времени, определяемый периодичностью издания версий БД.

К конструктивным относятся и все показатели защищенности информации. Защищенность реализуется, в основном, программными средствами СУБД, однако в сочетании с поддерживающими их средствами организации данных. В распределенных

базах данных показатели защищенности тесно связаны с характеристиками целостности данных. Эти показатели отражают степень тождественности данных в памяти удаленных компонентов распределенной БД.

При реальном функционировании баз данных важную роль играют временные характеристики взаимодействия конечных пользователей и администраторов БД в процессе эксплуатации базы данных по прямому назначению. Эти характеристики зависят от качества СУБД, а также от объема, структуры и показателей качества используемой информации. Выше они отражены критерием эффективности использования ресурсов ЭВМ программными средствами, в данном случае СУБД.

Для баз данных важнейшим ресурсом является память ЭВМ, занимаемая информацией БД, а также используемость этого ресурса. Эти показатели качества влияют на время реакции БД на разные виды запросов пользователей и на пропускную способность БД при эксплуатации.

2.2. Правовое регулирование отношений по охране и защите прав на программы для ЭВМ и базы данных[1]

2.2.1. Особенности программного обеспечения как интеллектуального продукта

Имею, но не осязаю. Отдаю (продаю) и не теряю.

Отношения, связанные с охраной и использованием объектов интеллектуальной собственности, входят в предмет регулирования российского гражданского законодательства (ст. 2 ГК РФ), включающего в себя институты авторского, патентного права и т. д. В гражданском законодательстве выделяются следую-

[1] Подраздел написан по материалам учебного пособия Ефимова А.А. «Правовое регулирование процесса создания и использования программ для ЭВМ и баз данных». — Томск: Томск. гос. ун-т систем управления и радиоэлектроники, 2007. — 184 с.

щие виды объектов интеллектуальной собственности, представленные на рис. 2.2.

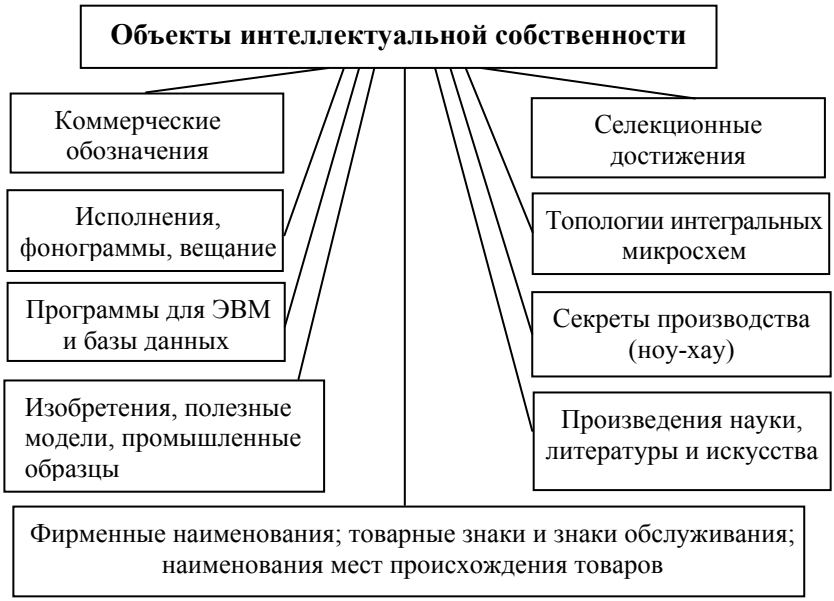

Рис. 2.2. Объекты интеллектуальной собственности

Программы для ЭВМ являются объектом интеллектуальной собственности, который, по сути, представляет собой новый специфичный вид собственности, порождающий в свою очередь иные отношения присвоения. Рассуждения о собственности как экономической категории традиционно сопровождаются ее системным представлением как некой совокупности взаимосвязанных элементов, включающей объекты, субъектов, преследующих свои экономические интересы, и содержание самих отношений. В этом случае объекты интеллектуальной собственности проявляются как результат взаимосвязи социальных субъектов в сфере интеллектуальной деятельности, специфика которой состоит в том, что ее результаты имеют ***нематериальную природу***. Не имея материальной природы, эти результаты не могут быть сохранены, преобразованы или переданы, а значит и не могут

вступать в экономический оборот, следовательно, их всегда сопровождает материальный носитель, в котором они овеществляются. При этом обладание материальным носителем информации не делает его приобретателя собственником самой информации, так как ничто не мешает его создателю вновь воспроизвести соответствующий материальный носитель в силу того, что эта информация по-прежнему существует в его сознании.

Объекты интеллектуальной собственности в силу нематериальной природы имеют ***специфические особенности*** и выступают на рынке как особый товар. А любой товар обладает двумя экономическими свойствами: потребительной стоимостью и меновой стоимостью. ***Потребительная стоимость*** — это полезность товара, его способность удовлетворять какие-либо человеческие потребности. ***Меновая стоимость*** — это свойство обмена, т. е. способность одного товара обмениваться на другой в определенных пропорциях и количественных соотношениях.

Потребительная стоимость объектов интеллектуальной собственности обладает целым рядом особенностей. В процессе потребления обычный материальный объект исчезает, прекращает свое существование, переносит свою стоимость на вновь создаваемый продукт. При этом в силу своей материальной природы он может быть использован только в одном месте одним лицом. При потреблении интеллектуального продукта его полезность не исчезает, количество не уменьшается, качество не ухудшается; он может быть использован неограниченным кругом лиц. В силу невещественной, нефизической природы он не подвергается физическому износу, но с течением времени изнашивается морально, поскольку лежащая в основе интеллектуального продукта информация устаревает. Именно моральный износ (утрата новизны) результата интеллектуальной деятельности свидетельствует о его потреблении, т. е. затраты труда на его создание постепенно переносятся на другие продукты труда, на него падает спрос, к нему теряется экономический интерес. В силу ***неуничтожимости, бесконечной воспроизводимости***, возможности ***многократного использования*** результаты интеллектуальной деятельности обладают высоким потенциалом доходности по сравнению с материальными объектами.

Объект интеллектуальной собственности, имея абстрактную форму выражения (в виде прав), может ***обмениваться любое количество раз***, и при каждой сделке собственник интеллектуального товара получает денежное вознаграждение, сохраняя свои права. В результате проданный объект используется и продавцом и покупателем одновременно. Этот эффект «размножения» объективно вытекает из нематериальной природы объекта интеллектуальной собственности. Высокий потенциал доходности объектов интеллектуальной собственности связан не только с неограниченным кругом субъектов, которые могут участвовать в сделках, но и с ***уникальной природой*** самих объектов, которая нередко порождает совершенно непредсказуемое несоответствие в соотношении доходов и затрат на их создание: незначительные затраты могут дать колоссальный по величине эффект, и, наоборот, колоссальные затраты часто заканчиваются ничем с точки зрения экономического эффекта. Непредсказуемая уникальная работа интеллекта автора создает условия «естественной монополии» на потенциальную прибыль от коммерческой реализации. Рассмотренные особенности потребительной стоимости объектов интеллектуальной собственности непосредственно влияют на формирование этой стоимости, придавая данной категории высокую степень неопределенности. Кроме того, ***временные ограничения*** и ***большая общественная значимость*** результата интеллектуального труда лишают его обладателя абсолютной экономической власти над ним, а также возможности только по своему усмотрению принимать решения о способах и продолжительности его употребления.

Экономическое содержание таких отношений реализуется посредством определения собственности как системы социально-экономических отношений между людьми по поводу присвоения и отчуждения условий и результатов производства: кто (субъекты хозяйствования), что (объекты) и как (посредством каких экономических связей) присваивает и отчуждает.

Присвоение материальных благ осуществляется в форме владения, пользования и распоряжения. *Владение* — хозяйственное (экономическое) господство над вещью, физическое обладание ею. Законное владение имеет правовое основание (закон, договор, административный акт). *Пользование* — это эксплуатация

объекта собственности в интересах субъекта, предполагающая получение выгоды в результате применения объекта. Границы права пользования определяются законом, договором или иным правовым основанием. *Распоряжение* — принятие решений по поводу целей и способов функционирования объекта собственности, возможность определения его юридической судьбы (изменение принадлежности). Осуществляется посредством совершения различных сделок (купли-продажи, дарения, обмена и т. д.).

Эта традиционная для материальных объектов система присвоения и отчуждения «не работает» применительно к результатам интеллектуальной деятельности по следующим причинам:

- физическое обладание идеями и знаниями невозможно в силу их неосязаемости, пользоваться же ими одновременно может бесчисленное количество субъектов;
- если при отчуждении материального объекта его создатель (пользователь) лишается всяких прав на него, а новый собственник получает возможность считать его исключительно своим, то нематериальный объект может передаваться неограниченному кругу лиц, оставаясь в то же время у своего создателя (пользователя);
- принципиально отличается и распоряжение нематериальными объектами.

Таким образом, присвоение результата интеллектуальной собственности во всех указанных формах может одновременно осуществляться разными субъектами.

Прикладные программные системы могут находиться ***во владении*** многих, по меньшей мере, тех, кто с ним ознакомился, ***использовать*** (тиражировать, распространять и т. п.) его могут некоторые, а ***распоряжаться*** судьбой произведения может только автор, зачастую не имеющий права на его использование и возможности уничтожения или отказа от него.

Отсюда следует, что традиционная система присвоения в форме владения, пользования, распоряжения не отражает особенностей собственности на данную ППС, а все отмеченные признаки результатов интеллектуальной деятельности ставят под сомнение применимость к этим результатам категории «собственность» в ее традиционном смысле.

На основании вышеизложенного следует признать, что интеллектуальный продукт является достаточно ***специфичным товаром***, обладая рядом свойств, не присущих обычным товарам, а именно:

- любой интеллектуальный продукт индивидуален по содержанию;
- обладает свойством обмениваемости, но не отчуждается полностью, а лишь заимствуется;
- может быть продан неоднократно, является одновременно объектом нескольких рыночных сделок;
- не исчезает в процессе потребления;
- состоит из материального носителя и идейной части, которая является объектом правовой защиты (в этом смысле говорят о нефизической природе интеллектуального товара);
- не может быть продан «для одного», воплощает в себе общественное начало, свойство быть предметом «для других»;
- не является продуктом «для всех» (невозможно воспользоваться без определенного минимума научных и эстетических знаний, профессиональной компетенции и т. д.);
- характеризуется ничтожными затратами на тиражирование по сравнению с затратами на разработку продукта, производится в условиях быстрой смены номенклатуры и, следовательно, повышенного риска.

2.2.2. Программы для ЭВМ и базы данных как объекты авторского права

Богатство и могущество программиста — это его авторские программы и оригинальные базы данных.

Согласно Закону РФ «О правовой охране программ для электронных вычислительных машин и баз данных» от 23 сентября 1992 г. программное обеспечение (ПО) и базы данных (БД) относятся к объектам авторского права.

Программа для ЭВМ в Законе РФ «О правовой охране программ для электронных вычислительных машин и баз данных» трактуется как объективная форма представления совокупности

данных и команд, предназначенных для функционирования ЭВМ и других компьютерных устройств с целью получения определенного результата. Необходимо отметить, что данное понятие включает также подготовительные материалы, полученные в ходе разработки программы для ЭВМ, и порождаемые данной программой аудиовизуальные изображения. В этой связи возникает вопрос ***о форме представления совокупности данных и команд, попадающей в сферу охраны***: представление данных и команд в виде, непосредственно предназначенном для ввода в ЭВМ, либо представление, требующее определенных преобразований перед вводом в компьютер.

В нормативных документах различаются понятия программы для ЭВМ и программного продукта (рис. 2.3).

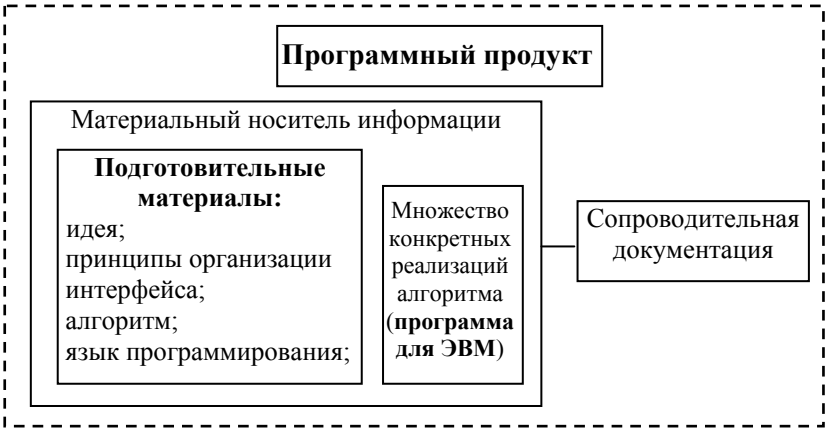

Рис. 2.3. Взаимосвязь программного продукта и программы для ЭВМ

Программа для ЭВМ — это текст, объективированный любым образом — на бумаге, в памяти ЭВМ, в виде изображения на экране монитора. При этом каждое аудиовизуальное произведение, взятое в отдельности (например, заставка к игровой программе), может рассматриваться и как часть программы, и как художественное произведение, и соответственно охраняться как отдельный объект авторского права.

В совокупности программы для ЭВМ и подготовительные материалы называют ***программным продуктом***. Под подготовительными материалами к программе ЭВМ понимается, прежде всего, алгоритм, т. е. идея или математическая формула, на которой основывается программа. Таким образом, программный продукт охватывает не только саму совокупность команд, записанную на том или ином языке программирования, но и алгоритм, на основе которого составляется программа, а также весь описательный и пояснительный материал, который может составляться на любом языке человеческого общения.

Однако математическое обеспечение в отдельности авторским правом охраняться не будет, поскольку его можно рассматривать как своего рода идею, которая авторским правом не охраняется. Так, согласно п. 5 ст. 3 Закона «О правовой охране программ для ЭВМ и баз данных» предоставляемая правовая охрана не распространяется на идеи и принципы, лежащие в основе программы для ЭВМ или базы данных, или какого-либо их элемента, в том числе на идеи и принципы организации интерфейса (взаимодействие программы и самого компьютера) и алгоритма (математическое обеспечение программы), а также языки программирования.

Записанная на материальный носитель (например, дискету) на машинном языке программа для ЭВМ с помощью содержащихся в ней команд обеспечивает заданное функционирование ЭВМ. Таким образом, **объектами охраны** будет сама программа как совокупность команд, написанных на одном из языков программирования и зафиксированных на самых разнообразных носителях (дискетах, бумаге, аудиокассетах и т. д.), и все подготовительные материалы. Охраняется же не идея, заложенная в алгоритм, а конкретная реализация этого алгоритма в виде последовательности операций и действий над этими операциями.

Авторское право на программы для ЭВМ и БД, как и на любой другой объект авторского права, не связано с правом собственности на их материальный носитель. Из этого следует, что ***передача прав на материальный носитель не предполагает передачи авторских прав на сами программы для ЭВМ и базы данных*** (п. 6 ст. 3 Закона «О правовой охране программ для ЭВМ и баз данных»).

База данных в Законе РФ «О правовой охране программ для электронных вычислительных машин и баз данных» трактуется как объективная форма представления и организации совокупности данных (статей, расчетов и т. д.), систематизированных таким образом, чтобы эти данные могли быть найдены и обработаны с помощью ЭВМ. Необходимо разграничивать понятия «база данных» и «информационный ресурс». ***Информационный ресурс*** — это отдельный документ или отдельный массив документов в каких-либо информационных системах (библиотеках, архивах, фондах и т. д.).

Для включения информации в базу данных необходимо, чтобы она находилась в электронной (машиночитаемой) форме, а при включении в информационный ресурс эта информация обязательно должна быть документированной (зафиксированной на материальном носителе с указанием реквизитов согласно стандартам на документы, позволяющим ее идентифицировать). Такое широкое понимание баз данных позволяет использовать охрану практически любой информации в электронной форме (таблицы, гипертекст, подборка файлов и т. д.).

Охрана базы данных характеризуется рядом особенностей. В частности, база данных охраняется независимо от того, является ли информация, на которой она основана, объектом авторского права. **Авторское право на БД**, состоящую из материалов, не являющихся объектами авторского права, принадлежит лицам, создавшим эту базу данных. Если же БД состоит из охраняемых произведений, то авторское право на нее признается лишь при соблюдении авторского права на каждое из входящих в ее состав произведений. Кроме того, авторское право на каждое из произведений, включенных в базу данных, сохраняется, и их использование может осуществляться независимо от этой БД.

Таким образом, составителю базы данных для включения в ее состав любого охраняемого произведения, требуется предварительно получить согласие автора или иного правообладателя на такое произведение. Последние, дав такое разрешение, могут продолжать использование этого произведения по своему усмотрению. При этом авторское право на базу данных не препятствует другим лицам самостоятельно осуществлять подбор и организацию произведений, входящих в эту базу данных.

Базы данных (как и программы для ЭВМ) представляют собой категорию идеального. Они неосязаемы, не потребляемы и **не могут выступать объектом правоотношений без использования материального носителя, на котором они фиксируются**.

При этом программы для ЭВМ и базы данных охраняются авторским правом независимо от назначения, достоинства и вида материального носителя, при соблюдении двух условий:

1) наличие объективной формы существования;

2) творческий характер деятельности автора, в результате которой создается программа для ЭВМ или база данных.

Закон «О правовой охране программ для ЭВМ и баз данных» не разъясняет, что имеется в виду под объективной формой, однако если обратиться к Закону об авторском праве, то в ст. 6 указаны следующие *разновидности объективной формы*:

- письменная (рукопись, машинопись и т. д.);
- устная (публичное произнесение, исполнение);
- изображение (рисунок, чертеж, кино-, теле-, видео- или фотокадр и др.);
- объемно-пространственная (скульптура, макет и т. д.).

Творческий характер деятельности автора, в результате которой создаются программы для ЭВМ и БД, по-разному толкуется юристами. Наиболее распространенным является мнение о том, что творческой признается самостоятельная деятельность по созданию произведения, обладающего новизной.

2.2.3. Возможности правовой охраны программ для ЭВМ и баз данных

Возможности правовой охраны программ для ЭВМ и БД предоставляются институтом авторского права. В отношении этих объектов применяются следующие **формы правовой охраны** (рис. 2.4):

- государственная регистрация с получением свидетельства;
- оценка соответствия продукции/услуг существующим стандартам с получением сертификата соответствия;
- регистрация торгового знака, знаков обслуживания, фирменного наименования с получением свидетельства;

- заключение и регистрация авторских договоров на передачу имущественных прав.

Помимо вышеперечисленного может использоваться международная регистрация программ для ЭВМ и баз данных.

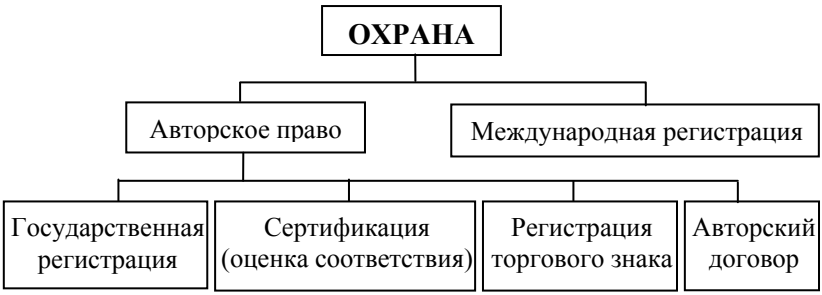

Рис. 2.4. Возможности правовой охраны программ для ЭВМ и БД

Автор программы для ЭВМ и базы данных для оповещения о своих правах может, начиная с первого выпуска в свет программы для ЭВМ или базы данных, использовать знак охраны авторского права, включающий три элемента:

1) букву С в окружности или в круглых скобках — ©;

2) наименование (имя) правообладателя;

3) год первого выпуска программы для ЭВМ или БД.

Кроме того, автор (его правопреемник непосредственно или через своего представителя) по своему желанию может зарегистрировать программы для ЭВМ или базы данных в течение срока действия авторского права.

Регистрация программных средств не является правообразующей и носит факультативный характер, однако ее осуществление значительно облегчает подтверждение факта авторства в спорной ситуации.

К государственным структурам, осуществляющим государственную регистрацию программ для ЭВМ и БД, относятся:

- Федеральная служба по интеллектуальной собственности, патентам и товарным знакам;
- Отраслевой фонд алгоритмов и программ.

В соответствии с Положением «О Федеральной службе по интеллектуальной собственности, патентам и товарным знакам» (утверждено постановлением Правительства РФ от 16 июня 2004 г. № 299, с изменениями, внесенными постановлением Правительства РФ от 22 апреля 2005 г. № 247) **Федеральная служба по интеллектуальной собственности, патентам и товарным знакам** является федеральным органом исполнительной власти, осуществляющим функции по контролю и надзору в сфере правовой охраны и использования объектов интеллектуальной собственности, патентов и товарных знаков и результатов интеллектуальной деятельности, вовлекаемых в экономический и гражданско-правовой оборот, соблюдения интересов Российской Федерации, российских физических и юридических лиц при распределении прав на результаты интеллектуальной деятельности, в том числе создаваемые в рамках международного научно-технического сотрудничества. Федеральная служба по интеллектуальной собственности, патентам и товарным знакам находится в ведении Министерства образования и науки Российской Федерации.

Федеральная служба по интеллектуальной собственности, патентам и товарным знакам осуществляет в отношении программ для ЭВМ и БД следующие полномочия:

- прием заявок на объекты интеллектуальной собственности, их рассмотрение, экспертизу и выдачу в установленном порядке свидетельств Российской Федерации на товарный знак, знак обслуживания, на право пользования наименованием места происхождения товара, свидетельств об официальной регистрации программы для ЭВМ, базы данных, топологии интегральных микросхем;
- регистрация договоров о предоставлении права на товарные знаки, знаки обслуживания, охраняемые программы для ЭВМ, базы данных.

