ТЕМА 3. ИНЖЕНЕРИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ (ПРОГРАММНАЯ ИНЖЕНЕРИЯ)

Понятие программной инженерии.

Различия между программной инженерией (software engineering) и информатикой (computer science).

Различия между программной инженерией и системной инженерией (systems engineering).

Базовые процессы разработки программных продуктов. Модели процесса создания программного обеспечения информационных систем. Структура затрат на создание программного обеспечения. Методы инженерии программного обеспечения. Основные проблемы, стоящие перед специалистами по программному обеспечению.

Содержание лекции

3.1 История появления. Определение программной инженерии.
3.2. SWEBOK: Руководство к своду знаний по программной инженерии
3.3. Различия между программной инженерией (software engineering) и информатикой (computer science).

- **Различия между программной инженерией и системной инженерией (systems engineering).**
- **Базовые процессы разработки программного обеспечения.**
- **Структура затрат на создание ПО.**
- **Методы инженерии программного обеспечения.**
- **Основные проблемы, стоящие перед специалистами по программному обеспечению.**
- **Кодекс этики и практической деятельности инженерии программного обеспечения.**

3.1 История появления

В конце 90-х годов прошлого века знания и опыт, которые были накоплены в индустрии программного обеспечения за предшествующие 30-35 лет, а также более чем 15-летних попыток применения различных моделей разработки, все это, наконец, оформилось в то, что принято называть дисциплиной программной инженерии – Software Engineering. В какой-то мере, такое формирование дисциплины на основе широко распространенного практического опыта напоминает те процессы, которые происходили в управлении проектами. Возникали и развивались профессиональные ассоциации, специализированные институты, комитеты по стандартизации и другие образования, которые, в конце концов, пришли к общему мнению о необходимости сведения профессиональных знаний по соответствующим областям и стандартизации соответствующих программ обучения

Термин программная инженерия был предложен Ф.Л. Бауэром в 1968 г.
В 1972 году IEEE (Computer Society of the Institute for Electrical and Electronic Engineers, Институт инженеров по электронике) выпустил первый номер Transactions on Software Engineering – Труды по Программной Инженерии. Первый целостный взгляд на эту область профессиональной деятельности появился 1979 году, когда Компьютерное Общество IEEE подготовило стандарт IEEE Std 730 по качеству программного обеспече-ния. После 7 лет напряженной работы, в 1986 году IEEE выпустило IEEE Std 1002 “Taxonomy of Software Engineering Standards”

В 1990 году началось планирование всеобъемлющих *международных* стандартов, в основу которых легли концепции и взгляды стандарта IEEE Std 1074 и результатов работы образованной в 1987 году совместной комиссии ISO/IEC JTC 1**. В 1995 году группа этой комиссии SC7 “Software Engineering” выпустила первую версию международного стандарта ISO/IEC 12207 “Software Lifecycle Processes”. Этот стандарт стал первым опытом создания единого общего взгляда на программную инженерию. Соответствующий национальный стандарт России – ГОСТ Р ИСО/МЭК 12207-99 [ГОСТ 12207, 1999] содержит полный аутентичный перевод текста международного стандарта ISO/IEC 12207-95 (1995 года).

В свою очередь, IEEE и ACM(Association of Computer Machinery, Всемирная научная и образовательная организация в области вычислительной технике), начав совместные работы еще в 1993 году с кодекса этики и профессиональной практики в данной области (ACM/IEEE-CS Code of Ethics and Professional Practice), к 2004 году сформулировали два ключевых описания того, что сегодня мы и называем основами программной инженерии – Software Engineering: ***Guide to the Software Engineering Body of Knowledge (SWEBOK)***, *IEEE 2004 Version* - Руководство к Своду Знаний по Программной Инженерии, в дальнейшем просто “SWEBOK” [SWEBOK, 2004]; *Software Engineering 2004. Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering* – Учебный План для Преподавания Программной Инженерии в ВУЗах* (данное название на русском языке представлено в вольном смысловом переводе) [SE, 2004].