Отраслевой фонд алгоритмов и программ (ОФАП) создан для координации работ в области разработки ПО учебного назначения, аккумулирования информации о разработанном программном обеспечении, пропаганды и внедрения передового опыта в области новых информационных технологий обучения,

информатизации научно-педагогических исследований, информационного обслуживания сферы образования. Пользователями фонда являются авторы и организации-разработчики программного и информационного обеспечения, специалисты системы образования, соискатели сертификата соответствия.

Основными целями отраслевого фонда алгоритмов и программ являются:

- проведение научно обоснованной государственной политики в области разработки, оценки качества, распространения, внедрения и использования программных средств образовательного назначения (ПСОН), в том числе: инструментальных программных средств; прикладных программных средств решения организационно-экономических задач в области образования; педагогических программных средств и т. д.;
- обеспечение разработчиков и потребителей ПСОН полной, достоверной и своевременной информацией по вопросам программного обеспечения вычислительной техники в образовательных учреждениях России;
- обеспечение условий для повышения эффективности создания высококачественного и наиболее полного использования программного обеспечения учебного назначения, в том числе за счет ранее созданных методик, алгоритмов и программ и устранения, тем самым, неоправданного дублирования разработок программного обеспечения и нерационального бюджетного финансирования соответствующих научно-технических проектов.

Регистрация программных средств через федеральный орган исполнительной власти по интеллектуальной собственности (Роспатент) производится путем подачи заявки. Подача заявки регламентирована Приказом Роспатента от 25.02.2003 г. № 25 «О правилах составления, подачи и рассмотрения заявки на официальную регистрацию программы для электронных вычислительных машин и заявки на официальную регистрацию базы данных» (далее Правила). Заявка на регистрацию должна относиться к одной программе для ЭВМ или одной базе данных.

Заявка на регистрацию содержит:

- заявление на официальную регистрацию программы для ЭВМ или БД (прил. 3) с указанием правообладателя, а также ав-

тора, если он не отказался быть упомянутым в качестве такового, и их местонахождения (местожительства);

- депонируемые материалы, идентифицирующие программу для ЭВМ или базу данных, включая реферат;
- документ, подтверждающий уплату регистрационного сбора в установленном размере или основания для освобождения от уплаты этого сбора, а также для уменьшения его размера.

Рассмотрение заявки на регистрацию осуществляется в двухмесячный срок с даты поступления заявки. При соответствии заявки Правилам программа для ЭВМ или база данных вносится в Реестр программ для ЭВМ или Реестр баз данных. Правообладателю направляется уведомление об официальной регистрации и выдается свидетельство об официальной регистрации. Роспатент публикует сведения об официальной регистрации программы для ЭВМ или базы данных в своем официальном бюллетене.

В соответствии с Правилами процесс государственной регистрации состоит из нескольких этапов, представленных на рис. 2.5.

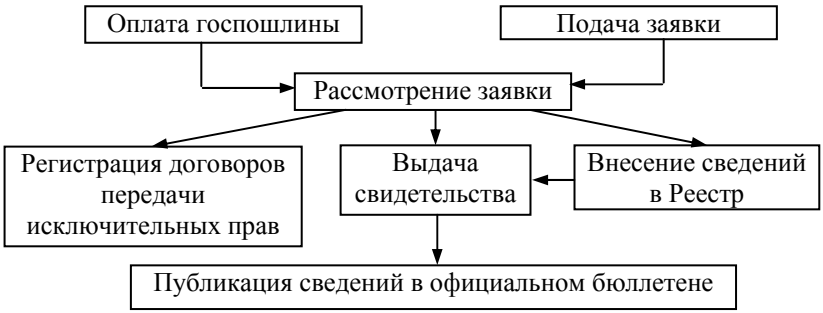

Рис. 2.5. Государственная регистрация программ для ЭВМ и БД и договоров передачи

Регистрация является официальным уведомлением общественности о правах заявителя в отношении данных программных средств. Публикация, помимо прочего, способствует продвижению программных продуктов на рынок, так как может исполнять роль оперативной рекламы. Предоставление права на регистрацию, сопровождающееся депонированием материалов, идентифицирующих регистрируемый объект, внесением программ или

БД, соответственно, в Реестр программ для ЭВМ или Реестр баз данных, выдачей свидетельства об официальной регистрации и публикацией сведений о зарегистрированных объектах в официальном бюллетене федерального органа исполнительной власти по интеллектуальной собственности, позволяет добросовестным правообладателям оповестить общественность о своих правах.

Свидетельство об официальной регистрации программы или БД является инструментом правовой охраны. Однако неправомерно приписывать этому свидетельству свойства других документов, которыми оно не обладает. В частности, оно не является аналогом и не заменяет сертификат соответствия.

Вопрос о правообладателе наиболее актуален для категории служебных произведений. Некоторые работодатели считают достаточным наличие подтверждения факта работы автора в их организации. Согласно п. 1 ст. 12 Закона «*исключительное право на программу для ЭВМ или базу данных, созданные работником (автором) в связи с выполнением трудовых обязанностей или по заданию работодателя, принадлежит работодателю, если договором между ним и работником (автором) не предусмотрено иное*». Таким образом, для признания исключительного права на произведение за работодателем необходимо подтверждение того, что создание этого произведения входило в трудовые обязанности автора или что автору было дано работодателем задание на его создание.

Еще одной возможностью «качественной» охраны прав и подтверждения соответствия созданного продукта стандартам и требованиям на созданный программный продукт или базу данных является сертификация в сфере их разработки.

Сертификация — это процедура письменного подтверждения третьей стороной, никоим образом не зависящей ни от изготовителя продукции, ни от ее потребителя, соответствия продукции требованиям определенных пунктов нормативных документов сферы деятельности. Сертификация программной, как и любой другой продукции, введена в целях защиты пользователей от продукции недоброкачественной. Сертификат предоставляет определенную гарантию соответствия приобретаемого программного продукта всем указанным в нем и в его приложении требованиям нормативных документов, условиям и характеристикам. Не-

редки случаи, когда в процессе сертификации происходит совершенствование, повышение качества программного средства и в части уточнения нормативных требований, и в части решаемых проектных задач, и в части назначения и области применения.

В современных условиях основным законодателем в области сертификации продукции является «Общеевропейский» рынок, и сейчас наметилась тенденция, когда оценка качества на соответствие международным стандартам рассматривается как обязательное условие успешной интеграции с мировым информационным пространством. Так, Правительством РФ внесены изменения в порядок подготовки и заключения государственных контрактов на закупку и поставку продукции для федеральных государственных нужд. Из него следует, что сертификация по Международному стандарту качества **ISO 9000**, где речь идет о качестве процессов, которые создают товар или обеспечивают выполнение услуги, становится необходимым условием для получения Государственного заказа. В настоящий момент эти стандарты признаны практически всеми странами мира. В России действует отечественная (аутентичная) версия ГОСТ Р серии 9000, введенная в действие с 31.08.2001 г. Для компаний, занимающихся разработкой ПО, существуют отдельные спецификации на базе ISO 9000. Основу сертификации составляют:

- национальные стандарты;
- международные стандарты;
- стандарты организаций;
- системы добровольной сертификации.

Нормативную базу сертификации программных средств составляют базовые российские и международные стандарты в областях управления, информационных технологий, открытых систем, а также базовые государственные стандарты на создание, документирование и испытание автоматизированных систем. Сертификация программного обеспечения на соответствие требованиям Госстандарта России традиционно проводится на основании нижеприведенных стандартов:

- ГОСТ Р ИСО\МЭК 9126-93 (по оценке программной документации);
- ГОСТ Р ИСО\МЭК ТО 9294-93 (по документированию ПО);

- ГОСТ Р ИСО 9127-94 (по оформлению документации пользователя);
- ГОСТ 28195-89 (по оценке качества программных средств);
- ГОСТ 28806-90 (по определению понятий в области качества ПС).

Выделяют ***обязательную*** и ***добровольную*** сертификацию. Органами, занимающимися сертификацией программной продукции, соответственно, могут являться:

– при обязательной сертификации — орган по сертификации и (или) испытательная лаборатория (центр), аккредитованные в порядке, установленном Правительством Российской Федерации («Россертификация» — проведение аккредитации для сертификации (Госстандарт), «Ростехрегулирование» — разработка стандартов);

– при добровольной сертификации создается система добровольной сертификации, организованная юридическим лицом и (или) индивидуальным предпринимателем или несколькими юридическими лицами и (или) индивидуальными предпринимателями («Росинфосерт» — система добровольной сертификации средств и систем в сфере информатизации, «Инкомтехсерт» — система добровольной сертификации информационно-коммуникационных технологий в образовании). В России на сегодня зарегистрировано более 200 систем добровольной сертификации.

Добровольное подтверждение соответствия осуществляется по инициативе заявителя на условиях договора между заявителем и органом по сертификации. Добровольное подтверждение соответствия может осуществляться для установления соответствия национальным стандартам, стандартам организаций, системам добровольной сертификации, условиям договоров. Система добровольной сертификации может быть зарегистрирована федеральным органом исполнительной власти по техническому регулированию. Объекты сертификации, сертифицированные в системе добровольной сертификации, могут маркироваться ***знаком соответствия*** системы добровольной сертификации. Порядок применения знака соответствия устанавливается правилами соответствующей системы добровольной сертификации.

Обязательное подтверждение соответствия проводится только в случаях, установленных соответствующим техническим регламентом, и исключительно на соответствие требованиям технического регламента. ***Объектом обязательного подтверждения соответствия*** может быть только продукция, выпускаемая в обращение на территории Российской Федерации. Форма и схемы обязательного подтверждения соответствия могут устанавливаться только техническим регламентом с учетом степени риска недостижения целей технических регламентов. Схемы сертификации, применяемые для определенных видов продукции, устанавливаются соответствующим техническим регламентом.

Отметим ***основные преимущества для сертифицированной программной продукции с точки зрения разработчика***:

- определенное признание и повышение имиджа организации-разработчика в регионе и отрасли;
- наличие обязательного условия для получения государственного или любого другого заказа в рамках целевых программ, которые финансируются из федерального бюджета или города;
- более широкие возможности сотрудничества в совместных работах и проектах с российскими и иностранными организациями, имеющими широкие возможности для интеграции, работы в корпоративном масштабе;
- конкурентное преимущество при участии, например, в тендерных торгах.

С точки зрения потребителя или заказчика преимущество при приобретении сертифицированной программной продукции состоит в том, что предоставляется определенная гарантия того, что приобретаемый ПП соответствует всем указанным в нем и его приложении требованиям нормативных документов, условиям и характеристикам.

Одним из наиболее эффективных способов охраны названия, присвоенного программе ЭВМ, оригинального изображения представления программного продукта является **использование товарных знаков**. Права на товарные знаки и знаки обслуживания регулируются Законом РФ «О товарных знаках, знаках обслуживания и наименованиях мест происхождения товаров» от 23.09.92, вступившим в силу 17.10.92. Товарный знак

может быть бессрочным, и в этом состоит его преимущество перед патентами и объектами авторского права.

Товарный знак — это символ, позволяющий отличить продукцию одного производителя от другого или услугу одной фирмы от другой. Если этого не сделать, плодами «раскрутки» может воспользоваться другая фирма, например, выпустив товар под аналогичной торговой маркой худшего качества.

Осуществив целый комплекс мер по правовой охране программного продукта, субъекты правоотношений (разработчик и потребитель) могут смело вступать во взаимные правоотношения по поводу использования программного обеспечения и баз данных, осознавая при этом, что они имеют дело с охраняемым и защищаемым законодательством объектом авторских прав и добросовестным контрагентом, деятельность которого урегулирована законодательно посредством договора (рис. 2.6).

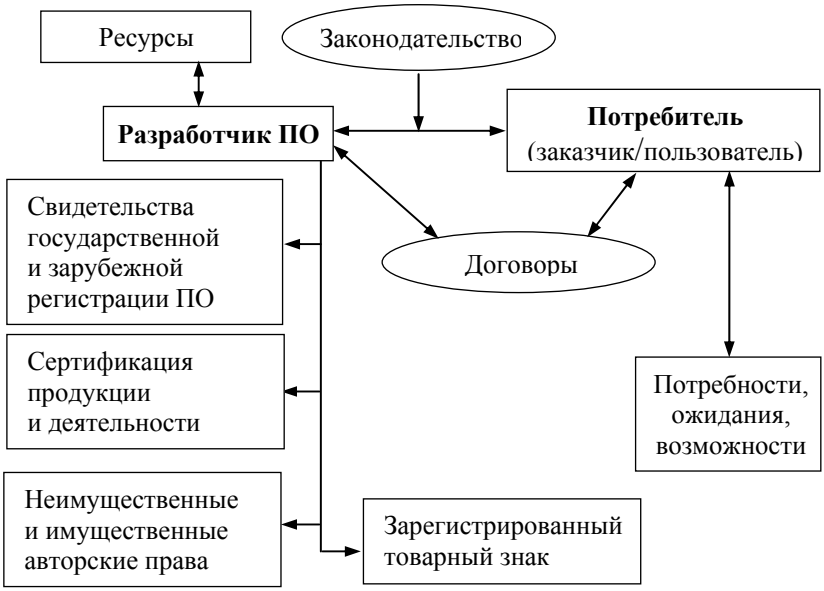

Рис. 2.6. Взаимоотношение «Разработчик-потребитель»

В этом процессе одна сторона (разработчик) предоставляет ПО, соответствующее основным требованиям, ожиданиям и возможностям другой стороны (потребителя), а также всем техническим нормам, требованиям по качеству, защищенности, авторской принадлежности и т. п.

Согласно ст. 30 и 33 Закона «Об авторском праве и смежных правах» имущественные права на объекты авторского права могут передаваться по **авторскому договору**, в котором регламентируются отношения между автором (иным правообладателем) и приобретателем, возникающие при передаче (или взятии обязательств на передачу) имущественных прав на произведение.

Для современного российского авторского права характерен принцип свободы авторского договора. В законодательстве закрепляются лишь возможные типы авторских договоров, а также указываются условия, которые должны быть в обязательном порядке согласованы сторонами.

Авторские договоры о передаче имущественных прав на программные средства классифицируются следующим образом (рис. 2.7). ***По содержанию передаваемых прав*** выделяют:

а) авторский договор о передаче исключительных прав:

– о передаче исключительных прав (исключительная лицензия);

– о полной уступке всех имущественных прав (полная лицензия);

б) авторский договор о передаче неисключительных прав, т. е. неисключительная (простая) лицензия.

Авторский договор по передаче исключительных прав может содержать следующие разделы: название договора; преамбулу; определение терминов; предмет договора; порядок поставки и сопровождения, обучения персонала лицензиата; гарантии и ответственность; права сторон на последующие кодификации; доступ к исходному коду; порядок защиты передаваемых прав и использования программных средств третьими лицами; возможность переуступки договора платежи; сборы и налоги; информацию и отчетность; обеспечение конфиденциальности; рекламу; порядок разрешения споров; срок действия и условия расторжения договора; особые условия; реквизиты сторон; приложения.

Рис. 2.7. Виды авторских договоров

При продаже экземпляров программ массовым пользователям применяется особая форма авторского договора — так называемая **«оберточная лицензия»**, предусмотренная ст. 32 закона об авторских и смежных правах и ст. 14 закона о правовой охране программ для ЭВМ и БД, являющаяся разновидностью конклюдентных сделок. Суть такой формы в том, что программное обеспечение распространяется путем продажи или передачи на определенной комплектации, в которую входит собственно программа и лицензионное соглашение (либо в письменном виде на внешней стороне носителя, либо в электронном виде как соответствующий текстовый или гипертекстовый файл). Факт начала использования программы или наступление определенного срока с момента начала использования означает принятие условий соглашения пользователем.

Договор на передачу неисключительных прав (неисключительная, пользовательская лицензия) не исключает возможности автора (правообладателя) заключать договоры с другими пользователями, а также не дает права пользователю запрещать другим лицам использование указанной программы для ЭВМ или БД.

По способу использования передаваемых прав выделяют:

а) договор на выпуск в свет (опубликование) программы для ЭВМ или базы данных;

б) договор на воспроизведение (изготовление одного или более экземпляров) программы для ЭВМ или базы данных (полное или частичное) в любой форме, любыми способами;

в) договор на модификацию программы для ЭВМ или базы данных, в том числе перевод программы с одного языка на другой;

г) договор на распространение программы для ЭВМ или базы данных путем предоставления доступа к воспроизведенной в любой материальной форме программы для ЭВМ или базы данных, в том числе сетевыми и иными способами, а также путем продажи, проката, сдачи в наем, предоставления в займы, включая импорт для любой из этих целей;

д) договор на иное использование программы для ЭВМ или БД.

Содержание авторского договора определено в ст. 31 Закона РФ «Об авторском праве и смежных правах». Так, в соответствии с п. 1 указанной статьи он должен предусматривать существенные и желательные условия (рис. 2.8).

Существенными являются такие условия, без включения которых договор считается незаключенным. К ним относятся:

- предмет авторского договора (предметом авторского договора не могут быть права на использование произведения, неизвестные на момент заключения договора);
- территория, на которую передается право (при отсутствии в авторском договоре условия о территории, на которую передается право, действие передаваемого по договору права ограничивается территорией Российской Федерации);
- способы использования программ для ЭВМ и БД (выпуск в свет, воспроизведение, распространение, адаптация, модификация, декомпилирование);
- срок действия авторского договора (при отсутствии в авторском договоре условия о сроке, на который передается право, договор может быть расторгнут автором по истечении пяти лет с даты его заключения, если пользователь будет письменно уведомлен об этом за шесть месяцев до расторжения договора);
- размер вознаграждения, порядок и сроки его выплаты.

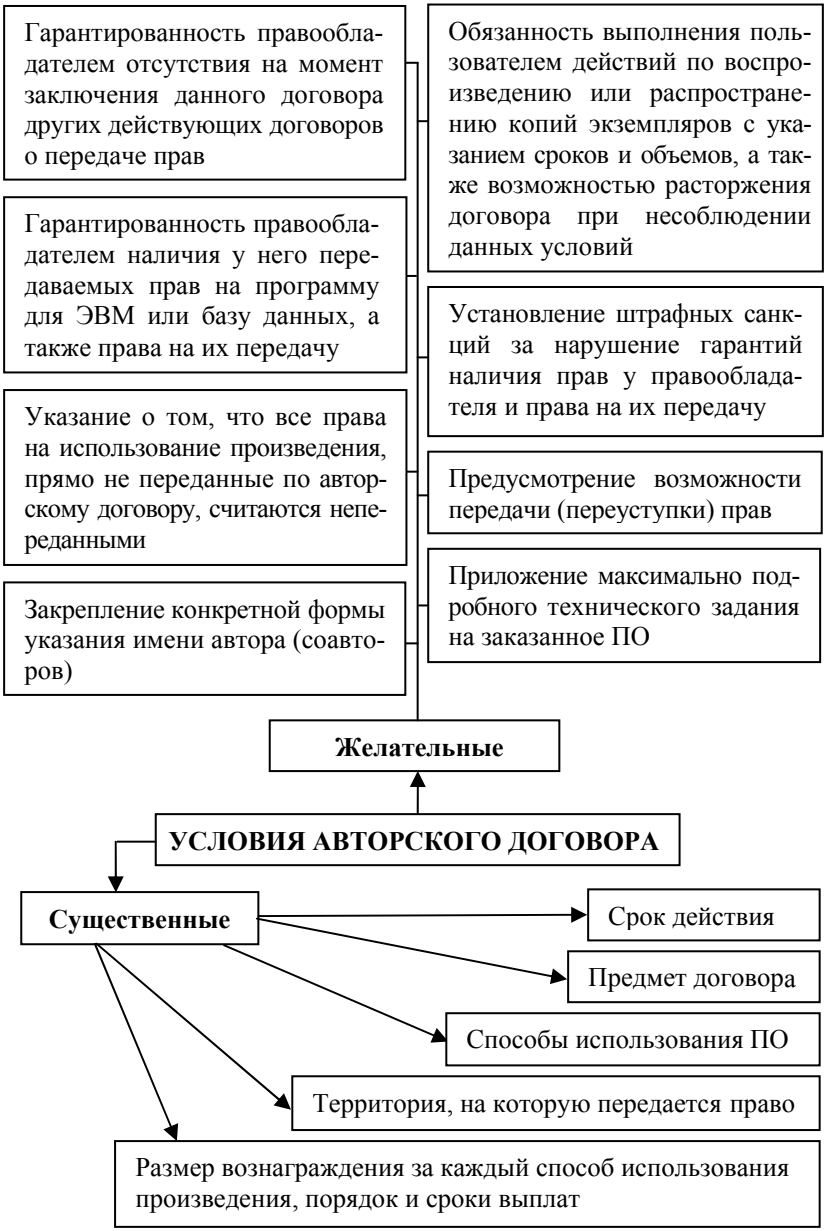

Рис. 2.8. Условия авторского договора

2.2.4. Юридическая ответственность за правонарушения

Незнание законов не освобождает от ответственности.
Соблюдение законов — залог успешного ведения бизнеса.

В настоящее время практически сформирована основная нормативная база по предупреждению и пресечению правонарушений. В информационной сфере за совершение правонарушений и преступлений предусматриваются различные виды ответственности: **гражданско-правовая:** ***имущественная*** (возмещение вреда, компенсация); ***дисциплинарная*** (замечание, выговор, увольнение); ***материальная*** (возмещение причиненного вреда); **административная** (штраф, предупреждение, конфискация); **уголовная** (штраф, лишение свободы).

Основанием для возникновения юридической ответственности является совершенное субъектом (участником) информационных правоотношений правонарушение в информационной сфере (табл. 2.6).

Таблица 2.6

Виды и основания ответственности

Основания	Виды ответственности				
	Уголовная	Административная	Имущественная	Дисциплинарная	Материальная
Особокрупный ущерб* Крупный ущерб**	да	-	да	-	-
Виновное деяние	да	да	да	да	да
Общественно-опасное деяние	да	-	-	-	-
Противоправное деяние	да	да	да	-	-
Ущерб, вред	-	да	да	-	да

* — более 500 МРОТ; ** — более 100 МРОТ

Российская правовая система предусматривает три вида ответственности физических лиц за правонарушения (рис. 2.9):

1) административная;
2) гражданско-правовая;
3) уголовная.