Определение программной инженерии:

Систематическое применение научных и технических знаний, методов и опыта для разработки, реализации, тестирования и документирования программного обеспечения. ISO/IEC 2382-1.

Применение систематизированного, упорядоченного, количественно измеримого подхода к разработке, эксплуатации и сопровождению программного обеспечения, что означает применение инженерии к программному обеспечению. ISO/IEC 24765.

По Соммервиллу:
Инженерия программного обеспечения — это инженерная дисциплина, которая охватывает все аспекты создания ПО от начальной стадии разработки системных требований через создание ПО до его использования. В этом определении присутствует две ключевые фразы.
"Инженерная дисциплина". Инженеры — это те специалисты, которые выполняют практическую работу. Они применяют теоретические построения, методы и средства там, где это необходимо, но делают это выборочно и всегда пытаются найти решение задачи, даже если не существует подходящей теории или методов решения. Специалисты-инженеры также всегда понимают, что они должны работать в организационных и финансовых рамках заключенных контрактов, т.е. ищут решение поставленной перед иими задачи с учетом условий контракта.

"Все аспекты создания программного обеспечения". Инженерия программного обеспечения ие рассматривает технические аспекты создания ПО — в ее ведении такие вопросы, как управление проектом создания ПО и разработка средств, методов и теорий, необходимых для создания программных систем.
Можно сказать, что специалисты (инженеры) по программному обеспечению адаптируют существующие методы инженерии ПО к решению своих задач, и зачастую это оказывается наиболее эффективным способом построения высококачественных программных систем.
Инженерия программного обеспечения предоставляет всю необходимую информацию для выбора наиболее подходящего метода для множества практических задач. Вместе с тем творческий неформальный подход в определенных обстоятельствах также может быть эффективным.
Например, при разработке программных систем электронной коммерции в Internet требуется неформальный подход в сочетании ПО и графического эскизного проектирования.
Так мы пришли к сегодняшнему состоянию Software Engineering как дисциплины.

3.2. SWEBOK: РУКОВОДСТВО К СВОДУ ЗНАНИЙ ПО ПРОГРАММНОЙ ИНЖЕНЕРИИ

С 1993 года IEEE и ACM координируют свои работы в рамках специального совместного комитета - Software Engineering Coordinating Committee (SWECC - http://www.computer.org/tab/swecc). Проект SWEBOK был инициирован этим комитетом в 1998 году

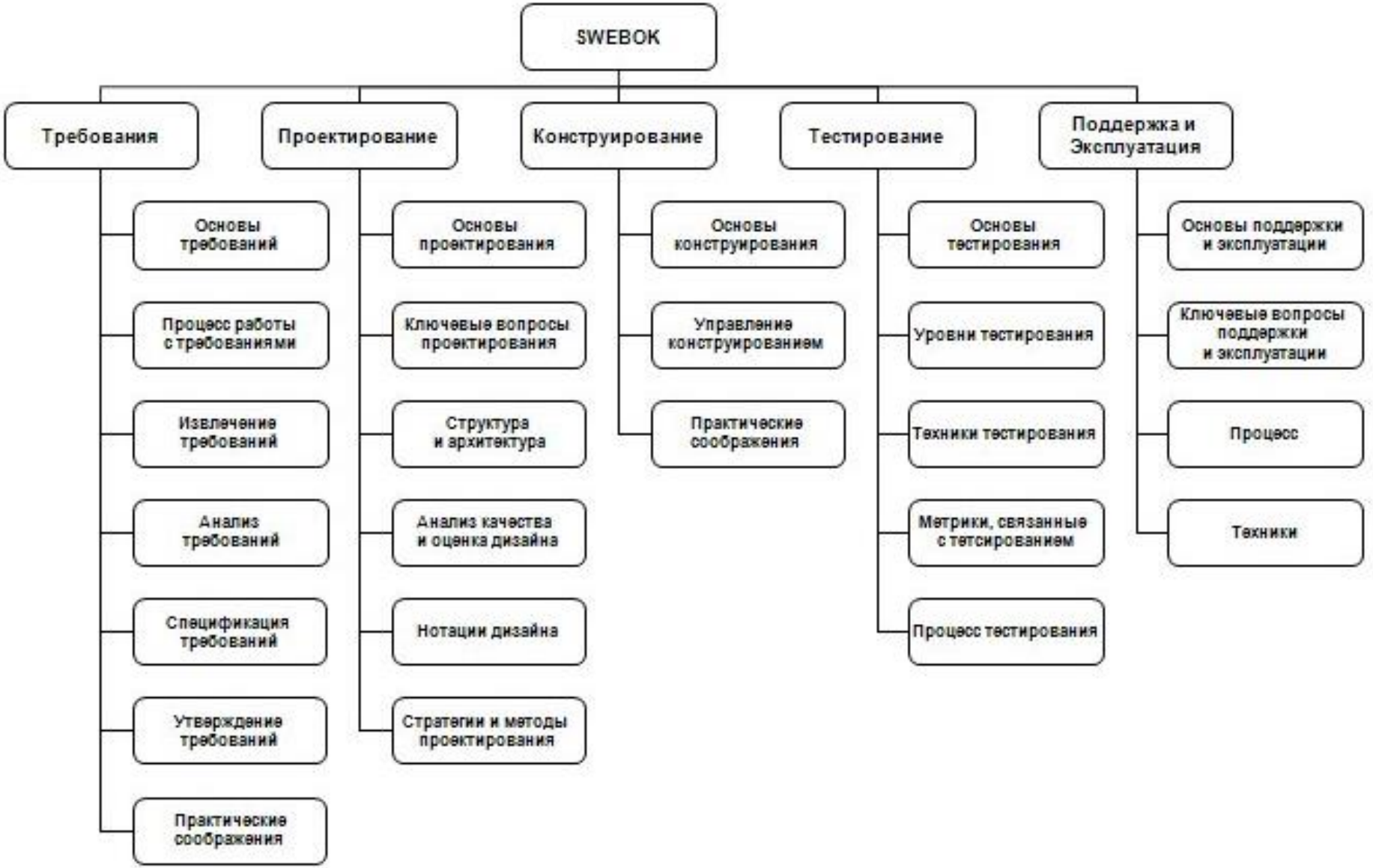
SWEBOK (**Software Engineering Body of Knowledge**) — документ, подготавливаемый комитетом *Software Engineering Coordinating Committee*, в который вовлечено сообщество IEEE Computer Society. Назначение SWEBOK — в объединении знаний по инженерии программного обеспечения (разработке программного обеспечения).

Документ является одним из трёх документов, созданных совместными усилиями IEEE-CS и ACM, призванных обеспечить следующее:
определить необходимый набор знаний и рекомендуемые практики;
определить этические и профессиональные стандарты;
определить учебную программу для студентов, аспирантов и продолжающих обучение.