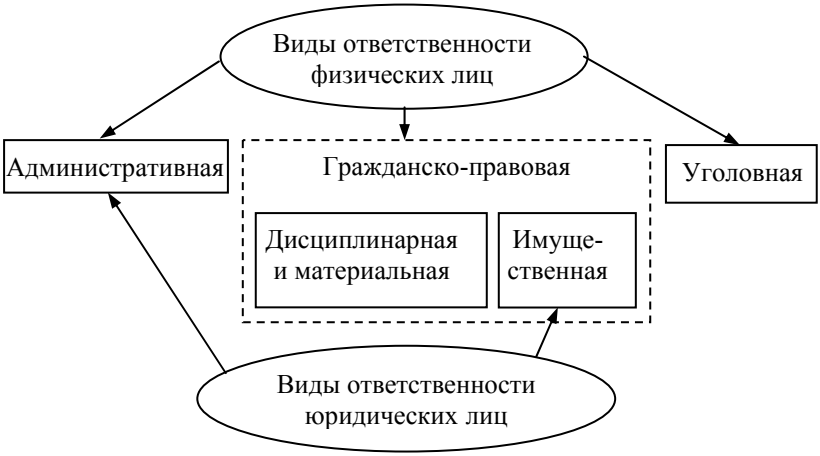

Рис. 2.9. Виды юридической ответственности

Уголовная ответственность

В тех случаях, когда правонарушения в информационной сфере носят систематический злостный характер, виновные привлекаются к уголовной ответственности в соответствии с Уголовным кодексом РФ. Субъектом преступления является физическое лицо (человек), вменяемое и достигшее установленного законом возраста, с которого начинается уголовная ответственность. За информационные преступления по общему правилу уголовной ответственности подлежат лица, достигшие 16-летнего возраста ко времени совершения преступления. Исключением является преступление, предусмотренное ст. 207 (заведомо ложное сообщение об акте терроризма), за совершение которого подлежат уголовной ответственности лица, достигшие 14-летнего возраста (ст. 20 УК РФ).

В действующем Уголовном кодексе РФ из всего объема информационных отношений, подлежащих специальной охране, выделены отношения, возникающие в области компьютерной информации, где содержатся нормы, объявляющие общественно опасными деяниями конкретные действия в сфере компьютерной информации и устанавливающие ответственность за их совершение.

К уголовно наказуемым отнесены:

- неправомерный доступ к компьютерной информации (ст. 272);
- создание, использование и распространение вредоносных программ для ЭВМ (ст. 273);
- нарушение правил эксплуатации ЭВМ, системы ЭВМ или их сети (ст. 274);
- присвоение авторства (плагиат);
- незаконное использование объектов авторских прав.

Правовая охрана программ для ЭВМ или баз данных может также осуществляться на основе норм о преступлениях в сфере компьютерной информации (гл. 28 УК РФ).

Административная ответственность

В соответствии со ст. 2.1 Кодекса об административных правонарушениях (КоАП) РФ **административным правонарушением** признается противоправное, виновное действие (бездействие) ***физического*** или ***юридического лица***, за которое КоАП РФ или законами субъектов Российской Федерации об административных правонарушениях установлена административная ответственность. Согласно ст. 3.2 КоАП РФ за совершение административных правонарушений могут устанавливаться и применяться следующие административные наказания:

1) предупреждение;

2) административный штраф;

3) возмездное изъятие орудия совершения или предмета административного правонарушения;

4) конфискация орудия совершения или предмета административного правонарушения;

5) лишение специального права, предоставленного физическому лицу;

6) административный арест (п.п. 1–4 КоАП РФ);

7) административное выдворение за пределы Российской Федерации иностранного гражданина или лица без гражданства;

8) дисквалификация.

Дисциплинарная и материальная ответственность

Дисциплинарную ответственность за проступки (административные правонарушения) в информационной сфере несут

работники предприятий, учреждений, организаций в соответствии с положениями, уставами, правилами внутреннего трудового распорядка и другими нормативными актами. Ст. 192 Трудового кодекса РФ от 30 декабря 2001 г. установлено, что за совершение дисциплинарного проступка, т. е. неисполнение или ненадлежащее исполнение работником по его вине возложенных на него трудовых обязанностей, работодатель имеет право применить следующие дисциплинарные взыскания: замечание, выговор, увольнение по соответствующим основаниям.

Порядок применения дисциплинарных взысканий определяется **трудовым законодательством** (ст. 193 ТК РФ).

В соответствии с трудовым законодательством за правонарушения в информационной сфере предусмотрена и **материальная ответственность** — имущественная ответственность работников, по вине которых предприятие, учреждение, организация понесли расходы по возмещению вреда, причиненного информационным правонарушением. Подробно порядок и условия материальной ответственности рабочих и служащих регламентированы трудовым законодательством. В соответствии со ст. 232 ТК РФ сторона трудового договора (в данном случае работник), причинившая ущерб другой стороне, возмещает этот ущерб в соответствии с ТК РФ и иными федеральными законами.

Имущественная ответственность

Имущественная ответственность наступает в тех случаях, когда в результате несоблюдения соответствующих норм информационного права причиняется вред предприятиям, учреждениям, организациям и гражданам. Такую ответственность несут как юридические, так и физические лица. В соответствии с Гражданским кодексом РФ имущественный вред возмещается виновной стороной в полном объеме (ст. 1064 ГК РФ).

Возмещение вреда производится добровольно или по решению суда либо арбитражного суда. При этом ущерб подлежит возмещению понесшей его стороне, в том числе и гражданам (их здоровью, имуществу, собственности). С согласия сторон или по решению суда вред может быть возмещен путем возложения на ответчика обязанности по восстановлению нарушенных прав и свобод за счет его сил и средств.

В соответствии со ст. 151 ГК РФ субъект (истец) может требовать компенсации морального вреда (физические или нравственные страдания), причиненного гражданину действиями, нарушающими его личные неимущественные права либо посягающими на принадлежащие гражданину другие нематериальные блага.

Гражданским кодексом РФ закреплены правила защиты деловой репутации юридического лица. Согласно ст. 1101 ГК РФ компенсация морального вреда осуществляется в денежной форме.

Закрепленные в гражданском законодательстве нормы, регулирующие защиту нарушенных авторских прав, учитывают две возможные группы нарушений: нарушение авторских личных неимущественных прав и нарушение исключительных правомочий на использование произведений, охраняемых авторским правом.

Нарушение личных неимущественных прав авторов не всегда затрагивает их имущественные права, и, наоборот, соблюдение личных неимущественных прав может повлечь нарушение имущественных прав (например, автор не получает вознаграждения за использование своего произведения).

Если было нарушено право на авторство или право на авторское имя, то автор может потребовать признания авторства. Признание выражается с помощью специального сообщения в печати. В публикации о допущенном нарушении должно быть указано, где и когда было допущено нарушение и каким образом.

В случае нарушения личных неимущественных прав автор также может потребовать признания прав и компенсации морального вреда. Что касается имущественных прав, то речь идет о нарушении исключительного права на использование произведения. При этом следует иметь в виду, что право требовать защиты нарушенного права имеют не только авторы, но и другие правообладатели.

В ст. 48 Закона об авторском праве говорится о контрафакции, т. е. самовольном и незаконном изготовлении и распространении экземпляров произведения без согласия правообладателя.

Согласно п. 2 ст. 50 Закона об авторском праве может быть наложен арест на контрафактные (изготовленные с нарушением авторских прав) экземпляры до рассмотрения дела по существу.

В случае нарушения имущественных прав автор может потребовать возмещения убытков. Убытки правообладателя на произведение могут включать и упущенную выгоду, т. е. те доходы, которые мог бы получить автор или иной правообладатель при правомерном использовании произведения.

Кроме того, помимо возмещения убытков, взыскания дохода или выплаты компенсации в твердой сумме суд может вынести решение о конфискации контрафактных экземпляров произведения, а также материалов и оборудования, используемых для их воспроизведения.

Что касается «пиратства», то оно характеризуется злонамеренностью действий нарушителя, особым масштабом незаконного использования произведений и, как правило, организованным характером таких правонарушений.

Автор и иные правообладатели программ для ЭВМ или баз данных вправе требовать соблюдения ряда мер, представленных на рис. 2.10.

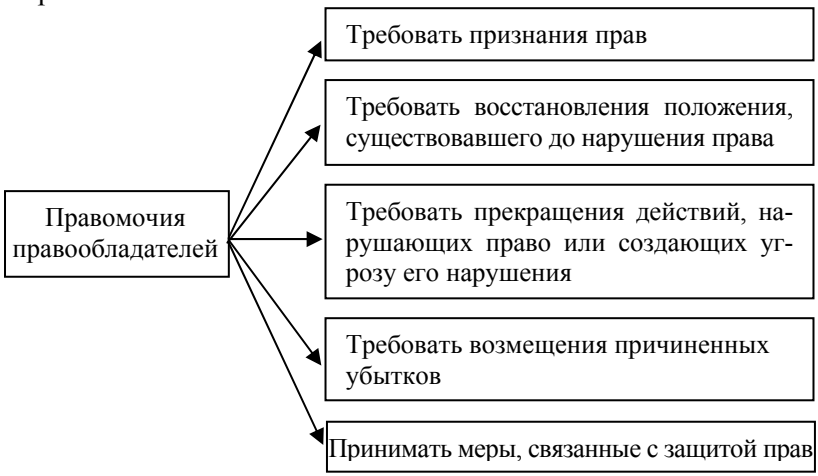

Рис. 2.10. Правомочные действия авторов по защите своих прав

Суть этих мер состоит в следующем:

– признание прав (возможно с опубликованием за счет нарушителя сообщения в печати по поводу принадлежности авторства, исключительных прав);

– восстановление положения, существовавшего до нарушения прав (уничтожения контрафактных экземпляров программного обеспечения или баз данных, приведения баз данных в первоначальный вид, до внесения в нее изменений, не согласованных с правообладателем и т. п.);

– прекращение действий, нарушающих право или создающих угрозу его нарушения (запрещения продажи программ для ЭВМ или баз данных, созданных с нарушением права на имя);

– возмещение нарушителем обладателю исключительных прав убытков или взыскание дохода, полученного нарушителем вследствие нарушения авторских и смежных прав или выплаты компенсации, определяемой судом или арбитражным судом;

– принятие иных предусмотренных законодательными актами мер, связанных с защитой их прав (ст. 12 ГК РФ): признание недействительным договора, компенсация морального вреда и пр..

Контрольные вопросы

1. Кому выгодно соблюдение стандартов: заказчику или разработчику? Существует ли мотивация для соблюдения стандартов?

2. Как выбрать минимально необходимый комплекс нормативных документов, удовлетворяющих требованиям заказчика?

3. Какие основные моменты регламентирует ГОСТ 19?

4. Какие основные моменты регламентирует ГОСТ 34? Каким образом разработчику избавить себя от написания излишней, с его точки зрения, документации?

5. Покажите сходство и различия стандарта ISO 12207.1995 с ГОСТами 19 и 34.

6. Заказчик всегда прав. Как в этом случае организовать процесс разработки, максимально соблюдая требования ГОСТов?

7. В каких случаях использовать стратегию мягкого внедрения, а в каких жесткого?

8. Предложите последовательность этапов проведения работ и их содержание при сравнении качества вашего программного продукта с аналогами.

9. Как документально оформить и закрепить авторское право при тиражном программном продукте?

10. Как автору следует выстраивать взаимоотношения с работодателем при разработке программного продукта «под заказ»?

13. Назовите основные особенности программного продукта как объекта интеллектуальной собственности и проблемы его защиты.

3. ТЕХНИКО-ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ ТРУДОЕМКОСТИ И ДОГОВОРНОЙ ЦЕНЫ ПРОГРАММНЫХ СИСТЕМ

3.1. Основные положения

Процесс разработки программных продуктов, как и любых продуктов человеческой деятельности, подлежит нормированию: объем работы, продолжительность работы, стоимость работы.

Под ***технико-экономическим обоснованием*** договорной цены (***стоимости***) программной системы будем понимать методику оценивания трудовых, временных и финансовых ресурсов по созданию программной системы, соответствующей требованиям заказчика.

В основу определения требуемых объемов ресурсов должны быть положены:

- совокупность бизнес-процессов, реализуемых в будущей программной системе и их относительная важность (приоритет) для заказчика;
- требования к функциональной полноте и качеству реализации каждого бизнес-процесса.

В качестве основных показателей оценки стоимости программной системы используются:

- сложность (размеры) программной системы;
- трудозатраты на разработку;
- длительность разработки программной системы в целом и ее отдельных этапов;
- численность и квалификация специалистов, привлекаемых к созданию программной системы;
- фонд оплаты труда специалистов на создание программной системы в целом и по конкретному этапу жизненного цикла;
- прочие прямые затраты и накладные расходы, связанные с созданием программной системы.

В основу определения размеров программной системы положено понятие ***сложности,*** под которым понимается количество элементов ПС (программных компонентов, файлов, входных и выходных документов) и взаимосвязей между ними.

Под термином «***трудозатраты»*** будем понимать суммарный объем труда специалистов для создания программного продукта. В качестве универсального измерителя трудозатрат используется показатель «***человеко-месяц».*** Каждый человеко-месяц содержит 160 человеко-часов (четыре недели по пять рабочих дней с длительностью 8 часов).

Длительность разработки и численность специалистов определяются на основе трудозатрат и нормативной производительности труда программиста, выражаемой в количестве строк кода, создаваемых программистом в единицу времени. При этом выделяются следующие этапы жизненного цикла ПС:

- анализ предметной области и разработка требований к программной системе;
- проектирование;
- программирование;
- тестирование и комплексные испытания;
- опытная эксплуатация.

В реализации проекта на каждом этапе принимает участие три группы специалистов:

1) руководитель проекта, системные аналитики;

2) непосредственные разработчики программных систем и специалисты по комплексированию;

3) технический персонал, обеспечивающий тестирование, документирование и опытную эксплуатацию программного обеспечения.

Фонд оплаты труда на реализацию проекта определяется, исходя из производительности труда специалиста и согласованной базовой месячной заработной платы.

Все нормативы и другие статистические данные, используемые в методиках технико-экономического обоснования стоимости ПС, основываются на статистических данных, обобщающих зарубежный и российский опыт разработки программных систем [8].

При этом по уровню сложности все множество программных систем (ПС) следует разбить на три типа:

1) комплексные программные системы (КПС) и технологии, отдельные части которых реализованы на различных платформах; территориально распределенные программные системы и технологии; системы автоматизированного либо автоматического управления, функционирующие в режиме реального времени;

2) информационно-справочные системы (ИПС), обеспечивающие информационную поддержку основных бизнес-процессов организации с большим количеством разновидностей исходной информации;

3) инженерные и научно-технические пакеты программ (ППП) и технологий, характеризующихся четко заданным алгоритмом обработки и малыми объемами исходных данных.

При формировании требований к программной системе необходимо использование стандарта ISO/IEC 9126-1:2001, согласно которому системы первого и второго типа должны удовлетворять следующим требованиям:

- архитектура системы должна соответствовать текущим и перспективным целям создаваемой системы;
- в структуре и компонентах следует предусматривать обеспечение максимально возможной сохранности капитальных вложений заказчика в аппаратные и программные средства, а также в базы данных;
- для обеспечения перспективы развития системы должны быть предусмотрены возможность интеграции компонентов и мобильность ПС на различные аппаратные и операционные платформы на основе концепции и стандартов Открытых систем;
- архитектура информационной системы должна быть достаточно гибкой и допускать простое, без коренных структурных изменений, развитие и наращивание функций и ресурсов системы в соответствии с расширением сфер и задач ее применения;
- необходимо обеспечить эффективное использование ресурсов системы и минимизировать интегральные затраты на обработку данных в типовых режимах ее функционирования с учетом текущих эксплуатационных затрат и капитальных вложений в создание ПС;

- следует обеспечить комфортный, максимально упрощенный доступ конечных пользователей к управлению и результатам функционирования системы на основе современных графических средств и наглядных пользовательских интерфейсов.

3.2. Методы определения технико-экономических показателей программной системы

3.2.1. Типовые нормы времени на программирование задач для ЭВМ

В прошлом опыте всегда можно найти рациональное зерно.

В период с 1987 по 1993 гг. в Советском Союзе при планировании трудоемкости (трудозатрат) разработки проекта по созданию автоматизированной системы управления предприятием использовались типовые нормы времени на программирование задач на ЭВМ, утвержденные Постановлением Государственного комитета СССР по труду и социальным вопросам № 454/22-70 от 27.07.1987 г. Материалы Постановления, раскрывающие содержание этих норм, можно найти в одной из баз данных нормативных документов «Кодекс», «Гарант», «Консультант».

Основу нормирования согласно данному Постановлению составляют нижеприведенные положения.

Состав подсистем и комплексов задач АСУ определяется общесоюзным классификатором подсистем и комплексов задач (ОКПКЗ). Классификатор содержит подсистемы по следующим направлениям деятельности:

- технико-экономическое планирование;
- оперативное управление;
- управление материально-техническим снабжением;
- управление сбытом продукции;
- управление финансовой деятельностью, бухгалтерский учет;
- управление организацией труда и заработной платой;
- управление кадрами;

- управление качеством;
- управление технической подготовкой производства;
- совершенствование документооборота и контроль исполнения документов.

Трудозатраты (в человеко-днях) по каждой из подсистем устанавливаются нормативно на каждый этап жизненного цикла создания АСУ: техническое задание, эскизный проект, технический проект, рабочий проект, внедрение. При этом выделяются две группы участвующих в проекте специалистов: постановщики задач и специалисты по разработке программного обеспечения.

Трудозатраты на выполнение работ на стадиях «Техническое задание» и «Эскизное проектирование» зависят от степени новизны проекта и задаются нормативно для каждой подсистемы (табл. 3.1).

Таблица 3.1

Затраты времени на стадии «Техническое задание»

Наименование подсистемы	Трудозатраты в зависимости от степени новизны проекта			
	А	Б	В	Г
1. Подсистема технико-экономического планирования	$x_{1А}$	$x_{1Б}$	$x_{1В}$	$x_{1Г}$
2. Подсистема оперативного управления	$x_{2А}$	$x_{2Б}$	$x_{2В}$	$x_{2Г}$
. . .	. . .	. . .	. . .	. . .
n. Подсистема совершенствования документооборота и контроля исполнения документов	$x_{nА}$	$x_{nБ}$	$x_{nГ}$	$x_{nГ}$

В случае если в работе участвуют постановщики задач и программисты, удельный вес их трудозатрат составляет 0,65 и 0,35.

Трудоемкость работ по разработке технического и рабочего проектов и внедрения зависит от количества входной и выходной информации и задается либо нормативно для каждой подсистемы отдельно для постановщиков задач и программистов (табл. 3.2), либо вычисляется по формуле

$$T = k\Phi_1^{\alpha_1}\Phi_2^{\alpha_2}, \qquad (3.1)$$

где k, α_1, α_2 — коэффициенты, значение которых зависит от типа подсистемы;

Φ_1, Φ_2 — количество макетов входных и выходных форм соответственно.

Таблица 3.2

Объем входной и выходной информации

Количество разновидностей форм входной информации	Количество разновидностей форм выходной информации				
	1	2	3–4	...	31–42
1	x_{11}	x_{12}	x_{13}	...	x_{19}
2	x_{21}	x_{22}	x_{23}	...	x_{29}
⋮	⋮	⋮	⋮	⋮	⋮
20	$x_{20;1}$	$x_{20;2}$	$x_{20;3}$	...	$x_{20;9}$

В случае если стадии технического и рабочего проекта совмещаются, разрабатывается техно-рабочий проект, для которого трудоемкость разработки составляет 85 % от трудоемкости технического проекта и 100 % от рабочего проекта.

Нормы времени, указанные в нормативных таблицах, разработаны для комплексов задач (задачи) степени новизны В, группы сложности алгоритма решения 3 при использовании переменной информации (о группах сложности алгоритма и степени новизны см. ниже). При этом объем входной информации не должен превышать 50 тыс. документострок.

Для определения трудоемкости разработки задач с другими характеристиками следует пользоваться поправочными коэффициентами.

Для учета специфики реализации и размерности подсистем и задач вводятся поправочные коэффициенты на нормативные трудозатраты. Численные значения коэффициентов определяются в зависимости от следующих факторов:

количества разновидностей форм входной информации (макетов входной информации);

количества разновидностей форм выходной информации (формы печатных документов и информации, переносимой на машинные носители). Если в процессе разработки программ формируются данные, которые используются другими программами этого же комплекса задач (задачи), то такие наборы данных не входят в число форм входной и выходной информации;

степени новизны комплекса задач (задачи). Предусмотрены четыре степени новизны разрабатываемых комплексов задач (задачи):

А — разработка комплекса задач (задачи), предусматривающая применение принципиально новых методов разработки, проведение научно-исследовательских работ;

Б — разработка типовых проектных решений, оригинальных задач и систем, не имеющих аналогов;

В — разработка проекта с использованием типовых проектных решений при условии их изменения; разработка проектов, имеющих аналогичные решения;

Г — привязка типовых проектных решений;

сложности алгоритма. Сложность алгоритма представлена тремя группами:

1 — алгоритмы оптимизации и моделирования систем и объектов;

2 — алгоритмы учета, отчетности, статистики поиска;

3 — алгоритмы, реализующие стандартные методы решения, а также не предусматривающие применения сложных численных и логических методов;

вида используемой информации (переменной информации (ПИ), нормативно-справочной информации (НСИ), банка данных (БД)), *ее объема* и *режимов обработки*;

сложности контроля входной и выходной информации. Сложность организации контроля входной и выходной информации представлена следующими группами:

1 — входные данные и документы разнообразного формата и структуры (контроль осуществляется перекрестно, т. е. учитывается связь между показателями различных документов);

2 — входные данные и документы однообразной формы и содержания (осуществляется формальный контроль);

3 — печать документов сложной многоуровневой структуры разнообразной формы и содержания;

4 — печать документов однообразной формы и содержания, вывод массивов данных на машинные носители;

языка программирования. Нормы времени на разработку рабочего проекта даны с использованием языка программирования высокого уровня [при использовании языков низкого уровня (типа Ассемблер) применяется коэффициент 1,15; при использовании языковых описателей, построителей отчетов и различных интерпретаторов следует применять коэффициент 0,8 (по превалирующему языку)];

объема входной информации. Поправочные коэффициенты на объем входной информации вводятся для следующих интервалов: до 50 тыс. документострок (д.с.), до 100 тыс. д.с., до 200 тыс. д.с., свыше 200 тыс. д.с.;

степени использования типовых проектных решений (пакетов прикладных программ), типовых проектов, типовых программ, стандартных модулей. Если при разработке проекта используются типовые проектные решения (пакеты прикладных программ), типовые проекты, типовые программы и стандартные модули, то нормы времени уменьшаются на соответствующие коэффициенты.

Общая трудоемкость проекта определяется по формуле

$$T = \sum_{i=1}^{n} t_i^{\text{н}} \prod_{j=1}^{m} K_{ij}, \qquad (3.2)$$

где i — количество стадий жизненного цикла проекта;

$t_i^{\text{н}}$ — суммарная нормативная трудоемкость i-й стадии;

j — количество поправочных коэффициентов, используемых на i-й стадии;

K_{ij} — численное значение j-го поправочного коэффициента.

Требуемая численность специалистов при заданном директивном сроке реализации проекта определяется по формуле

$$\text{Ч} = \frac{T}{\Phi_{\text{пл}}}, \qquad (3.3)$$

где $\Phi_{\text{пл}}$ — плановый фонд работы одного специалиста, чел.-дней.

Несмотря на то что срок действия вышеназванного Постановления ограничен 1993 годом, основные идеи, изложенные в нем, в том числе набор степенных функций вычисления нормативной трудоемкости, можно использовать для первоначального (предварительного) обоснования трудозатрат на создание программных систем информационной поддержки основных бизнес-процессов промышленных компаний.

3.2.2. Прямой метод определения технико-экономических показателей

Правильная декомпозиция программной системы и опытные эксперты — залог успешного применения прямого метода определения размеров ПС.

Прямой метод определения технико-экономических показателей программной системы основан на использовании ***метода экспертных оценок.*** Будущую программную систему следует декомпозировать до уровня элементарных компонентов, а для оценки размеров каждого из компонентов использовать либо внешних экспертов, имеющих опыт разработки подобных систем и готовые прототипы, либо использовать в качестве экспертов специалистов разработчика и заказчика.

При декомпозиции целесообразно использовать следующие термины и определения (рис. 3.1):

- ***интегрированная программная система*** — совокупность двух и более ПС, в которых функционирование одной из них зависит от результатов функционирования другой;
- ***программная система*** — совокупность программных комплексов, реализующих множество бизнес-процессов организации;
- ***программный комплекс*** — совокупность программных компонентов, реализующих конкретный бизнес-процесс;
- ***сложный программный компонент*** — совокупность программных кодов, реализующих сложную функцию бизнес-процесса;
- ***программный компонент*** — совокупность программных кодов, реализующих элементарную функцию бизнес-процесса.

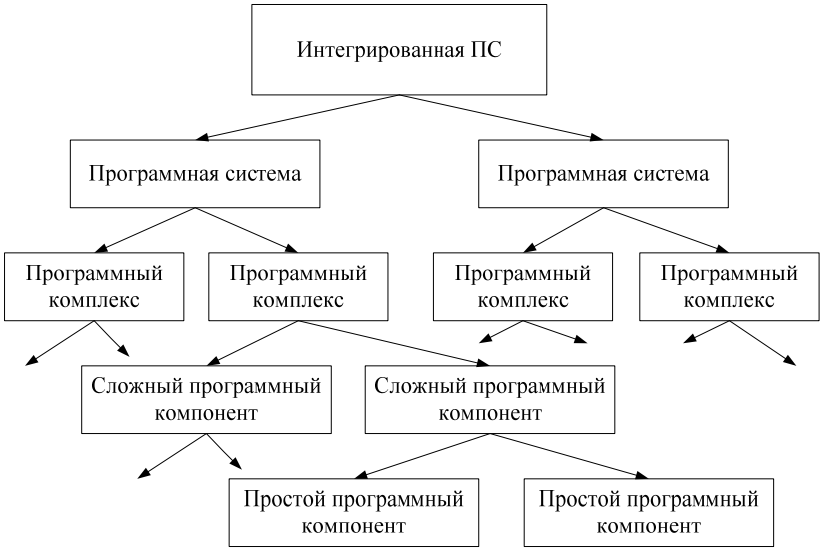

Рис. 3.1. Структура программной системы

Размеры программной системы определяются в виде количества строк исходного кода в терминах Lines of code (LOC) [9].

При оценке количества строк исходного кода следует учитывать следующие положения:

- строка исходного кода содержит только один оператор;
- определение (описание) исходных данных учитывается один раз;
- не учитываются строки, содержащие комментарии и отладочные операторы;
- учитывается каждая инициализация, вызов либо включение макроса в качестве исходного кода.

В качестве базового показателя количества строк исходного кода следует использовать число операторов языка Ассемблер. Варианты преобразования размеров программы, оцененной по этому измерителю, в размеры программы кода, написанного на других языках программирования, представлены в табл. 3.3.

Каждый из экспертов должен дать оптимистическую o, пессимистическую p и реалистическую b оценки (табл. 3.4).

Таблица 3.3

Соответствие среднего числа строк текста программы на языке Ассемблер одной строке других языков программирования

Язык программирования	Ассемблер (LOC)	Показатель LOC на 1 функциональную точку
1. Basic Assembler	1	320
2. Macro Assembler	1,5	213
3. Basic	3	107
4. Pascal	3,5	91
5. C++	6	53
6. Java	6	53
7. Oracle, Sybase	8	40
8. Access	8,5	38
9. Delphi	11	29
10. Oracle Developer/2000	14	23
11. Smalltalk	15	21
12. Cobra	16	20
13. HTML 3.0	22	15
14. SQL (ANSI)	25	13
15. Excel	50	6

Таблица 3.4

Бланк экспертного оценивания размерности программной системы. Язык программирования _____

Состав программной системы	Оценки		
	Оптимистическая	Реалистическая	Пессимистическая
1. Программная система			
1.1. Программный комплекс			
1.1.1. Программный компонент 1			
1.1.2. Программный компонент 2			
……			
……			

Средняя оценка (r) размерности ПС в строках кода по бета-распределению определяется по формуле

$$r_{ij}^{k} = (o_{ij} + 4b_{ij} + p_{ij}) / 6. \qquad (3.4)$$

После оценивания всех компонентов на каждом уровне, начиная с нижнего, определяется интегральная оценка путем суммирования результатов измерения по принципу «снизу-вверх»:

$$R = \sum_{k=1}^{q} \sum_{i=1}^{n} \sum_{j=1}^{m_i} r_{ij}^{k} / q,\ k = \overline{1,q},\ i = \overline{1,n},\ j = \overline{1,m}, \qquad (3.5)$$

где q — количество экспертов; m_i — количество программных компонентов на i-ом уровне; n — число уровней.

Очевидно, что эффективность оценивания может быть существенно повышена при наличии прототипов будущей ПС. В этом случае эксперту предлагается оценить необходимость степени модернизации имеющегося прототипа, увеличив либо сократив исходный размер программного компонента на некоторое количество строк.

Определение трудозатрат, длительности реализации проекта и средней численности разработчиков

Оценка трудозатрат, длительности и средней численности разработчиков при реализации проекта основывается на согласовании между разработчиком и заказчиком производительности труда программиста P.

В [9] приводятся следующие среднестатистические оценки производительности труда программиста:

- при разработке ПС первого класса сложности преимущественно на языке Ассемблер 60–80 строк/чел.-месяц;
- при разработке ПС второго класса сложности (ИПС) на языках высокого уровня 250–260 строк/чел.-месяц.

В табл. 3.5 представлены статистические показатели производительности, рекомендуемые в базовой модели Constructive Cost Model (COCOMO) [9].

Таблица 3.5

Нормативы трудоемкости разработки программ

Класс сложности ПС	Размеры ПС	
	Простые (до 30 тыс. строк)	Сложные (до 500 тыс. строк)
Первый тип (КПС)	до 140 строк/чел.-месяц	до 80 строк/чел.-месяц
Второй тип (ИПС)	до 220 строк/чел.-месяц	до 160 строк/чел.-месяц

Приведенные нормативы отражают не только трудоемкость непосредственного написания текстов программ, но и процессы комплексирования и испытания всего программного комплекса.

С учетом вышеизложенного трудозатраты на разработку системы могут быть определены по формуле

$$T = R / P. \qquad (3.6)$$

Длительность разработки Д может быть задана директивно заказчиком, исходя из реальных потребностей его бизнеса и наличия финансовых ресурсов. В этом случае средняя численность специалистов, которые должны быть привлечены к реализации программной системы, определяется по формуле

$$Z = T / \text{Д}. \qquad (3.7)$$

Прямой метод целесообразно использовать на ранних стадиях проектирования при разработке концепции и технического задания на будущую программную систему. Это позволит разработчику и заказчику определить трудоемкость реализации каждого бизнес-процесса, проранжировать бизнес-процессы в соответствии с пожеланиями заказчика, соизмерить финансовые возможности заказчика и сроки реализации проекта.

3.2.3. Определение технико-экономических показателей с использованием метода функциональных точек

Наличие функциональных моделей бизнес-процессов и опыт будущих пользователей — залог успешного применения метода функциональных точек.

Метод функциональных точек FP (Function point) основывается на том, что размерность программной системы оценивается в терминах количества и сложности бизнес-процессов (функций), реализуемых в данном программном коде [9, 10]. Множество функциональных точек каждого бизнес-процесса включает в себя ***ввод, вывод, опросы, структуры данных, интерфейсы***.

Будущая система с использованием методологии структурного анализа и проектирования описывается в виде многоуровневой графической модели, представленной в виде совокупности взаимосвязанных функциональных диаграмм (пользовательских бизнес-процессов). Каждый из бизнес-процессов включает

в себя входные и выходные данные, преобразования, внешние интерфейсы. Процедура оценивания размеров ПС соотносится с одним из пользовательских бизнес-процессов и состоит из следующей последовательности этапов:

- выделение множества бизнес-процессов;
- подсчет количества функциональных точек бизнес-процесса в разрезе каждой категории;
- определение весовых коэффициентов сложности каждой функции;
- учет факторов и требований среды разработки ПС;
- вычисление интегральных показателей сложности;
- вычисление итогового количества функциональных точек;
- определение размеров программного комплекса бизнес-процесса в показателях LOC согласно табл. 3.3;
- определение размеров программной системы в целом.

При определении количества функций каждого бизнес-процесса следует руководствоваться следующими требованиями:

- учитываются только сложные функции, перечисленные в техническом задании (в требованиях);
- при декомпозиции сложной функции учитываются все логические преобразования с данными.

Расчет количества функциональных точек по каждому бизнес-процессу рекомендуется сводить в таблицу (табл. 3.6).

Таблица 3.6

Таблица определения количества функциональных точек

Наименование функции	Количество функциональных точек, соответствующее категории функции			Количество функциональных точек
	Простая	Средняя	Сложная	
1. Определение количества выводов	$\alpha_{11} \times x_{11}$	$\alpha_{12} \times x_{12}$	$\alpha_{13} \times x_{13}$	X_1
2. Определение количества вводов	$\alpha_{21} \times x_{21}$	$\alpha_{22} \times x_{22}$	$\alpha_{23} \times x_{23}$	X_2

Продолжение табл. 3.6

Наименование функции	Количество функциональных точек, соответствующее категории функции			Количество функциональных точек
	Простая	Средняя	Сложная	
3. Определение количества опросов вывода	$\alpha_{31} \times x_{31}$	$\alpha_{32} \times x_{32}$	$\alpha_{33} \times x_{33}$	X_3
4. Определение количества опросов ввода	$\alpha_{41} \times x_{41}$	$\alpha_{42} \times x_{42}$	$\alpha_{43} \times x_{43}$	X_4
5. Определение количества файлов	$\alpha_{51} \times x_{51}$	$\alpha_{52} \times x_{52}$	$\alpha_{53} \times x_{53}$	X_5
6. Определение количества интерфейсов	$\alpha_{61} \times x_{61}$	$\alpha_{62} \times x_{62}$	$\alpha_{63} \times x_{63}$	X_6
Общее количество функциональных точек				$\sum_{i=1}^{6} X_i$

Примечание: α_{ij} — весовой коэффициент сложности i-й функции j-й категории сложности; x_{ij} — количество элементов данных i-й функции j-й категории сложности.

Определение количества выводов. Под выводами будем понимать следующие единицы информации, получаемые на выходе рассматриваемого бизнес-процесса:

- файлы, продуцируемые в данном бизнес-процессе для передачи другим бизнес-процессам либо за пределы ПС;
- единицы деловой информации, предназначенные для конечных пользователей и оформленные в виде экранных форм либо бумажных документов.

Каждый из выводов в зависимости от количества файлов, используемых при формировании выходов, рекомендуется отнести к одной из категорий сложности: простой, средней, сложной. В табл. 3.7 приводятся весовые коэффициенты сложности выводов.

Таблица 3.7

Весовые коэффициенты сложности выводов

Количество файлов	Значение коэффициент α в зависимости от количества элементов данных		
	от 1 до 5, α_{11}	от 6 до 19, α_{12}	20 и более, α_{13}
1	4	4	5
2–3	4	5	7
4 и более	5	7	7

Определение количества вводов. Под вводами будем понимать следующие единицы информации, поступающие на вход рассматриваемого бизнес-процесса:

- входные файлы, полученные из других бизнес-процессов либо других программных систем;
- уникальная единица деловой информации, вводимая конечным пользователем.

По аналогии с выводом все вводы также рекомендуется разделять на простые, средние и сложные (табл. 3.8).

Таблица 3.8

Весовые коэффициенты сложности ввода

Количество файлов	Значение коэффициент α в зависимости от количества элементов данных		
	от 1 до 5, α_{21}	от 6 до 19, α_{22}	20 и более, α_{23}
1	4	4	5
2–3	4	5	7
4 и более	5	7	7

Определение количества опросов ввода, вывода. Под опросами будет понимать следующие действия, исполняемые программной системой в рассматриваемом бизнес-процессе:

- обращение к внешним процедурам, оформленным в виде специфических команд или запросов, генерируемых извне и выполняемых программной системой;
- выполнение процедур, обеспечивающих непосредственный доступ к базе данных и выполняющих выборку с помощью простых ключей в режиме реального времени, но не выполняющих функции обновления.

Рекомендуется учитывать каждую уникальную единицу опроса в следующих случаях, если формат опроса отличается от формата ввода/вывода либо формат опроса совпадает с форматом ввода/вывода, но требует дополнительной логики обработки.

При определении количества опросов не следует учитывать ***запросы*** к базам данных, использующие несколько ключей и выполняющие определенные операции либо вычисления с последующим оформлением выводов.

Все опросы также рекомендуется разделять на простые, средние и сложные. В табл. 3.9 и 3.10 приведены рекомендации по выбору весовых коэффициентов.

Таблица 3.9

Весовые коэффициенты сложности опросов вывода

Количество файлов	Значение коэффициент α в зависимости от количества элементов данных		
	от 1 до 5, α_{31}	от 6 до 19, α_{32}	20 и более, α_{33}
1	4	4	5
2–3	4	5	7
4 и более	5	7	7

Таблица 3.10

Весовые коэффициенты сложности опросов ввода

Количество файлов	Значение коэффициент α в зависимости от количества элементов данных		
	от 1 до 5, α_{41}	от 6 до 19, α_{42}	20 и более, α_{43}
1	3	3	4
2–3	3	4	6
4 и более	4	6	6

Определение количества файлов. Под файлами будем понимать следующие единицы информации, использующиеся программной системой в рассматриваемом бизнес-процессе:

- внутренние логические файлы программной системы;
- структуры данных, представляющие собой первичную логическую группу пользовательских данных, которые постоянно находятся внутри границ программной системы;
- внешние файлы, доступные пользователям с помощью ввода, вывода, опросов либо интерфейсов.

Весовые коэффициенты оценки сложности файлов в зависимости от количества взаимосвязей между таблицами представлены в табл. 3.11.

Таблица 3.11

Весовые коэффициенты сложности структурных данных (файлов)

Количество логических взаимосвязей	Значение коэффициент α в зависимости от количества элементов данных		
	от 1 до 5, α_{51}	от 6 до 19, α_{52}	20 и более, α_{53}
Одна логическая запись типа формат/взаимосвязь	7	7	7
От 2 до 5 записей	7	10	10
Более 6 записей	10	15	10

Определение количества интерфейсов. Под интерфейсами, используемыми рассматриваемым бизнес-процессом, будем понимать:

- файлы, сгенерированные другими программными системами и использующиеся в данной ПС;
- потоки данных, хранящиеся за пределами программной системы, но используемые при управлении вычислительным процессом в любом направлении пересылки;
- структуры данных, использующиеся в нескольких программных системах.

Весовые коэффициенты оценки сложности интерфейсов представлены в табл. 3.12.

Таблица 3.12

Весовые коэффициенты сложности интерфейсов

Количество логических взаимосвязей	Значение коэффициент α в зависимости от количества элементов данных		
	от 1 до 5, α_{61}	от 6 до 19, α_{62}	20 и более, α_{63}
Одна логическая запись типа формат/взаимосвязь	5	5	7
От 2 до 5 записей	7	7	10
Более 6 записей	7	10	10

Общее количество функциональных точек F определяется по формуле

$$F = \sum_{i=1}^{3} \sum_{j=1}^{6} \alpha_{ij} \cdot x_{ij}. \qquad (3.8)$$

Учет факторов и требований среды разработки. Сложность предметной области и качества создаваемого программного обеспечения зависит от среды разработки приложений и требований конечных пользователей. Влияние этих факторов на размеры программного обеспечения оценивается по ряду показателей (табл. 3.13). Каждый из показателей, в свою очередь, оценивается по пятибалльной шкале. Рекомендуемая шкала измерения показателей приведена в табл. 3.14.

Таблица 3.13

Факторы среды разработки

Факторы среды	Описание фактора
Каналы передачи данных	Передача входных и выходных данных осуществляется по локальной сети, магистральным каналам связи, Internet
Распределенные вычисления	Использование пользовательскими приложениями данных, хранящихся в едином хранилище или получаемых из других систем
Производительность системы	Существенная зависимость для конкретного бизнес-процесса от скорости передачи данных и времени отклика системы
Конфигурирование	Выполнение пользовательских приложений с применением интенсивно используемой, ограниченной либо наполненной конфигурации
Частота транзакций	Высокий сетевой трафик вследствие использования приложений, изменение экранных форм, высокая концентрация выходных форм
Интерактивная обработка	Частота использования пользовательских приложений, участие пользователя при выполнении запросов
Пользовательский интерфейс	Организация взаимодействия конечных пользователей с программной системой с учетом человеческого фактора

Продолжение табл. 3.13

Факторы среды	Описание фактора
Интерактивное обновление базы данных	Степень динамики обновления данных
Сложность обработки запросов	Уровень сложности алгоритмов обработки, количество транзакций, требования к безопасности и надежности
Сложность инсталляции (установки) ПС	Наличие автоинсталляции, качество технической документации
Сложность эксплуатации системы	Наличие процедур запуска, резервирования, копирования, восстановления при ошибках, уровень (сложность) участия пользователей в этих процессах
Степень распределенности системы	Количество и удаленность пользовательских приложений
Гибкость изменения функций	Модульная реализация, наличие настроек, уровень поддержки со стороны пользователей, возможность изменения запросов

Таблица 3.14

Шкала измерения факторов внешней среды

Влияние фактора несущественно	Влияние фактора существенно	Влияние фактора очень существенно
[0 — 1]	[2 — 3]	[4 — 5]

Уровень влияния факторов (W) внешней среды рекомендуется определять по формуле

$$W = 0{,}65 + 0{,}01N, \qquad (3.9)$$

где N — суммарное значение весовых коэффициентов факторов внешней среды.

Уточненное количество функциональных точек $R(F)$ с учетом факторов внешней среды определяется по формуле

$$R(F) = F \cdot W. \qquad (3.10)$$

Размерность программного обеспечения для конкретного языка программирования определяется с учетом нормативов, представленных в табл. 3.3 по формуле

$$R(LOC) = R(F) \cdot LOC\ , \qquad (3.11)$$

где LOC — среднее количество операторов конкретного языка программирования, необходимое для реализации одной функциональной точки (см. табл. 3.3).

Итоговая размерность ПС определяется путем суммирования величины каждого $R(LOC)$ бизнес-процесса.

Определение трудозатрат, длительности и средней численности специалистов на основе базовой конструктивной модели трудозатрат COCOMO

В основу ***оценки трудозатрат*** положена степенная функция вида

$$T = A \cdot R^E(KLOC)/12\ , \qquad (3.12)$$

где T — трудозатраты, чел.-месяцев;

$R(KLOC)$ — размерность ПС, тыс. строк кода.

Первый сомножитель A является доминирующим, он прямо пропорционален размерности программного обеспечения R и отражает линейную зависимость роста трудозатрат от размерности.

Второй сомножитель R^E отражает тот факт, что при увеличении размерности программной системы возрастает относительная трудоемкость разработки каждой строки программного кода за счет увеличения количества взаимосвязей между компонентами.

Значения параметров A и E, полученные путем статистической обработки данных по результатам реализации множества проектов, представлены в табл. 3.15 [10]. Оценки по модели COCOMO получены в результате обработки статистических данных по 160 реальным зарубежным проектам, а оценки по модели «ПРОМЕТЕЙ» являются результатом обобщения статистики по 250 отечественным проектам.

Таблица 3.15

Коэффициенты математической модели оценки трудозатрат в зависимости от типа программных систем

Тип программной системы	СОСОМО		ПРОМЕТЕЙ	
	A	*E*	*A*	*E*
Первый тип (КПС)	3,6	1,2	10	1,21
Второй тип (ИПС)	3	1,12	6,1	1,17
Третий тип (ППП)	2,4	1,05	—	—

Длительность разработки программной системы определяет общие сроки разработки ПС, начиная от разработки технического задания (требований) на систему и завершая этапом проведения комплексных испытаний, и может быть задана директивно заказчиком, исходя из реальных потребностей его бизнеса и наличия финансовых ресурсов.