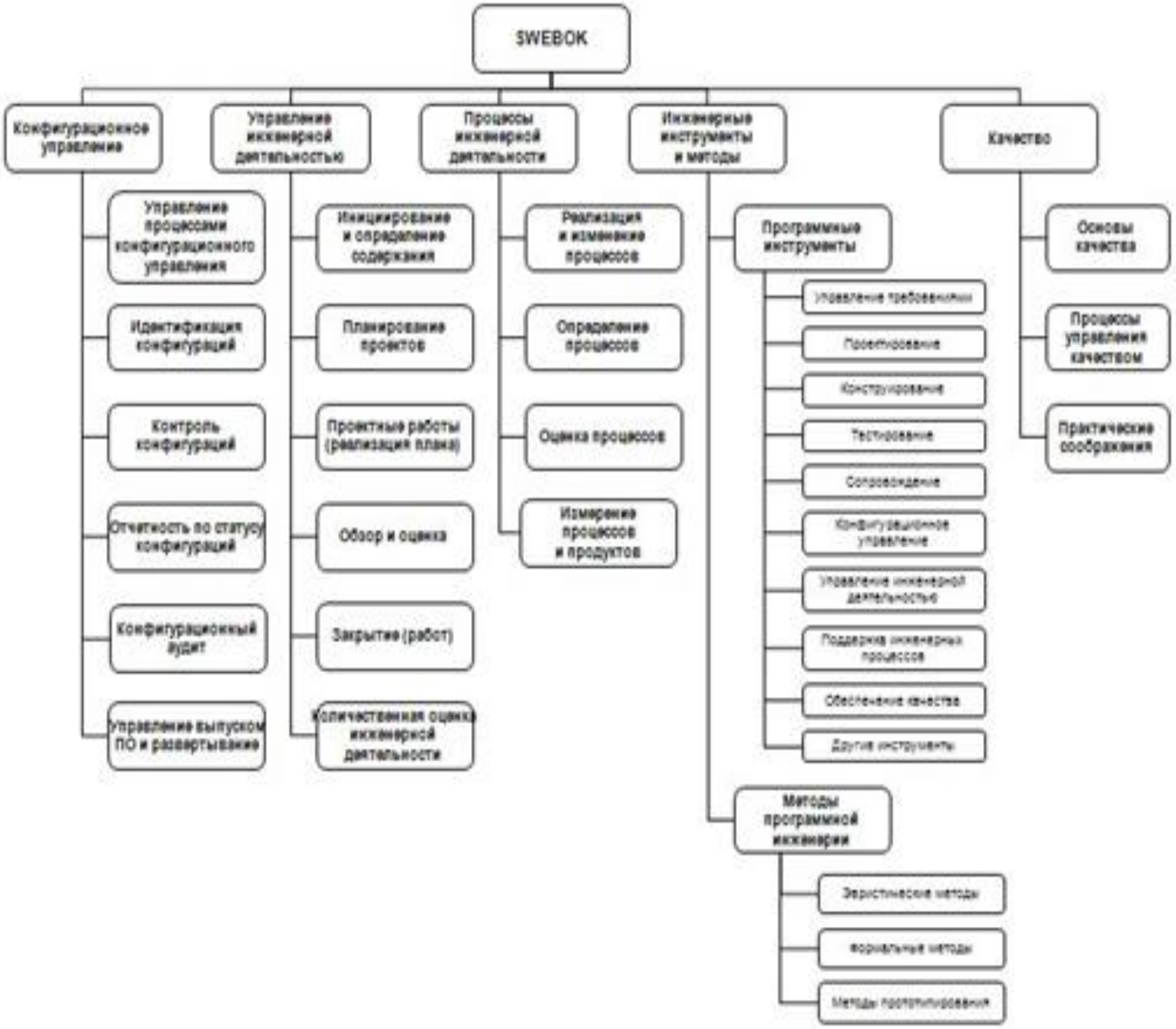
Документ **SWEBOK** делит знания по программной инженерии на 10 областей знаний (Knowledge Areas):
Software Requirements — требования к ПО.
Software Design — проектирование ПО.
Software Construction — конструирование ПО.
Software Testing — тестирование ПО.
Software Maintenance — сопровождение ПО.
Software Configuration Management — управление конфигурацией.
Software Engineering Management — управление IT проектом.
Software Engineering Process — процесс программной инженерии.
Software Engineering Tools and Methods — методы и инструменты.
Software Quality — качество ПО.

Программная инженерия является развивающейся дисциплиной. Более того, данная дисциплина не касается вопросов конкретизации применения тех или иных языков программирования, архитектурных решений или, тем более, рекомендаций, касающихся более или менее распространенных или развивающихся с той или иной степенью активности/заметности технологий (например, web-служб).