Средняя численность сотрудников, занятых в проекте, определяется по формуле:

$$Z = T / Д. \qquad (3.13)$$

Расчет длительности и, соответственно, численности специалистов для разработки программного обеспечения может быть произведен также из среднестатистической производительности труда программиста.

Определение трудозатрат, длительности и средней численности специалистов на основе модифицированной модели COCOMO II

В модифицированной модели COCOMO II при определении трудоемкости ***учитываются дополнительно пять групп факторов***, влияющих на технико-экономические показатели проекта:

1) масштабность проекта;

2) требования к показателям качества программного обеспечения;

3) квалификация коллектива разработчиков;

4) характеристики технологической среды разработки;

5) характеристики программно-аппаратной среды разработки.

В табл. 3.16 представлен перечень конкретных показателей по каждой из вышеперечисленных групп и их максимальные значения. Конкретные значения показателей определяются совместно представителями разработчика и заказчика. При значении показателя, ***равном единице***, считается, что соответствующий фактор не влияет на трудоемкость разработки программной системы.

Таблица 3.16

Состав и максимальные значения факторов модифицированной модели COCOMO II

Наименование фактора	Обозначение	Максимальное значение
1. Масштабные факторы		
1.1. Новизна проекта	$N1$	1,33
1.2. Согласованность с требованиями и интерфейсами	$N2$	1,26
1.3. Управление рисками и архитектурой проекта	$N3$	1,39
1.4. Слаженность работы коллектива	$N4$	1,29
1.5. Технологическая зрелость обеспечения разработки	$N5$	1,43
2. Требования к показателям качества ПО		
2.1. Надежность функционирования	$M1$	1,54
2.2. Размер базы данных	$M2$	1,42
2.3. Сложность функций и структуры	$M3$	2,38
2.4. Требование повторного использования компонентов	$M4$	1,31
2.5. Полнота и соответствие документации проекта	$M5$	1,52
3. Характеристики коллектива специалистов		
3.1. Квалификация аналитиков	$M9$	2,00
3.2. Квалификация программистов	$M10$	1,76
3.3. Стабильность коллектива	$M11$	1,51
3.4. Опыт работы по тематике проекта	$M12$	1,51
3.5. Опыт работы в инструментальной среде	$M13$	1,40
3.6. Опыт работы с языками программирования	$M14$	1,43

Продолжение табл. 3.16

Наименование фактора	Обозначение	Максимальное значение
4. Характеристики технологической среды разработки		
4.1. Уровень инструментальной поддержки проекта	$M15$	1,50
4.2. Необходимость распределенной разработки проекта	$M16$	1,53
4.3. Ограничения длительности разработки проекта	$M17$	1,43
5. Характеристики программно-аппаратной среды разработки		
5.1. Ограниченность времени исполнения программ	$M6$	1,63
5.2. Ограниченность доступной оперативной памяти	$M7$	1,46
5.3. Изменчивость виртуальной среды разработки проекта	$M8$	1,49

Оценка трудоемкости разработки программной системы по модифицированной модели производится по выражению

$$T = A \times R^{E}(KLOC) \times \left(\prod_{i=1}^{n} M(i) \right)^{1/n}, \qquad (3.14)$$

где $A = 2{,}94$; $E = B + 0{,}01$, где $B = 0{,}91$;

Данная модель предусматривает возможность прогнозирования длительности разработки на основе регрессионной модели

$$Д = G \cdot T^{H}, \qquad (3.15)$$

где $G = 3{,}67$; $H = K + 0{,}02(E - B)$, где $K = 0{,}28$.

Средняя численность сотрудников определяется по формуле

$$Z = T / Д. \qquad (3.16)$$

Использование модифицированной модели COCOMO II позволяет в среднем на 5–10 % повысить точность определения технико-экономических показателей проекта.

3.2.4. Определение технико-экономических показателей проекта на основе размерности базы данных

Просто рассчитать размерность программной системы — не просто определить масштабы информационной модели.

Размерность программной системы определяется количеством объектов, атрибутов и их взаимосвязями на объектных диаграммах бизнес-процессов [11].

Атрибут — простейший элемент базы данных информационной модели, содержащий одну из характеристик предметной области и вводимый либо непосредственно пользователем, либо заносящийся в базу из справочников и классификаторов.

Объект — элемент базы данных, формируемый из атрибутов и содержащий информацию о реальном процессе, явлении, предмете.

Размерность программного обеспечения определяется по формуле

$$R = 2N \cdot 5K_1 \cdot 10M, \qquad (3.17)$$

где N — количество объектов (таблиц) предметной области, количество связей между таблицами неограниченно и определяется структурой базы данных;

K_1 — суммарное количество взаимосвязей между объектами;

M — суммарное количество атрибутов предметной области, количество связей между атрибутами определяется количеством источников формирования атрибутивной информации.

Нормализованной величиной при создании программной системы является количество формируемых атрибутов, входящих в электронные таблицы посредством установленных связей. При значениях N, K_1 и M, равным единице, величина, выражающая их количество, равна 100. Трудозатраты разработки определяются по формуле 3.18 на основе статистических нормативов трудоемкости, приведенных в табл. 3.17.

$$T = 0,01 \cdot R \cdot \theta, \qquad (3.18)$$

где θ — норматив трудоемкости разработки ПС.

Таблица 3.17

Нормативы трудоемкости разработки программной системы

Категория сложности	Значение норматива θ, чел./месяц
Разработка прикладных программ (пользовательских приложений) с использованием стандартных средств СУБД. Количество прикладных программ не более 3-х. Размерность базы данных до 90 тыс. полей	0,00566
Разработка прикладных программ (пользовательских приложений) с использованием стандартных пакетов прикладных программ. Количество прикладных программ от 3-х до 10-ти. Размерность базы данных от 90 до 200 тыс. полей	0,00808
Разработка прикладных программ (пользовательских приложений) с использованием языков высокого уровня. Количество прикладных программ не ограничено. Размерность базы данных от 200 до 500 тыс. полей	0,01537

Длительность разработки может быть задана директивно заказчиком, исходя из реальных потребностей его бизнеса и наличия финансовых ресурсов, при этом средняя численность специалистов определяется по формуле:

$$Z = T / Д. \qquad (3.19)$$

Данный метод рекомендуется использовать при разработке программных систем на базе стандартных СУБД при следующих условиях:

- большие размерности базы данных, формируемой из различных источников;
- наличие специализированных компонентов, реализующих произвольные информационные запросы пользователей.

3.3. Определение договорной цены на создание программной системы

3.3.1. Определение фонда оплаты труда на разработку и комплексные испытания программной системы

Материальное благосостояние разработчиков зависит от их квалификации и желания заказчика это признать.

В основу определения фонда оплаты труда положены:

- длительность реализации каждого этапа жизненного цикла проекта;
- количественный и качественный состав специалистов, привлекаемых на каждом этапе проекта;
- базовая месячная ставка специалиста-программиста.

В табл. 3.18, 3.19 приведены среднестатистические распределения первых двух величин по основным этапам жизненного цикла создания программных систем [9].

Таблица 3.18

Распределение трудозатрат и длительности по основным этапам жизненного цикла создания программных систем

Этапы жизненного цикла	Трудозатраты α, %	Длительность β, %
1. Анализ предметной области и разработка требований	10	10
2. Проектирование	22	30
3. Программирование	40,5	35
4. Тестирование и комплексные испытания	27,5	25

Используя эти распределения, по выражению 3.20 можно рассчитать среднюю численность сотрудников, занятых на каждом из этапов создания программной системы.

$$Z_i = \alpha_i T / \beta_i Д, \; i = \overline{1,4}. \qquad (3.20)$$

В табл. 3.19 дано относительное распределение численности специалистов на каждом из четырех этапов жизненного цикла создания программной системы.

Таблица 3.19

Распределение специалистов по этапам жизненного цикла ПС

Этапы жизненного цикла	Типы специалистов, %		
	Аналитики	Программисты	Технические специалисты
1. Анализ предметной области и разработка требований	40	20	40
2. Проектирование	35	35	30
3. Программирование	10	65	25
4. Тестирование и комплексные испытания	15	60	25

Численность каждого типа специалистов на каждом из этапов жизненного цикла создания программной системы определяется по следующему выражению:

$$Z_{ij} = P_{ij} \cdot Z_i,\ i = \overline{1,4};\ j = \overline{1,3}, \qquad (3.21)$$

где P_{ij} — относительная доля специалистов *j*-го типа, привлекаемых для реализации проекта на *i*-ом этапе, %.

Фонд заработной платы для реализации *i*-го этапа проекта определяется по формуле

$$S_i = \sum_{i=1}^{4} Z_{ij} \cdot Д_i \cdot S_j,\ j = \overline{1,3}, \qquad (3.22)$$

где $Д_i$ — длительность *i*-го этапа проекта;

S_j — месячный фонд заработной платы *j*-го типа специалиста.

В основу определения S_j может быть положена месячная базовая ставка программиста, размер которой может быть принят как одна из альтернатив: базовая ставка программиста заказчика; базовая ставка программиста разработчика; рыночная базовая ставка программиста в данном регионе.

Соотношение месячной ставки специалиста-программиста к месячной ставке системного аналитика составляет 1:1,3, а к месячной ставке технического специалиста как 1:0,7.

Общий фонд заработной платы на реализацию проекта определяется по формуле

$$S = \sum_{i=1}^{4} S_i. \qquad (3.23)$$

На **этапе опытной эксплуатации программной системы** в соответствии с ГОСТ 34.603-92 производится заполнение нормативной базы, справочников, классификаторов, эксплуатация программного обеспечения в регламентном режиме, доработка программного обеспечения и рабочей документации в случае несоответствия текущей версии системы требованиям технического задания. Срок опытной эксплуатации оговаривается в техническом задании.

Численность сотрудников, привлекаемых к опытной эксплуатации, определяется по формуле

$$Z_{\text{оп}} = t_{\text{оп}} \cdot N, \qquad (3.24)$$

где $t_{\text{оп}}$ — срок опытной эксплуатации;

N — норматив трудоемкости при проведении опытной эксплуатации.

Нормативы трудоемкости на стадии опытной эксплуатации программной системы определяются исходя из следующих среднестатистических нормативов [11] (табл. 3.20).

Таблица 3.20

Нормативы трудоемкости на стадии опытной эксплуатации ПС

Измеритель — 1 сеанс работы

Категория сложности	Значение норматива N, чел./месяц
Количество пользователей — не более 10. Количество сеансов работы с системой в течение года — от 10 до 150	0,00354
Количество пользователей не более 20. Количество сеансов работы с системой в течение года — от 150 до 650	0,00504
Количество пользователей не ограничено. Количество сеансов работы с системой в течение года — от 650 до 6000	0,0095

Опытная эксплуатация проводится группой внедрения разработчика с привлечением в случае необходимости программистов. Относительный норматив заработной платы специалиста составляет 0,85 % от базовой ставки программиста.

Фонд заработной платы на проведение опытной эксплуатации определяется по формуле

$$S_{\text{оп}} = Z_{\text{оп}} \cdot t_{\text{оп}} \cdot S_n \cdot 0{,}85, \qquad (3.25)$$

где S_n — месячная базовая ставка программиста.

3.3.2. Структура договорной цены на программную систему

Чем крупнее компания, тем труднее идут переговоры с заказчиком о договорной цене.

Структура договорной цены представлена в табл. 3.21 и состоит из следующих статей:

1) ***прямые расходы:***

- фонд оплаты труда;
- единый социальный налог;
- амортизация программно-аппаратного комплекса, используемого разработчиком при реализации проекта. Устанавливается, исходя из нормативного срока окупаемости вычислительной техники (5 лет);
- командировочные расходы. Определяются на договорной основе, а для бюджетных учреждений — на основании Приказов Минфина РФ;
- оплата коммунальных услуг, размер которых определяется, исходя из численности специалистов и нормативов площади на 1 рабочее место;
- прочие расходы (расходные материалы, услуги связи и т. д.) определяются на договорной основе (по согласованию с заказчиком), при этом услуги связи (телефоны, Интернет) определяются действующими расценками на данные услуги со стороны региональных отделений ОАО «Ростелеком» и провайдеров сети Интернет;

2) ***накладные расходы*** организации-разработчика (расходы на АУП, охрану, обслуживающий персонал и т. д.), размер

которых подтверждается соответствующими документами разработчика;

3) ***фонд развития производства***, который устанавливается как определенный процент от прямых расходов по согласованию между разработчиком и заказчиком.

4) ***налог на добавленную стоимость***, который определяется Налоговым кодексом РФ и формой организации предприятия-разработчика. Следует учесть при этом, что согласно налоговому кодексу РФ не подлежат налогообложению (освобождаются от НДС) следующие виды работ:

- выполнение научно-исследовательских и опытно-конструкторских работ за счет средств бюджетов, а также средств Российского фонда фундаментальных исследований, Российского фонда технологического развития и образуемых для этих целей, в соответствии с законодательством РФ, внебюджетных фондов министерств, ведомств, ассоциаций*;*
- выполнение НИР и ОКР учреждениями образования и науки на основе хозяйственных договоров;

5) ***увеличение стоимости основных средств***. По данной статье приобретаются все основные средства организации, которые включают средства вычислительной и офисной техники, сетевое оборудование, системное программное обеспечение.

Контрольные вопросы

1. На какой из стадий разработки программной системы (ГОСТ 19) целесообразно использовать следующие методы:

а) прямой метод;

б) метод функциональных точек;

в) метод, основанный на определении размерности и сложности баз данных?

2. Перечислите достоинства и недостатки каждого из методов определения технико-экономических показателей программной системы.

3. В каких случаях и как целесообразно использовать методику на типовые нормы времени на программирование задач для ЭВМ?

Таблица 3.21

Структура договорной цены

Наименование статей расходов	Методика определения
1. Оплата труда непосредственных исполнителей	Определяется по методике, изложенной в разделе 3.2 пособия
2. Начисления на фонд оплаты труда (единый социальный налог)	Определяется Налоговым кодексом РФ и формой организации предприятия-разработчика
3. Увеличение стоимости основных средств	Учету подлежат средства, связанные с осуществлением уставной деятельности хозяйствующего субъекта, т. е. с производством продукции и другими видами деятельности (здания, сооружения, оборудование, вычислительная техника, машины, транспортные средства и др.)
4. Прочие выплаты (командировки в части суточных)	Определяется приказами Минфина РФ – для бюджетных организаций, на договорной основе – для организаций других форм собственности
5. Транспортные услуги (командировки в части оплаты транспортных расходов)	Определяется на договорной основе
6. Прочие услуги (командировки в части проживания, оплата услуг сторонних организаций по НИР, ОКР и технологическим работам, договора подряда с ЕСН)	Определяется на договорной основе

Продолжение табл. 3.21

Наименование статей расходов	Методика определения
7. Прочие расходы (расходные материалы, услуги связи и т.д.)	Определяются на договорной основе (по согласованию с заказчиком), при этом услуги связи (телефоны, Интернет) определяются действующими расценками на данные услуги со стороны региональных отделений ОАО «Ростелеком» и провайдеров сети Интернет
8. Амортизация программно-аппаратного комплекса	Определяется, исходя из нормативного срока окупаемости вычислительной техники (5 лет)
9. Накладные расходы организации-разработчика (расходы на АУП, охрану, обслуживающий персонал и т.д.)	Подтверждаются соответствующими документами организации-разработчика
10. Налог на добавленную стоимость	Определяется Налоговым кодексом РФ и формой организации предприятия-разработчика.

4. ЭКОНОМИКА ПРОГРАММНЫХ СИСТЕМ И ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

4.1. Экономическая эффективность

4.1.1. Влияние информационных технологий на эффективность ведения бизнеса компании

Информация — слуга бизнеса, кто владеет информацией, тот обязан выиграть в конкурентной борьбе.

Разработка и внедрение программных систем и информационных технологий, как правило, связаны с большими затратами. В этой связи функционирование компаний в рыночной среде требует как минимум анализа экономических последствий, а еще лучше — оценки экономической эффективности того или иного шага автоматизации системы управления компанией.

Информационные технологии как высокоэффективный инструмент управления непосредственно не влияют на конечные финансово-экономические результаты компании. Эффективность использования ИТ напрямую зависит от квалификации менеджеров-управленцев, наличия организационно-экономических механизмов, способствующих внедрению IT-технологий, внутренней корпоративной культуры компании и т. д. Кроме того, практически невозможно аналитически вычленить долю IT-технологий в общем экономическом эффекте от совершенствования бизнес-процессов.

Данное обстоятельство создает некоторую сложность в определении экономии финансовых средств от внедрения информационных технологий. Как правило, многие методы основываются на экспертной оценке влияния IT-технологий на различные сферы деятельности компании. Например, можно рассматривать эффективность инвестиций в IT-технологии через семь факторов повышения эффективности организации:

1) использование методов оптимизации при выработке и принятии решений;

2) наличие больших объемов оперативной релевантной информации, позволяющей управленцу принять перспективное, упреждающее решение;

3) своевременный доступ всех заинтересованных пользователей к важной информации, находящейся в одной централизованной БД;

4) усовершенствование процессов принятия решений на базе достоверной и оперативной информации, экономия времени при анализе второстепенных деталей специалистами, принимающими решения;

5) повышение рыночной привлекательности компании в различных аспектах деятельности;

6) расширение информационной компетентности сотрудников — чем большее количество имеет доступ к корпоративным данным, тем «умнее» и мобильнее становится организация в целом;

7) создание единой среды взаимодействия специалистов.

Автоматизация логистических бизнес-процессов позволяет увеличить производительность работы складов, эффективнее использовать производственные площади, сократить страховые запасы, увеличить оборачиваемость финансовых ресурсов.

Следует отметить, что еще не все руководители однозначно оценивают влияние информационных технологий на развитие бизнеса компании. В табл. 4.1 и 4.2 приведены количественные оценки опроса руководителей российских компаний о позитивных и негативных факторах влияния информационных технологии на развитие бизнеса.

Таблица 4.1

Позитивные факторы использования ИТ

Наименование фактора	Количество положительных ответов, %
1. Совершенствование учета и контроля	91,6
2. Снижение издержек	93,0
3. Совершенствование управления территориально распределенной структурой	68,5
4. Увеличение доли рынка	52,0

Продолжение табл. 4.1

Наименование фактора	Количество положительных ответов, %
5. Прозрачность финансовой и бухгалтерской деятельности	36,0
6. Завоевание новых рынков	33,0
7. Преимущества в конкурентной борьбе	7,0
8. Престижность компании	1,8

Таблица 4.2

Негативные факторы, вызывающие отказ от использования ИТ

Наименование фактора	Количество положительных ответов, %
1. Отсутствие финансов	34
2. Неуверенность в успехе	32
3. Неясность реальной выгоды	23
4. Отсутствие объективной потребности во внедрении ИТ	22
5. Негативный опыт партнеров	8,7
6. Нарушение информационной безопасности	6
7. Незнание рынка IT-проектов	5,5
8. Нежелание показывать прозрачность финансовой и бухгалтерской деятельности	3,4

Как видно из табл. 4.1 наиболее значимыми факторами влияния информационных технологий на бизнес компании являются совершенствование управленческого учета и снижение издержек производства. С развитием в последнее время холдинговых структур компаний существенное значение придается совершенствованию территориально распределенных структур управления. Факторам развития рыночных отношений, которые должны являться существенными, отводится второстепенная роль (от 7 до 52 %).

Среди негативных факторов, препятствующих внедрению информационных технологий, следует отметить слабое финансовое состояние компаний и отсутствие уверенности в практической полезности от внедрения.

Приведенные данные говорят о том, что у разработчиков и руководителей IT-служб имеются проблемы взаимопонимания с руководителями компаний, и от того, насколько грамотны и убедительны аргументы в пользу эффективного использования ИТ при подготовке и принятии решений, зависит успех в бизнесе.

4.1.2. Показатели экономической эффективности программных продуктов и информационных технологий

Лучший метод оценки эффективности вашего программного продукта — «эффект отключения»: если программа не работает и никто не бьет тревогу — ваши дела плохи. Но как определить экономическую эффективность? Как оценить то, что оценке не поддается?

Основные положения по определению экономичности программных систем и автоматизированных систем управления и принципы оценки экономической эффективности изложены в ГОСТ 24.702-85, введенном в действие с 01.01.1987 г. Экономическая целесообразность автоматизации бизнес-процессов компании определяется путем сопоставления показателей приращения эффективности Э, получаемой за счет внедрения информационных систем и технологий, и произведенными затратами Z.

Рассматривая программные продукты и информационные технологии как средство реализации автоматизированных систем управления, остановимся на возможности использования вышеуказанного ГОСТа для определения экономической эффективности.

Приведенные затраты на разработку, внедрение и эксплуатацию программных продуктов определяются по формуле

$$Z = C + E_{\text{н}}K - N, \qquad (4.1)$$

где C — текущие издержки, включая затраты на эксплуатацию;

K — все виды единовременных (капитальных) затрат;

N — объем выбываемых (высвобождаемых) основных фондов;

$E_{\text{н}}$ — нормативный коэффициент экономической эффективности капитальных вложений (в отрасли информатизации и связи $E_{\text{н}} = 0{,}33$).

Структура и содержание текущих затрат подробно рассмотрены в п. 4.1.3 при описании показателя совокупной стоимости владения.

Единовременные затраты на разработку и внедрение включают в себя:

- затраты на разработку;
- капитальные затраты на приобретение (изготовление), транспортирование, монтаж и наладку вычислительной техники, средств связи, программных средств, вспомогательного оборудования, оргтехники, производственно-хозяйственного инвентаря;
- затраты на подготовку (переподготовку) кадров.

Экономичность от внедрения программных продуктов и информационных технологий определяется через систему обобщающих и частных показателей.

Основными обобщающими показателями экономической эффективности являются:

- годовой экономический эффект;
- расчетный коэффициент эффективности приведенных затрат на разработку, внедрение и эксплуатацию;
- срок окупаемости проекта.

К частным показателям, характеризующим экономическую эффективность, относятся:

- годовая экономия (годовой прирост прибыли);
- снижение издержек производственно-хозяйственной деятельности на объекте управления в результате разработки и внедрения;
- повышение производительности труда в сфере управления;
- экономия по всем видам ресурсов (материальным, трудовым, финансовым);
- условное высвобождение численности специалистов, работающих в сфере управления;
- повышение качества выпускаемой продукции, снижение потерь от брака.

Годовой экономический эффект от разработки и внедрения информационных технологий определяется как разница между расчетной годовой экономией $Э$ и расчетными приведенными затратами: Z : $F = Э - Z$.

Расчетный коэффициент экономической эффективности капитальных затрат представляет собой отношение годовой экономии (годового прироста прибыли) к приведенным затратам:

$$F = Э / Z.$$

Срок окупаемости проекта представляет собой отношение приведенных затрат к годовой экономии (к годовому приросту прибыли): $T = K / Э$.

При оценке экономической эффективности от внедрения информационных технологий выделяют, как правило, экономию, полученную в сфере производства и экономию в сфере управления. В сфере производства экономия может быть получена за счет изменения производственных и финансово-экономических показателей, а в сфере управления — за счет условного высвобождения численности сотрудников при увеличении количества и сложности решаемых задач. В силу своей природы как та, так и другая составляющие эффекта не могут быть определены при реальном функционировании ИТ. В этом случае основная роль отводится экспертным методам оценки влияния информационных технологий на ведение бизнеса.

Экономический эффект, полученный в сфере производства, может быть получен в денежном выражении через финансово-экономические показатели деятельности компании по формуле

$$Э = \sum_{i=1}^{n} \alpha_i \cdot P_i \ , \qquad (4.2)$$

где P_i — значение *i*-го показателя, предлагаемого экспертами для оценки экономичности;

α_i — доля влияния информационных технологий на *i*-й показатель.

В настоящее время в литературе отсутствуют какие-либо количественные характеристики оценки влияния информационных технологий на повышение эффективности российского бизнеса. В табл. 4.3 приведены статические данные по оценке западных экспертов, а в табл. 4.4 — статистика советского периода [12].

Таблица 4.3

Среднестатистические мировые показатели эффекта от внедрения информационных технологий

Показатель	Значение показателя, %
Снижение количества задержек при поставках продукции заказчикам	90–97
Уменьшение сверхнормативных остатков на складах материалов	30–45
Повышение оборачиваемости запасов	20–30
Сокращение непроизводственных запасов	17–25
Повышение оборачиваемости средств в области реализации готовой продукции	12–21
Повышение производительности работников и оборудования	10–17
Снижение затрат на закупку материалов и комплектующих	4–6

Таблица 4.4

Показатели экономического эффекта от внедрения информационных технологий

Показатель	Значение показателя, %
Увеличение объема выпуска	34
Снижение себестоимости	60
Снижение внутрисменных потерь рабочего времени	40–45
Сокращение расходов на сырье и материалы	1–3
Сокращение потерь от брака	6–16
Снижение непроизводственных расходов	10–15
Снижение затрат на 1рубль продукции	0,3
Рост производительности труда	2,8

Естественно, при использовании этой статистики, с учетом специфических условий конкретного предприятия, экспертами могут быть внесены корректировки.

Экономия финансовых средств в сфере управления за счет условного высвобождения численности сотрудников определяется, исходя из предположения, что увеличение количества и сложности решаемых задач и должно привести к адекватному росту численности аппарата управления компании. Внедрение

же информационных технологий и «перенос» «ручного труда» специалиста при анализе обработки информации на компьютер существенно сокращает этот рост.

В этом случае математические выражения для определения экономии выглядят следующим образом:

$$Э = \sum_{j=1}^{n} \left(Z_1 \cdot S_1 \cdot T_j - Z_2 \cdot S_2 \cdot T_j \right) - Z, \qquad (4.3)$$

где T_j — количество арифметических и логических операций *j*-й задачи;

Z_1, Z_2 — себестоимость (затраты) в единицу времени на выполнение одной операции специалистом и компьютером;

S_1, S_2 — время обработки одной операции специалистом и компьютером;

Z — дополнительные затраты на увеличение численности сотрудников.

4.1.3. Показатели эффективности вложений в информационные технологии как инвестиционные проекты

Деньги обладают свойством «изменяться во времени».

Инвестиционные проекты существуют обычно в виде некоторого множества альтернативных вариантов, реализация которых сопряжена с определенными финансовыми затратами. В этой связи перед руководством компании возникают следующие вопросы:

- какой объем из общего финансового плана инвестиций вкладывать в развитие инфраструктуры информатизации;
- какой из альтернативных вариантов создания (развития) технического и программного обеспечения выбрать.

В любом случае затраты должны быть экономически оправданы. Аналогично тому, как текущие затраты в общем случае должны покрываться текущими доходами (в противном случае эти расходы могут считаться экономически нецелесообразными), затраты капитального характера должны покрываться будущими

доходами, а точнее будущей прибылью от коммерческого применения капиталовложений. Следовательно, с экономической точки зрения рассмотрение проекта по созданию и развитию информационных технологий в качестве инвестиционного означает необходимость экономического обоснования требуемых затрат и оценки эффективности предполагаемых инвестиций.

Своевременный анализ затрат и ожидаемых результатов позволяет менеджерам:

- сравнивать альтернативные варианты реализации ПС;
- получать соотношения преимуществ одной программной системы над другой при выборе ПС для внедрения;
- определять экономические конкурентные преимущества разработанной системы над аналогичными продуктами конкурентов.

Существует ряд различных моделей и методов, используемых для сравнения анализа затрат и ожидаемых результатов. На практике преобладают дисконтные методы и соответствующий им набор показателей [13–15]:

- чистая дисконтированная стоимость (net present value, *NPV*);
- внутренняя ставка доходности (internal rate of return, *IRR*);
- срок окупаемости проекта;
- коэффициент доходности инвестиций в активы (return on assets, *ROI*);
- совокупная стоимость владения *TCO*.

Прежде чем перейти к описанию расчета каждого из показателей отметим, что величину затрат и размеры инвестиций можно определить на текущий момент времени, а ожидаемый результат в виде прибыли появится в будущих периодах.

Поскольку деньги, полученные в будущем, будут «стоить» меньше, чем если бы они были получены сегодня, будущая прибыль должна быть скорректирована со стоимостью денег в каждом из рассматриваемых интервалов планирования.

Инструментом приведения является норма дисконта E, называемая также требуемым уровнем (нормой) доходности или ставкой сравнения, или барьерной ставкой, и олицетворяющая собой приемлемый для инвестора процент возврата на инвестируемый капитал за определенный период начисления.

Технически приведение выполняется путем умножения объема платежей и поступлений в *t*-ом периоде планирования на коэффициент (процентную ставку) дисконтирования α_t, определяемый для постоянной нормы дисконта E и t_0 по формуле

$$\alpha_t = 0; \alpha_t = \frac{1}{(1+E)^t}, \tag{4.4}$$

где $0 \le \alpha_t \le 1$, $t = 1, 2, ..., n$.

Чистая дисконтированная (приведенная) стоимость NPV относится к показателям эффективности капиталовложений и характеризует чистый абсолютный результат или отдачу от реализации инвестиционного проекта:

$$NPV = \sum_{t=0}^{T} (R_t - З_t) \cdot \alpha_t, \tag{4.5}$$

где R_t — общий (валовой) результат эксплуатации инвестиций, достигаемый на *t*-ом шаге расчетного периода;

$З_t$ — общие затраты на *t*-ом шаге расчетного периода;

T — расчетный период;

$(R_t - З_t)$ — чистый экономический эффект $Э_t$ от использования инвестиций на *t*-ом шаге.

При выделении из общих затрат размера капитальных вложений (инвестиционных затрат) K_t и текущих затрат N_t формула для расчета NPV выглядит следующим образом:

$$NPV = \sum_{t=0}^{T} (Д_t - K_t) \cdot \alpha_t, \tag{4.6}$$

где $Д_t = R_t - N_t$.

Если значение $NPV > 0$, то данный проект можно считать прибыльным, в противном случае он будет убыточным.

Внутренняя ставка (норма) доходности (рентабельности) IRR показывает, какой процент прибыли необходимо заложить в расчеты, чтобы чистая дисконтированная стоимость проекта всех будущих периодов была равна нулю. Другими словами, необходимо определить значение дисконтной ставки, которая

уравнивает приведенную стоимость будущих доходов и приведенную стоимость затрат.

В случае если проект инвестируется за счет банковского кредита, но IRR определяет верхнюю границу допустимого уровня банковской процентной ставки, при котором ожидаемая прибыль будет равняться нулю:

$$\sum_{t=0}^{T} \frac{(Д_t - K_t)}{(1+E)^t} = 0. \qquad (4.7)$$

Численное значение E можно определить одним из приближенных методов решения уравнений.

Срок окупаемости проекта PP определяется количеством периодов, необходимых для полного возврата первоначальных инвестиций. Как правило, компания, планируя инвестиции в проекты, устанавливает и минимальный срок их окупаемости, в течение которого сумма дисконтированных чистых поступлений покрывает сумму приведенных инвестиционных затрат, и в дальнейшем чистый интегральный эффект остается неотрицательным.

$$PP = t^* - \frac{NPV(t^*)}{NPV(t^*) - NPV(t^* - 1)}, \qquad (4.8)$$

где $t^* = \min\{t\}$ — номер периода, с которого чистый интегральный эффект остается неотрицательным (для IT-проектов рекомендуемый срок окупаемости не более 3 лет).

Коэффициент доходности инвестиции ROI позволяет оценить относительную эффективность вложений в информационную структуру фирмы и определяется по формуле

$$ROI = Э_t / З_t, \qquad (4.9)$$

где $Э_t$ — экономический эффект, полученный от внедрения проектов;

$З_t$ — инвестиции (затраты) в проекты.

Каждый из вышеперечисленных параметров не лишен недостатков, так чистый приведенный доход NPV описывает в стоимостном выражении будущую прибыль, однако не позволяет напрямую сравнивать проекты с разным уровнем финансирования. Показатель IRR устраняет указанный недостаток, однако

он не имеет понятной экономической интерпретации и не учитывает объемы первоначальных инвестиций. Срок окупаемости проекта не учитывает изменение денежных потоков во времени.

Основным достоинством *ROI* является простота его определения. Как финансовый показатель он понятен менеджеру и позволяет приводить к одному измерению и сравнивать разномасштабные проекты, легко сравнивается с общей стоимостью активов компании.

Однако расчеты показателей основываются на определении будущей прибыли от внедрения проекта, величина которой не поддается точному определению. Несмотря на математическую точность приведенных формул, расчеты носят приблизительный характер и требуют определения некоторых погрешностей (допусков).

Вместе с тем отмеченные недостатки не влияют на своевременную оценку инвестиций в инфраструктуру информатизации, что позволяет руководителям быть не просто «просителем» средств, а выступать инициатором эффективного инвестиционного проекта, конкурирующего на равных с иными инвестиционными предложениями по развитию бизнеса.

В целом инвестиции в ИТ складываются из капитальных затрат на создание и развитие инфраструктуры информатизации**,** а также ежегодных вложений в оборотный капитал, необходимый для эксплуатации этой инфраструктуры. На практике существует несколько приблизительных оценок объема инвестиций, гарантирующих успешный бизнес компаний. Наиболее распространенной характеристикой оценки объемов в IT-проекты является показатель ***совокупной стоимости владения*** *TCO*. Совокупная стоимость владения информационными технологиями — это качественная ключевая характеристика, отображающая экономические аспекты состояния инфраструктуры информатизации в компании и показывающая эффективность работы компании. Показатель *TCO* рассчитывается по формуле

$$TCO = \text{Пр} + \text{Кр1} + \text{Кр2}, \qquad (4.10)$$

где Пр — прямые расходы;

Кр1 — косвенные расходы первой группы;

Кр2 — косвенные расходы второй группы.

При этом прямые расходы определяются по формуле

$$\text{Пр} = \sum_{i=1}^{8} \text{П}_i, \qquad (4.11)$$

где П_1 — капитальные затраты текущего года;

П_2 — расходы на содержание IT-служб;

П_3 — расходы на технологическую поддержку программно-технического и информационного обеспечения;

П_4 — расходы на разработку прикладного ПО внутренними силами;

П_5 — расходы на аутсорсинг;

П_6 — командировочные расходы;

П_7 — расходы на услуги связи;

П_8 — другие группы расходов.

Каждую из составляющих прямых расходов следует детализировать, например, при определении капитальных затрат на оборудование следует учитывать:

- расходы на приобретение нового оборудования и его замену;
- средства, вырученные от продажи или передачи оборудования;
- амортизацию оборудования;
- затраты на сетевое оборудование и соединения (кабели; концентраторы; карты, которые, как правило, не производятся; амортизационные отчисления);
- расходы на приобретение периферийных устройств;
- расходы на приобретение дополнительной оперативной памяти с учетом расходов на амортизацию оборудования;
- расходы на дополнительные дисковые устройства (учитывается амортизация оборудования);
- расходы на замену оборудования;
- прочие расходы по оборудованию.

Прямые расходы присущи подразделениям, занятым в разработке, внедрении и сопровождении ИТ.

Аналогичным образом следует детализировать и остальные составляющие прямых расходов.

Косвенные расходы возникают в подразделениях, использующих информационные технологии в практической деятельности. К *первой группе косвенных расходов* относятся потери компании в связи с низким качеством IT-проекта:

- дополнительные затраты сотрудников других подразделений на рутинную обработку информации;
- вынужденные простои пользователей в связи с выходом из строя аппаратно-программного обеспечения и т. д.

Природа *второй группы косвенных расходов* кроется в некачественной работе специалистов IT-подразделений, когда пользователи за счет своего рабочего времени занимаются восстановлением работоспособности программно-аппаратных средств.

Косвенные расходы находятся за рамками бюджетов на содержание IT-служб, однако они могут играть существенную роль в оценке проекта. При этом первая их группа («неработоспособность системы») может быть рассмотрена с использованием метода определения производственных потерь. Вторая группа («непроизводительные усилия конечного пользователя»), связанная с информационными технологиями, определяется с помощью статических исследований.

Основная идея использования показателя совокупной стоимости владения заключается в выявлении избыточных статей расходов, оценке возможностей возврата вложенных в информационные технологии средств, в сравнении *TCO* своего предприятия с аналогичным показателем других родственных компаний.

Для западных компаний объемы затрат на развитие и обеспечение функционирования информационных технологий колеблются в пределах 0,9–3,4 % от общего оборота компаний, для российских компании эти проценты несколько ниже: 0,6–1,5 %.

Таким образом, сравнение собственных затрат со среднестатистическими, с учетом амбиций компании и жесткости конкуренции, может помочь в обосновании бюджета на содержание и развитие инфраструктуры информатизации компании.

4.2. Доходы, затраты и риски при создании программного обеспечения

Формула успеха бизнеса проста как мир: увеличивайте доходы, сокращайте расходы, рискуйте в меру.

4.2.1. Доход разработчика

Высокий и стабильный доход — это высокое качество, оптимальные затраты, умеренные цены и безразмерный объем рынка.

Основной целью разработчика (поставщика) ПО является получение прибыли, представляющей собой разницу между доходами и расходами. **Величина доходов** складывается из объемов продаж и поставок новых версий взамен устаревших. **Расходы разработчика** определяются затратами на разработку первой версии системы, адаптации ее к другим пользователям, модернизации системы в связи с изменениями требований заказчика. При заказном программировании заказчики платят только за сложные проекты, в которых разработчик берет на себя их авторское сопровождение. В этом случае доходы разработчика складываются из цены заказа и доходов от сопровождения, а расходы — из себестоимости первичной разработки, сопровождения, создания новых версий, отвечающих изменившимся потребностям заказчика. Доходы тем выше, чем больше длительность авторского сопровождения. И в том, и в другом случаях расходы тем ниже, чем ниже себестоимость первичной разработки и сопровождения. В структуре затрат на производство ПП стоимость сопровождения системы превышает стоимость разработки в 3–4 раза. Кроме того, величины доходов и расходов во многом зависят от рисков, возникающих у разработчика при проектировании и продвижении на рынок программного продукта. На уровень дохода разработчика влияет ряд основных факторов (рис. 4.1) [16].

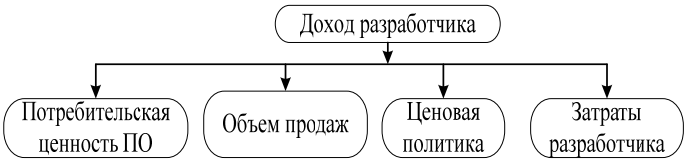

Рис. 4.1. Факторы, определяющие доход разработчика

При равной маркетинговой активности продавцов заказчики, как правило, отдают предпочтение программным продуктам, позволяющим получить ***максимальную выгоду*** при их использовании, определяемую как разность между *потребительской ценностью и общей стоимостью продукта*, включающей единовременные и текущие расходы. В этом случае, очевидно, стратегия разработчика должна быть направлена на увеличение потребительской ценности и снижение общей стоимости.

Потребительская ценность продукта зависит от ряда факторов (рис. 4.2).

Рис. 4.2. Факторы, определяющие потребительскую ценность

Качество программного обеспечения должно максимально соответствовать требованиям стандарта ГОСТ Р ИСО/МЭК9126-93, который рекомендует оценивать программное обеспечение по следующим критериям: функциональная пригодность, надежность, практичность, эффективность, сопровождаемость, переносимость. Технология и метрики измерения каждого из критериев достаточно подробно описаны в соответствующих ГОСТах. Особенно большое значение имеют для пользователя функциональные возможности программного продукта.

Рекламируя свою программную продукцию, фирме следует максимально использовать эти критерии, претендуя, прежде всего,

на *уникальность* созданной системы. Вместе с тем при выборе новых направлений разработки программных систем следует избегать создания продуктов, напрямую альтернативных решениям других производителей, поскольку это «играет на руку» тому поставщику, который контролирует большую долю рынка. Поставщики с упрочившимся положением на рынке имеют более высокий доход при сопоставимых затратах. Очевидно, что серьезное отличие от конкурентов по максимально большему числу критериев имеет решающее значение. С другой стороны, компания, первой на рынке предложившая новую категорию продуктов или значительно улучшившая характеристики ПО из уже имеющейся категории, получает важное преимущество. Таким образом, при проектировании ПО в первую очередь следует уделять внимание полезным и выгодным для заказчиков новаторским решениям. Тезис «мы делаем лучше других» должен быть заменен на тезис «мы делаем отличное от других».

Дополнительным фактором, увеличивающим практическую полезность продукта, является ***возможность использования заказчиком приложений других производителей***. В этом случае ПО должно соответствовать некоторым стандартам на взаимодействие различных приложений. Открытый интерфейс является, как правило, двусторонним, он создает взаимозависимость и поддерживает конкурентоспособность обеих сторон. Поддержка стандартов делает успех на рынке более предсказуемым, в частности, потому что гарантирует интероперабельность с дополняющими решениями и популярность у пользователей.

При распространении любого программного обеспечения круг потенциальных заказчиков определяется выбором инфраструктурной ***платформы.*** Возможность переноса и поддержка программного обеспечения для многих платформ помогает увеличить доходы, но при этом возрастают расходы на разработку, обслуживание, тестирование и поддержку. Переносимость за счет выбора разных языков позволяет несколько снизить эти расходы, однако делает инфраструктуру более сложной для внедрения. Устранение зависимости возможно при минимальных затратах на дополнительное ПО, что увеличивает круг потенциальных заказчиков, но подготовка корректных интерфейсов может привести к росту затрат на разработку и развитие продукта.

Определенное влияние на практическую полезность ПС оказывают «***сетевые эффекты***», позволяющие повысить комплексность и интеграцию различных программных систем, увеличить комфортность использования приложений различными категориями пользователей. Такой процесс взаимовлияний различных проектных решений может привести к интегральному росту размеров рынка.

На общий доход разработчика влияет тот факт, что при создании программного обеспечения предполагаются высокие расходы на создание первой копии и низкие — на тиражирование и распространение. Тем самым обеспечивается солидная экономия **за счет объемов продаж** (расходы в расчете на экземпляр снижаются с ростом количества выпущенных экземпляров), хотя некоторые текущие расходы, скажем, на распространение и поддержку заказчиков, с ростом объема продаж увеличиваются.

Следующим фактором увеличения дохода является использование методик согласования ценовой политики между разработчиком и заказчиком.

В настоящее время используются следующие альтернативные подходы:

1) согласование договорной цены на основе прямых переговоров с заказчиком;

2) определение стоимости лицензии за право использования программного обеспечения в течение определенного периода времени;

3) согласование стоимости абонентской платы за предоставление услуг по поддержке функционирования программного обеспечения.

При определении **договорной цены** следует учитывать, что единовременные затраты и текущие расходы на сопровождение существенно различаются в рыночных условиях создания программного продукта (рис. 4.3). В случае если программный продукт универсален и ориентирован на массовое тиражирование в различных областях применения, можно говорить об относительно невысокой цене продаж, а требуемый уровень дохода можно получить за счет больших объемов (в этом случае расходы на создание первой версии снижаются с ростом количества реализованных экземпляров).

Если же разработка программного обеспечения проводится для конкретного потребителя-заказчика с жестко заданными предметной областью и средой применения, затраты на разработку первой версии предполагаются достаточно большими. В этом случае очевидно стремление заказчика снизить договорную цену проекта, от разработчика же требуется корректное обоснование ее величины.

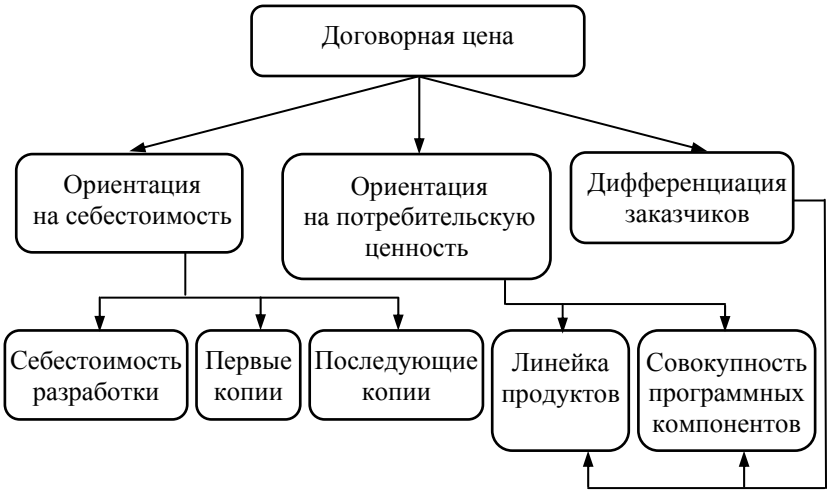

Рис. 4.3. Варианты определения договорной цены

Проблема определения договорной цены заключается в том, что процесс создания ПО, по сути, является процессом преобразования спецификаций, требований и пожеланий заказчика в код и документацию программного продукта. Сложность управления таким процессом вызвана сложностью получения количественных оценок входной и выходной информации, а также операций по ее преобразованию. Он отчасти является творческим и тяжело поддается формализации и моделированию. В результате точное планирование затруднено и только 28 % проектов являются успешными, 23 % — оканчиваются неудачно, 49 % проектов реализуются не в полном объеме. Основные причины неудач связаны с изменением бюджета проекта — 45 %, нарушением календарных планов — 63 %, несоответствием с согласованными

ранее функциональными возможностями. Таким образом, можно утверждать, что наличие эффективной методики оценивания проектов позволит устранить значительную часть причин срыва проектов и будет способствовать повышению конкурентоспособности организации-разработчика на рынке.

В основу определения договорной цены программного продукта должны быть положены следующие показатели: размеры программной системы, трудозатраты на разработку и внедрение, длительность разработки, численность и квалификация специалистов разработчика, нормативы оплаты труда при создании программного кода, структуру накладных расходов разработчика.

Следует отметить, что возможность получения объективных количественных оценок характеристик проектов полезна не только разработчикам, но и заказчикам. Обе заинтересованные стороны могут видеть, какой вклад в стоимость проекта вносит та или иная функциональность, и оценить степень ее важности. По обоюдному согласию из проекта может быть оперативно исключена не оправдывающая своей цены функциональность или ее реализация перенесена на следующую версию или даже в другой проект.

Немаловажным фактором при определении договорной цены является факт наличия у организации-разработчика международного сертификата менеджмента качества по модели зрелости процессов разработки ПО. Модель зрелости процессов разработки программного обеспечения, или CMM (Capability Maturity Model), разработана в Институте программной инженерии (Software Engineering Institute, SEI) при университете Карнеги-Меллона в Питтсбурге.

CMM определяет пять уровней зрелости организации, разрабатывающей ПО:

1) **начальный** (Initial): процесс разработки носит хаотический характер, определены лишь немногие из процессов, успех проекта зависит от личных качеств членов команды, предсказуемость крайне мала;

2) **повторяемый** (Repeatable): установлены основные процессы управления проектами (отслеживаются затраты, график работ, функциональность; упорядочиваются некоторые процессы, позволяющие повторить предыдущие достижения);

3) **определенный** (Defined): процессы разработки ПО и управления проектами документированы и стандартизированы;

4) **управляемый** (Managed): собираются и оцениваются подробные количественные показатели процесса и качества ПО, анализируется динамика этих данных;

5) **оптимизирующий** (Optimizing): процессы постоянно совершенствуются на основе количественных данных по процессам и внедрении новых идей и технологий.

Заметим, что CMM не содержит никаких численных критериев оценки и рекомендаций и не указывает, как конкретно оценить продукт, а лишь рекомендует, что надо сделать для достижения требуемого качества ПО. CMM содержит также способы контроля за правильностью выполнения ключевых действий и методы их корректировки. Для сертификации по CMM необходимо выполнить несколько крупных заказов, чтобы показать на их примере полный цикл разработки. Ясно, что применение CMM наиболее эффективно в компаниях, где многие процессы управления жестко формализованы.

Как отмечалось ранее, на доход разработчика, несомненно, влияет и **объем продаж**. Однако следует учитывать, что из-за значительных расходов на первую копию программного продукта затраты в расчете на один экземпляр зависят от объема продаж, поэтому они слабо связаны с рыночной ценой. Таким образом, при рыночном варианте поставок ПО разработчик должен пытаться определять базовые цены, опираясь не на собственные затраты, а на выгоду для заказчика, и учитывать при этом не только расходы заказчика, но и потребительскую ценность данного продукта.

Если при расчете цен опираться на выгоду заказчика, а не на затраты разработчика, то следует учитывать, что заказчики серьезно различаются по уровню платежеспособности. Чтобы максимально увеличить доход, необходимо ***дифференцированно*** устанавливать цены для различных заказчиков, определяя стоимость программного продукта не только с учетом первоначальных издержек, но и особенностей заказчика. В этой ситуации возможны несколько подходов: разделение заказчиков на группы и установление для каждой из них своих расценок и определение цены непосредственно при переговорах с ними.

Ценовая политика разработчика заключается в предоставлении заказчику линейки различных вариантов реализации программного обеспечения и выборе одного из них в зависимости от реальных потребностей и платежеспособности. Чтобы такой подход стал возможным, архитектура программной системы должна позволять формировать различные комбинации модулей в зависимости от предполагаемых требований к производительности и функциональности при условии, что пользователи не могут самостоятельно менять конфигурацию. При этом разработчики должны установить, оправдана ли подобная гибкость с точки зрения затрат на поддержку и обслуживание, которые будут расти с увеличением числа вариантов и даже числа комбинаций конфигурационных возможностей.

Одной из стратегий ценовой политики, ориентированной на соотношение потребительской ценности и цены, является комплексирование и продажа различных программных продуктов в одном пакете. Совокупная потребительская ценность такого пакета зачастую меньше ценности его компонентов, поскольку разным пользователям нужны разные компоненты, и, соответственно, компоненты пакета для них имеют разную ценность. Поэтому в пакете компонент может обойтись заказчику дороже, чем при его отдельной покупке. Если компоненты пакета рассчитаны на композицию (целое с точки зрения функциональности больше, чем сумма его частей), то ценность пакета становиться выше.

Бизнес-модуль, предусматривающий предоставление заказчику прикладного программного обеспечения как услуги, получил название IT-аутсорсинга (рис. 4.4).

Этот вариант предоставления ПО устраняет зависимость заказчика от выбранной платформы, сокращает затраты на разработку за счет уменьшения числа либо вообще устранения гетерогенных платформ, которые необходимо поддерживать. Кроме того, эта модель сокращает расходы заказчиков, поскольку предполагает, что поддержкой пользователей будет заниматься не поставщик ПО, а дистрибьютор. Однако данная технология требует решения ряда организационно-экономических проблем: определение соотношения между стоимостью разработки и услуг организации-посредника; разграничение прав и обязанностей между разработчиком и дистрибьютором.

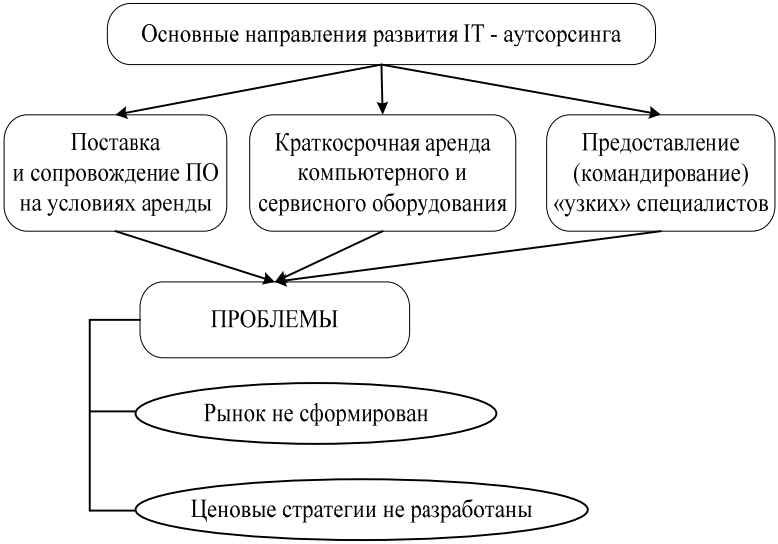

Рис. 4.4. Основные направления развития IT-аутсорсинга

Экономику фирм, специализирующихся на **продажах лицензий на программные продукты**, определяют следующие составляющие:

1) доходы от продажи лицензий;

2) доходы от комиссионного обслуживания;

3) доходы за предоставленные услуги.

Естественно, главным индикатором и движущей силой здесь является первый вид дохода: при отсутствии продаж новых лицензий доход от комиссионных за обслуживание и услуги также снижается. Следующий по значимости движущий фактор — это имеющаяся клиентская база: в среднем более 60 % дохода многим успешным компаниям, продающим прикладное ПО, приносят старые клиенты, которые уже что-то приобрели у них ранее.

В результате понять, как идут дела у поставщика прикладного ПО, довольно просто: надо всего лишь взглянуть на составляющие его дохода. Если большую его часть компания получает от комиссионных за обслуживание, значит она либо не может продать своим клиентам новые лицензии, либо у нее отсутствует

инновационная жилка. Если же основная часть доходов идет от предоставляемых услуг, значит, компания не может продать достаточного количества лицензий. Наконец, предприятия, которые не могут обеспечить значительный доход от своей старой клиентской базы, скорее всего, либо слишком завышают цены, либо просто уходят с рынка.

4.2.2. Затраты на разработку и эксплуатацию

> *Оптимизируйте не только свои расходы, но и будущие расходы заказчика, убедите его в том, что выбор только вашего продукта позволит ему быть рентабельным, минимально зависеть от изменения структуры собственного бизнеса и развития информационных технологий.*

Уровень затрат разработчика определяется расходами на разработку и тестирование первой копии, распространение, обслуживание, модернизацию и поддержку пользователей в течение всего срока существования программного обеспечения. Обслуживание и поддержка заказчика являются долговременной финансовой нагрузкой, которую разработчик принимает на себя в момент поставки первой версии продукта, и это следует учитывать при оценке затрат и рисков. Отказ от поддержки предполагает периодическое создание нового продукта и предоставление заказчикам альтернативных возможностей, что не всегда для них приемлемо. В этом случае речь должна идти о модернизации имеющихся версий ПО. Модернизация должна поддерживать приемлемую обратную совместимость (например, допускать применение как старых, так и новых форматов файлов), причем так, чтобы возможность перейти на конкурирующие продукты не стала для заказчиков более привлекательной, чем обновление существующего решения. Таким образом, модернизация порождает наиболее сложные дилеммы при проектировании, когда приходится выбирать оптимальное соотношение затрат и доходов.

Существенное сокращение текущих затрат на разработку и, соответственно, увеличение дохода связано ***с повторным применением рассчитанных на множественное использование программных компонентов***, которые разработчик может конфигурировать и объединять в различных комбинациях без каких-

либо внутренних модернизаций. Вместе с тем следует заметить, что повторное использование программного кода имеет и ряд сложностей, поскольку создание кода, который можно легко использовать повторно, требует значительных дополнительных усилий, что противоречит целям и задачам руководителей проекта, стремящихся закончить его в срок и в рамках выделенного бюджета. Изменения в коде, которые приходится делать в следующем проекте, также ставят под угрозу возможную экономию на поддержке и модернизации, усложняют процесс ликвидации обнаруженных позднее ошибок. В этом случае следует максимально использовать модульную структуру организации ПС, допускающую раздельную модификацию каждого модуля и также создание программного средства на языках высокого уровня с помощью специализированного инструментария и автоматической генерации (полной или частичной) программного средства по этому высокоуровневому описанию.

Немаловажное значение при взаимоотношениях разработчика и заказчика приобретают ***вопросы снижения совокупных затрат заказчика при внедрении и эксплуатации ПО***.

К совокупным затратам обычно относят (рис. 4.5):

- затраты на разработку;
- прямые текущие затраты на эксплуатацию (сопровождение);
- затраты на реорганизацию производства, связанные с модернизацией бизнес-процессов, например, при внедрении ERP-систем (управление ресурсами предприятия);
- затраты на обучение персонала;
- затраты на модернизацию ПО, в том числе по переходу на другие программные продукты.

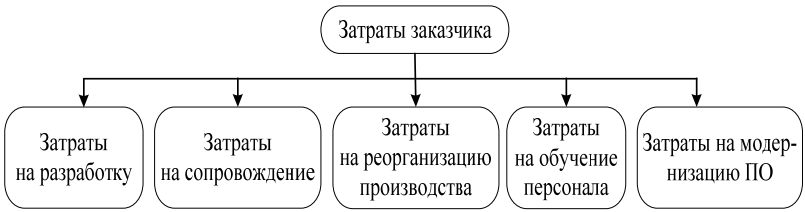

Рис. 4.5. Факторы, определяющие затраты заказчика

Затраты на модернизацию имеют для заказчиков принципиальное значение, поскольку влекут нежелательные последствия: вследствие перехода на другие программные продукты заказчик вынужден нести значительные расходы. В этом случае, при прочих равных условиях, архитектура ПО должна создаваться с таким расчетом, чтобы расходы заказчиков, желающих перейти на продукты конкурентов, были минимальны. Примером такого подхода может служить предоставление возможностей преобразования форматов данных, используемых в продуктах конкурентов, в формат данных конкретного производителя (не наоборот).

Заказчики хорошо представляют себе последствия «привязки» к производителю. Их беспокоит не только отсутствие возможности использовать конкурирующие продукты, но и перспектива того, что производитель либо прекратит поддержку и модернизацию своего продукта, либо вообще выйдет из бизнеса. Поэтому заказчики все чаще отказываются от крупных монолитных приложений, которые, с одной стороны, минимизируют затраты на интеграцию и обеспечиваются поддержкой поставщика, но, с другой стороны, ограничивают при этом имеющуюся функциональность и увеличивают потенциальные расходы в случае перехода на конкурирующие продукты. Альтернативным подходом является применение модульных решений, которые дают возможность объединять и использовать совместно продукты разных производителей или модули одного и того же поставщика (процесс, обратный комплектации).

Затраты на эксплуатацию у заказчика также косвенно влияют на доход разработчика. Поэтому поиск разумного решения по ценовой политике заставляет искать компромисс между затратами поставщика и заказчика. Например, увеличение расходов на поддержку и обслуживание может привести к снижению расходов заказчика на внутреннее администрирование и поддержку пользователей.

Затраты на сопровождение и модернизацию информационной системы могут значительно увеличиться, если заказчик и разработчики допустили при обследовании существенные ошибки, вследствие которых представленная версия не будет соответствовать требованиям существующих бизнес-процессов. В этом случае возможны четыре варианта развития событий (рис. 4.6):

1) пользователям приходится смириться с отсутствием некоторых функциональных возможностей программного обеспечения или их несоответствием новым реалиям и выполнять их либо вручную, либо при помощи подручных инструментов, например Word и Excel. Соответственно возрастает трудоемкость обработки, вероятность ошибок (человеческий фактор), требуется обучение и коррекция мотивации персонала, а главное — теряется драгоценное время;

2) в обход ограничений системы (врожденных или приобретенных в ходе эволюции бизнеса) создаются дополнительные модули или компоненты («заплатки»). Здесь к риску допуска новых ошибок добавляются издержки на исследование «обходных путей», их программирование и отладку;

3) если первые два варианта решения неприемлемы, то тогда разработчик начинает реализацию изменений на уровне исходных текстов, что требует ресурсов, а главное — времени на завершение полного цикла разработки, тестирование и разворачивание новой версии системы;

4) наиболее пессимистичное развитие ситуации, когда критичные для заказчика требования не только не покрываются системой и всеми ее штатными средствами настройки, но и разработчик в силу различных причин не берется реализовать их путем доработок или переработок программного кода и предлагает заново осуществлять выбор новой платформы и разрабатывать новое решение. Это может привести просто к катастрофе бизнеса, сокращению доли рынка, реальных доходов от продаж, снижению конкурентных преимуществ.

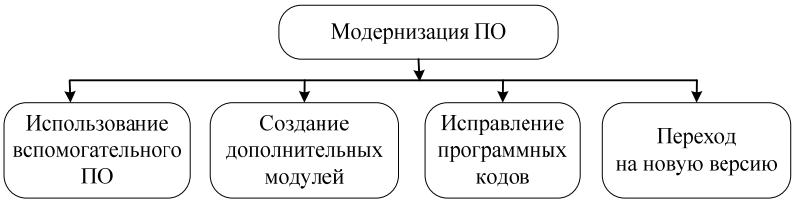

Рис. 4.6. Варианты модернизации ПО

4.2.3. Риски

Минимизируйте риски и уверенно смотрите в будущее.

Под риском понимают незапланированное отклонение от заданных (целевых) показателей доходов и затрат. К минимизации рисков стремятся как разработчики (поставщики) ПО, так и покупатели, а также предполагаемые инвесторы, вкладывающие финансовые средства в разработку программных систем. Основные факторы, определяющие риски, представлены на рис. 4.7. Одним из важнейших факторов, влияющим на риски разработчика, являются ошибки в планировании объемов продаж.

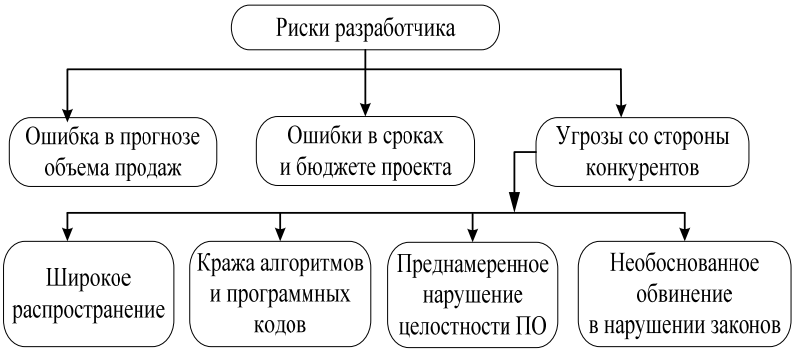

Рис. 4.7. Факторы, определяющие риски разработчика

Возможные ошибки в объемах продаж непосредственно связаны с непредвиденной конкуренцией. Практические рекомендации по минимизации рисков в этом направлении сводятся к следующему:

- поставляемый продукт не должен быть полным аналогом существующих на рынке программных систем;
- рекомендуется первым завершить разработку и захватить рынок, даже если это приведет к увеличению затрат на первый экземпляр.

Кроме того, риски сильно зависят от неоправданных увеличений затрат на разработку, рекламу, влияют на риски и компьютерное пиратство. Снижение рисков на этапе разработки может

быть достигнуто за счет эффективного использования в проекте готовых компонентов, а также постоянной оценки себестоимости разработки с возможностью внесения корректировок в случае незапланированных отклонений.

Следует принять ряд мер по возможной дискредитации программного продукта со стороны конкурентов, которая может осуществляться в следующих направлениях: нарушение целостности продукта, кражи и присвоение алгоритмов и программных кодов, пиратское распространение копий, необоснованное обвинение в нарушении права интеллектуальной собственности.

Риски заказчика связаны, как правило, с ошибочным определением потребительской ценности ПО, просчетами при оценке стоимости проекта и будущих текущих затрат, неудачным выбором разработчика (поставщика). При решении последней задачи рекомендуется руководствоваться критериями, представленными на рис. 4.8.

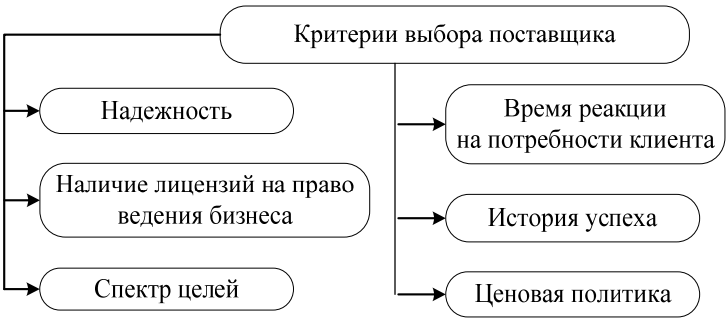

Рис. 4.8. Критерии выбора поставщика

Эти критерии необходимо закладывать в конкурсную документацию при объявлении конкурса либо аукциона на поставку (разработку) программного обеспечения.

4.3. Определение и анализ рыночной стоимости программного обеспечения

Надежная рентабельность бизнеса — это правильный выбор рыночной цены и качественный прогноз объемов рынка.

4.3.1. Концепция безубыточности

Успех любой компании измеряется величиной полученной прибыли и ее динамикой. Для успешного ведения бизнеса необходимо не только просчитывать, сколько компания заработает при достижении запланированного объема продаж, но и четко представлять, какой минимальный объем продаж необходим для обеспечения безубыточной работы.

Задача состоит в определении такого объема продаж, ниже которого предприятие будет терять деньги, выше которого — зарабатывать. Этот минимально допустимый объем продаж, покрывающий все затраты на изготовление продукции, не принося при этом ни прибыли, ни убытков, получил название ***точки безубыточности*** или ***точки равновесия*** (break-event point).

С точки зрения экономической теории безубыточность есть нормальное состояние фирмы на современном конкурентном рынке, находящемся в состоянии долгосрочного равновесия. При этом принимается в рассмотрение экономическая прибыль, т. е. такое определение прибыли, при котором в расходы фирмы включается среднерыночная ставка дохода на инвестированный капитал, а также нормальный доход предприятия.

При таких допущениях определение безубыточности звучит следующим образом: **«Точка безубыточности —** это такой объем продаж продукции фирмы, при котором выручка от продаж полностью покрывает все расходы на производство продукции, в том числе среднерыночный процент на собственный капитал фирмы и нормальный предпринимательский доход» [17].

В этом случае, если фирма имеет бухгалтерскую прибыль, т. е. сальдо доходов от продаж и денежных затрат на производство проданной продукции положительно, и при этом прибыль фирмы меньше, чем среднерыночный процент на собственный капитал фирмы, то она может не достичь точки безубыточности в смысле экономической прибыли. Следовательно, существуют более

выгодные способы использования капитала, которые позволяют получить более высокую прибыль.

Таким образом, понятие точки безубыточности является одновременно и неким критерием эффективности деятельности фирмы. Фирма, не достигающая точки безубыточности, действует неэффективно с точки зрения сложившейся рыночной конъюнктуры. Однако этот факт сам по себе не служит однозначной причиной для прекращения существования фирмы. Для того чтобы решить вопрос о дальнейшей деятельности фирмы, необходимо детально исследовать структуру ее издержек.

Из курса микроэкономической теории известно, что, начиная с некоторого объема выпуска, кривая переменных издержек будет возрастающей, а кривая предельного дохода — убывающей. Очевидно, что в случае, когда предельные издержки меньше предельного дохода, увеличение выпуска повлечет за собой увеличение дохода фирмы. В случае же когда предельные издержки больше предельного дохода, увеличение предельного дохода фирмы приведет к уменьшению выпуска продукции.

Таким образом, можно сформулировать ***критерий точки безубыточности***: максимальная прибыль достигается фирмой при таком объеме производства, при котором предельный доход равен предельным издержкам.

4.3.2. Виды и составляющие издержек

В рамках определения уровня безубыточности все **затраты** разделяют на две группы: ***условно-переменные*** и ***условно-постоянные***.

FC *(Fixed Cost) — постоянные (фиксированные) издержки —* денежные издержки, в целом не изменяющиеся в зависимости от изменения объёма выпускаемой продукции.

К основным составляющим фиксированных издержек можно отнести следующие виды затрат:

- зарплата руководящего состава и административного персонала (служащих);
- аренда и лизинг;
- амортизация зданий и оборудования;

- налоги;
- платежи за внешние услуги;
- плата за телефон и коммунальные услуги;
- плата за страхование;
- уплата процентов по кредитам;
- общие административные расходы.

При достижении определенного объема производства постоянные издержки могут быть увеличены на определенную величину.

***VC** (Variable Cost) — переменные издержки* — издержки, меняющиеся пропорционально объему производства, а именно:

- затраты на сырье и труд основных производственных рабочих (заработная плата);
- комиссионные отчисления торговым агентам;
- затраты на приобретение тары, упаковочного материала;
- транспортные расходы.

Деление затрат на постоянные и переменные дает возможность финансовому менеджеру определить сроки окупаемости затрат, определить запас финансовой прочности предприятия, рассчитать оптимальную величину прибыли предприятия.

Кроме того, иногда выделяют и смешанные издержки (постоянно-переменные затраты), которые включают в себя элементы как постоянных, так и переменных расходов: оплата топлива, почтовые расходы, телефон, отопление, затраты на текущий ремонт оборудования, электроэнергию и т. д. Постоянно-переменные затраты увеличиваются при увеличении объема производства, но не пропорционально его росту. При конкретных расчетах необходимо выделять в составе смешанных издержек постоянную и переменную части, причисляя их к соответствующему виду затрат. Текущие затраты, обеспечивающие жизнедеятельность предприятия, — это постоянная составляющая постоянно-переменных затрат, а затраты, связанные с развитием производства, — это переменная составляющая.

***TC** (Total Cost) — полные издержки* — сумма постоянных и переменных издержек.

Маржа на продажах **C** *(Contribution margin)* — выручка за вычетом переменных издержек, в других терминах — средства, получаемые в распоряжение на операциях производства (закуп-

ки) и продажи партии товара. Причем эти средства необязательно вкладывать на покрытие постоянных издержек. Предприятие будет рентабельным при условии, если маржинальный доход будет выше постоянных затрат.

Маржинальные издержки ***MC*** *(Marginal Cost)* — издержки производства на выпуск дополнительной продукции, при условии, что постоянные (фиксированные) издержки фирмой уже погашены.

Невозвратные издержки ***SC*** *(Sunk Cost)* — одноразовые невозвратные издержки.

Упущенная выгода ***OC*** *(Opportunity Cost)* — доход, который можно было бы получить, но он не получен из-за отказа от использования предоставленной возможности.

4.3.3. Определение точки безубыточности

Основным методом определения точки безубыточности является **CVP**-анализ (*Cast Value Profit* — затраты, объем, прибыль) [8], основанный на анализе соотношений затрат и объемов выпуска и прибыли.

Чистая прибыль фирмы определяется как разница между выручкой и переменными и постоянными издержками.

$$P = s \cdot x - (a + b \cdot x) = (s - b) \cdot x - a, \quad (4.12)$$

где P — прибыль фирмы;

x — объем выпуска продукции;

S — договорная цена продажи единицы продукции;

a — величина фиксированных расходов;

b — величина переменных издержек на единицу продукции.

Объем выпуска, при котором достигается точка безубыточности (прибыль равняется нулю), определяется по формуле

$$x_0 = \frac{a}{s - b}. \quad (4.13)$$

В случае если объем рынка задан, можно определить рыночную цену единицы продукции при нулевом уровне прибыли по формуле

$$s_0 = \frac{a + bx_0}{x_0}. \quad (4.14)$$

Если фирма планирует получить дополнительную прибыль (сверхнормативную) и рыночная цена известна, то объем продаж при заданном уровне прибыли P_0 и рыночной цене s_0 можно определить по формуле

$$x_0 = \frac{P_0 + a}{s_0 - b}. \qquad (4.15)$$

В некоторых случаях в качестве исходных данных при определении необходимого объема продаж используется величина предельного (маржинального) дохода m_0 от реализации единицы продукции как разницы между ценой и переменными издержками: $m_0 = s_0 - b$.

Тогда объем продаж определяется выражением

$$x_0 = \frac{P_0 + a}{m_0}. \qquad (4.16)$$

Графическая интерпретация определения и анализа точки безубыточности представлена на рис. 4.9.

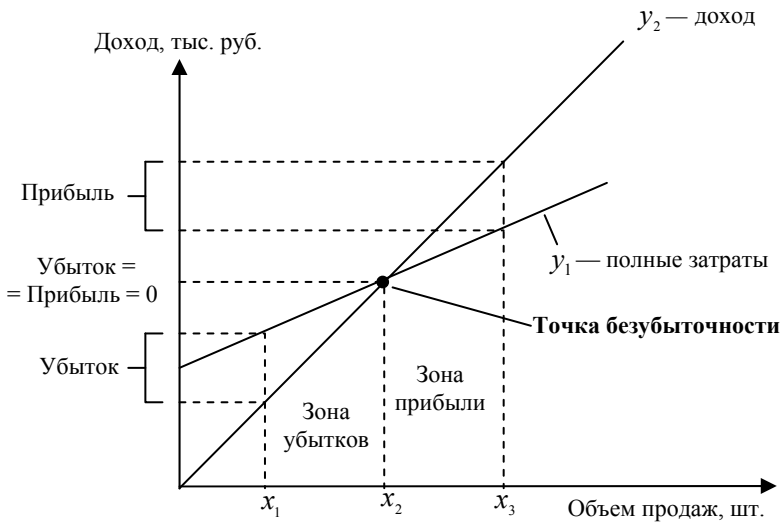

Рис. 4.9. Графическая интерпретация определения и анализа точки безубыточности

Следует отметить, что корректное использование методов CVP-анализа верно лишь в ограниченном диапазоне объемов выпуска. Ограниченность проистекает, прежде всего, из-за того, что при достаточно большом объеме выпуска продукции перестают быть верными многие предпосылки, лежащие в основе CVP-анализа, например, неизменный характер и величина постоянных расходов, и т. д. В этой связи в литературе выделяют следующие условия применимости CVP-анализа при определении точки безубыточности:

- изменение общих затрат и выручки жестко определено и линейно в пределах области релевантности;
- цены на продукцию, материалы и услуги, используемые в производстве, неизменны;
- производительность труда специалистов постоянна;
- ассортимент продукции и объемы выпуска каждого вида неизменны — отсутствуют структурные сдвиги в производстве;
- на конец анализируемого периода объем производства равен объему продаж, либо изменения начального и конечного уровня запасов незначительны.

Контрольные вопросы

1. Как убедить руководство компании вкладывать средства в развитие информатизации?
2. Приведите и обоснуйте набор частных показателей экономической эффективности, характерных для вашего проекта.
3. Как убедить инвестора вложить финансы в ваш проект?
4. Как убедить руководство компании в том, что инфраструктура информатизации вашей фирмы нуждается в развитии?
5. Как увеличить доходы разработчика, не ущемляя при этом интересы заказчика?
6. Назовите основные направления по сокращению текущих расходов заказчика.
7. Раскройте нормативно-правовые аспекты сокращения рисков разработчика при недобросовестной конкуренции.
8. Как обосновать размеры кредита на разработку программного продукта, исходя из концепции безубыточности фирмы?

Вместо заключения

Сегодня как никогда особенно актуальна проблема приобретения и использования лицензионного программного обеспечения. Уже после завершения работы над учебным пособием автором был проанализирован материал по данной проблеме в недавно вышедших в свет периодических изданиях, который с дополнениями предлагается вместо заключения как представляющий несомненный интерес для любого специалиста, работающего в области информационных технологий.

Управление лицензиями на программное обеспечение

Эффективно управлять лицензиями на ПО — значит быть уверенным в завтрашнем дне, избегать большого числа рисков и в итоге иметь более предсказуемый и более устойчивый, а следовательно, и более прибыльный бизнес [18].

Лицензирование программ — это процедура, позволяющая организации приобрести, установить и использовать программное обеспечение на отдельном компьютере или в сети соответственно лицензионному соглашению с производителем этого программного обеспечения. **Цель лицензирования** — защитить как инвестиции компании разработчика, минимизировав вероятность его взлома пиратами, так и инвестиции предприятия, снизив риск наказания за использование пиратского программного обеспечения. Как правило, разработчик реализует лицензионное соглашение путем встраивания в продукт специальных механизмов, не позволяющих использовать программу в случае нарушения пользователем каких-либо пунктов этого соглашения.

Нормативным документом, регламентирующим правила и условия взаимодействия разработчика (продавца) и пользователя (покупателя), является лицензионное соглашение.

Процесс управления лицензиями включает в себя несколько этапов.

1. Проведение инвентаризации установленного программного обеспечения. Инвентаризация имеющегося программного обеспечения может проходить по следующим сценариям:

- если компьютеры компании *не объединены в сеть*, инвентаризацию каждого компьютера придется производить вручную, сканируя жесткий диск и занося полученную информацию в отчет;
- если компьютеры клиента *объединены в сеть*, то необходимо воспользоваться программами для инвентаризации ПО, которые автоматически просканируют жесткие диски рабочих станций и серверов. Программы создают отчеты обо всем найденном ПО, которые можно взять за основу при выборе модели лицензирования и разработке регламентов по управлению лицензиями [19].

2. Выбор моделей лицензирования. Основные модели лицензирования представлены в нижеприведенной таблице.

Наименование модели	Основные особенности модели
1. Пакетная	Лицензия выдается одному пользователю либо на один компьютер
2. Сетевая	Лицензия выдается на определенное количество пользователей (компьютеров) в сети
3. По подписке	Лицензия приобретается в сетевой версии модели на определенный период времени
4. Повременная	Лицензия приобретается в сетевой версии модели с оплатой за время фактического использования программы

Традиционно защита программ от пиратского копирования при пакетной модели обеспечивалась различными механизмами блокировки, реализуемыми через IP-адреса, адреса хостингов компьютеров, серийные номера системных дисков, специальные аппаратные ключи.

В настоящее время интенсивно используется технология распространения лицензий через Internet, которая позволяет самим пользователям при запуске пользовательской программы и указании серийного номера устройства или кода активации активизировать лицензионные ключи и получать информацию о разрешении на использование программы.

Сетевая модель лицензирования выросла из необходимости развертывания множества программных продуктов на настольных

и мобильных компьютерах сотрудников. Наиболее известны две модели — лицензия на одновременный доступ пользователей (concurrent), называемая также «плавающей», и лицензирование по сетевым именам (network-named).

В сетевой модели ответственность за управление лицензированием программ возлагается на серверные приложения. Каждый клиент в определенные интервалы времени выходит на связь с сервером лицензий (так называемый «пульс»). Если лимит доступных лицензий исчерпан, клиенту будет отказано в праве использовать лицензию, а следовательно, и сам продукт.

Лицензия с одновременным доступом обеспечивает определенную гибкость, позволяя предприятию купить пул лицензий, которые оно само может впоследствии распределять между пользователями компьютеров, на которых применяется данный продукт.

Лицензии на сетевые имена, в общем, менее дорогостоящи, чем лицензии с одновременным доступом, но дают меньшую гибкость, так как каждая лицензия выделяется для одного пользователя и не предназначена для совместного доступа. Вместо этого их работа основана на списках, отражающих связь отдельных пользователей с доступом к лицензируемому продукту.

Сетевая модель наиболее предпочтительна для крупных организаций и больше соответствует требованиям разработчиков, чем традиционные пакетные модели. Последние обычно не справляются с реалиями больших организаций, где сотрудники устраиваются на работу и увольняются, и становится все труднее обеспечивать мониторинг и поддержку программ на каждом компьютере. На передний план выходят механизмы, обеспечивающие автоматическое восстановление сетевых лицензий, так как они предлагают более щадящий метод учета установленных программных продуктов.

Обратная сторона сетевого лицензирования — дополнительный сетевой трафик, который является результатом информационного обмена между клиентом и сервером. Сетевая модель требует постоянного обмена данными между клиентским приложением и сервером лицензий. При отсутствии правильного регулирования этот обмен может стать настолько интенсивным, что вызовет отказ сети.

Пакетное сетевое лицензирование осуществляется в двух видах: пожизненная лицензия и пробная лицензия. Пожизненные лицензионные соглашения, как правило, включают техническую поддержку, доработки и обновления версий на какой-то определенный период. Пробные лицензии обычно не включают никакой технической поддержки и делают программу недоступной по истечении 30–60 дней с момента установки либо до тех пор, пока покупатель и разработчик не перейдут к пожизненному лицензированию.

В [19] отмечается, что пожизненное лицензирование в настоящее время доминирует на рынке. Вместе с тем набирают популярность новые модели: повременная модель и модель по подписке.

При повременном моделировании организации покупают необходимое количество лицензий с оплатой за время использования, а компания-разработчик назначает плату за каждое использование соответственно тому, как это понятие определено в лицензионном соглашении. Такая модель лицензирования хорошо интегрируется с сетевой моделью с одновременным доступом, поскольку система управления сетевыми лицензиями может регистрировать каждый факт использования продукта и обновлять внутренние параметры, чтобы вовремя заметить, что пользователи превысили лимит использования. Предприятие может настроить систему с одновременным доступом таким образом, чтобы она регистрировала каждый факт использования программного обеспечения и представляла отчеты уполномоченным сотрудникам, а они, в свою очередь, периодически направляли эти отчеты компании-разработчику.

В рамках модели по подписке организация приобретает лицензии на определенный срок (обычно на год), в течение которого автоматически получает обновления и дополнительные возможности. Плата, как правило, вносится за год. Когда срок действия лицензии истекает, пользователь перестает получать обновления до тех пор, пока предприятие не возобновит подписку. Модель лицензирования по подписке основана на предположении, что непрерывные автоматические обновления выгодны пользователям и разработчикам. Стоимость дополнительных ресурсов, как правило, невысока, и это несет в себе еще больше

возможностей для максимизации возврата инвестиций в программное обеспечение по сравнению с традиционными моделями.

В качестве примера перечислим основные модели лицензирования фирмы Microsoft [18]:

OEM-версия программного обеспечения, которая может быть приобретена с новым компьютером. Лицензионность ПО подтверждают:

- лицензионное соглашение Microsoft с конечным пользователем;
- сертификат подлинности — COA (сертификат подлинности ОС Windows должен быть наклеен на корпус компьютера);
- исходные носители;
- руководство пользователя (если прилагается);
- квитанция или счет;

«коробочные» продукты, которые можно покупать в розницу в магазинах и у партнеров Microsoft. Лицензионность ПО подтверждают:

- индивидуальное лицензионное соглашение с конечным пользователем (EULA);
- исходные носители;
- руководство пользователя (если прилагается);
- квитанция или счет;

Open License, предназначенная для организаций, которые приобретают не менее пяти лицензий;

Microsoft Multi-Year Open License — программа корпоративного лицензирования, позволяющая организациям, имеющим пять и более компьютеров, приобрести лицензии и ключевые продукты Microsoft в рассрочку;

Microsoft Open Subscription License — корпоративное лицензионное соглашение, предоставляющее организациям, имеющим пять и более ПК, уникальную возможность лицензировать основные продукты Microsoft для настольных компьютеров с минимальными единовременными затратами;

Microsoft Enterprise Agreement и *Microsoft Enterprise Agreement Subscription* — корпоративное лицензирование, предназначенное для организаций, выбравших платформу Microsoft в качестве корпоративного стандарта, с числом компьютеров от 250.

3. Разработка регламента по управлению лицензиями организации. При разработке регламента следует подробно описать следующие процедуры: порядок приобретения лицензионного ПО, порядок использования ПО, процедуру автоматизированной установки ПО, процедуру хранения и обновления лицензионного программного обеспечения и документации на него, права и обязанности ответственных лиц за управление лицензиями компании, технологию ведения инвентаризационной базы данных для контроля за лицензиями, процедуру периодического мониторинга (официального и неофициального) используемого ПО в подразделениях компании.

Грамотное управление лицензиями на программное обеспечение обеспечивает:

1) эффективное использование финансовых ресурсов на приобретение ПО.

В организациях, где отсутствует системный взгляд на управление лицензиями, часто происходит дублирование в приобретении лицензионного программного обеспечения, отсутствует возможность приобретения корпоративных лицензий, что не позволяет получать определенные скидки или рассрочки в платежах;

2) сведение до минимума юридических и технических рисков при эксплуатации ПО.

Своевременный мониторинг состояния ПО в структурных подразделениях компании позволит: препятствовать появлению нелицензионных копий, обеспечить соблюдение авторских прав на программы для ЭВМ, свести до минимума вероятность сбоев в программном обеспечении и несовместимость форматов данных различных версий одноименных программных продуктов;

3) возможность легального и законного бизнеса.

Это касается как компаний, использующих информационные технологии в качестве инструментария по поддержке и принятию решений, так и компаний, работающих на рынке информационных технологий. Последние, используя в своей деятельности лицензионное ПО, могут активно включаться в российский и международный бизнес и систему проектирования, активно участвовать в различных выставочно-ярмарочных мероприятиях, пройти сертификацию по стандартам ISO9000;

4) централизованное управление лицензиями, которое позволяет:

- хранить все соглашения и лицензии в одном месте (это облегчит поиск);
- сократить расходы (приобретая только нужные виды лицензий);
- проверять соответствие бюджета и реальных расходов на ПО (это позволит более эффективно использовать имеющиеся средства);
- повторно использовать ПО, передавая его в другие отделы (предварительно проверив, допускают ли это условия лицензионных соглашений).

Контрольные вопросы

1. Определите, к каким классическим моделям лицензирования относятся модели лицензирования на фирме Microsoft.
2. Что дает компании грамотный подход к управлению лицензиями?

Литература

1. Ехлаков Ю.П. Теоретические основы автоматизированного управления: учеб. пособие / Ю.П. Ехлаков. — Томск: Томск. гос. ун-т систем управления и радиоэлектроники, 2001. — 337 с.

2. Ракитов А. Выход российского программного продукта на международные рынки возможен / А. Ракитов, А. Райков // Информационное общество. — 2000. — № 2. — С. 47–53.

3. Комаров К. Как продавать программы / К. Комаров // Подводная лодка. — 1998. — № 2. — С. 14–18.

4. Везиров В. О механизмах настройки отношений на рынке научно-технической информации / В. Везиров, А. Переслегин // Информационные ресурсы России. — 2000. — № 6. — С. 4–14.

5. Ехлаков Ю.П. Особенности развития рынка прикладного программного обеспечения / Ю.П. Ехлаков // Промышленные контроллеры АСУ. — 2002. — № 6. — С. 23–24.

6. Смольянинов А. Некоторые секреты командной разработки / А. Смольянинов, А. Ложечкин // Открытые системы. — 2005. — №№ 7, 8. — С. 49–56.

7. Липаев В.В. Системное проектирование сложных программных средств для информационных систем / В.В. Липаев. — М.: СИНТЕГ, 1999. — 268 с.

8. Липаев В.В. Технико-экономическое обоснование проектов сложных программных систем / В.В. Липаев. — М.: СИНТЕГ, 2004. — 284 с.

9. Фатрелл Роберт Т. Управление программными проектами. Достижение оптимального качества при минимуме затрат / Роберт Т. Фатрелл, Дональд Ф. Шафер, Линда И. Шафер. — М.: Издательский дом «Вильямс», 2004. — 1136 с.

10. Стрельцов И.А. Оценка проектов по созданию программных систем: теория и практика / И.А. Стрельцов // Корпоративные системы. — 2004. — № 3. — С. 53-60.

11. Сборник временных норм на работы по ведению Государственного мониторинга геологической среды, информационной деятельности, цифровому картографированию. — М.: Министерство природных ресурсов Российской Федерации, 2000. — 24 с.

12. Жимерин Д.З. Автоматизированные и автоматические системы управления / Д.З. Жимерин, В.А. Мясников. — М.: Энергия, 1975. — 592 с.

13. Огипесен А. Время и деньги / А. Огипесен // Открытые системы. — 2004. — № 7. — С. 34–38.

14. Елашкин М. Как оценить эффективность в IT? / М. Елашкин // Открытые системы. — 2004. — № 7. — С. 38–42.

15. Карпович И.А. Экономическая оценка эффективности инновационных проектов: науч.-метод. издание / И.А. Карпович. — Новосибирск: Изд-во НГТУ, 2002. — 36 с.

16. Мессеригмин Д. Дж. Продукт от слова продавать (Вопросы рынка в планировании и проектировании программного обеспечения) / Д. Дж. Мессеригмин, К. Неиперски // Открытые системы. — 2004. — № 7. — С. 26–33.

17. Алферова Л.А. Маркетинг: учеб. пособие / Л.А. Алферова. — Томск: Томск. гос. ун-т систем управления и радиоэлектроники, 2005. — 251 с.

18. Гайдаманчук Р. Управление лицензиями на программное обеспечение / Р. Гайдаманчук. — Корпоративные системы. — 2004. — № 2. — С. 38–40.

19. Ферранте Дэниел. Модели лицензирования программ: что дальше? / Дэниел Ферранте. — Открытые системы. — 2007. — № 1. — С. 40–45.