Руководство к своду знаний, каковым является SWEBOK, включает базовое определение и описание областей знаний (например, конфигурационное управление – configuration management) и, безусловно, является *недостаточным* для охвата всех вопросов, относящихся к вопросам создания программного обеспечения, но, в то же время *необходимым* для их понимания.



Первые пять областей знаний SWEBOK на русском языке



Вторые пять областей знаний SWEBOK на русском языке

3.3. РАЗЛИЧИЯ МЕЖДУ ПРОГРАММНОЙ ИНЖЕНЕРИЕЙ (SOFTWARE ENGINEERING) И ИНФОРМАТИКОЙ (COMPUTER SCIENCE)

Существенное различие заключается в том, что компьютерная наука (computer science) охватывает теорию и методы построения вычислительных и программных систем, тогда как инженерия программного обеспечения акцентирует внимание на практических проблемах разработки ПО. Знание компьютерной науки необходимо специалистам по программному обеспечению так же, как знание физики — инженерам-электронщикам.

В идеале вся инженерия программного обеспечения должна основываться на фундаменте компьютерной науки, но на самом деле это не так. Часто специалисты по программному обеспечению используют подходы, применимые для решения только конкретной задачи (без обобщения на класс подобных задач). Элегантные методы компьютерной науки не всегда можно применить к реальным сложным задачам, требующим программного решения.

РАЗЛИЧИЯ МЕЖДУ ПРОГРАММНОЙ ИНЖЕНЕРИЕЙ И СИСТЕМНОЙ ИНЖЕНЕРИЕЙ (SYSTEMS ENGINEERING).

Системная инженерия (system engineering) или, точнее, технология создания вычислительных систем охватывает все аспекты создания и модернизации сложных вычислительных систем, где программное обеспечение играет ведущую роль. Сюда можно отнести технологию разработки аппаратных средств, внутренних вычислительных процессов и развертывания всей системы, а также технологию создания ПО. Инженеры-системотехники на основе спецификации системы (технических требований) определяют ее архитектуру и затем, собрав воедино ее отдельные части, создают законченную систему. Они рассматривают систему преимущественно как составной объект с заданными компонентами и уделяют сравнительно мало внимания самим системным компонентам (конкретным аппаратным средствам, соответствующему программному обеспечению и т.д.). Системотехника более старая дисциплина, чем инженерия программного обеспечения. Человечество создает сложные индустриальные системы (такие, как железные дороги и химические заводы) уже более 100 лет. Вместе с тем по мере увеличения в системах роли программного компонента методы инженерии программного обеспечения, например автоматизированное моделирование систем, управление разработкой спецификаций и т.п., все шире используются в процессе создания самых разнообразных систем.

БАЗОВЫЕ ПРОЦЕССЫ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Создание ПО — это совокупность процессов, приводящих к созданию программного продукта. Эти процессы основываются главным образом на технологиях инженерии программного обеспечения. Существует четыре фундаментальных процесса, которые присущи любому проекту создания ПО.

Разработка спецификации требований на программное обеспечение. Требования определяют функциональные характеристики системы и обязательны для выполнения.

Создание программного обеспечения. Разработка и создание ПО согласно спецификации на него.

Аттестация программного обеспечения. Созданное ПО должно пройти аттестацию для подтверждения соответствия требованиям заказчика.

Совершенствование (модернизация) программного обеспечения. ПО должно быть таким, чтобы его можно было модернизировать согласно измененным требованиям потребителя.

При выполнении разнообразных программных проектов эти процессы могут быть организованы различными способами и описаны на разных уровнях детализации. Длительность реализации этих процессов также далеко не всегда одинакова. И вообще, различные организации, занимающиеся производством ПО, зачастую используют разные процессы для создания программных продуктов даже одного типа. С другой стороны, определенные процессы более подходят для создания программных продуктов одного типа и менее — для другого типа программных приложений. Если использовать неподходящий процесс, это может привести к снижению качества и функциональности разрабатываемого программного продукта.

СТРУКТУРА ЗАТРАТ НА СОЗДАНИЕ ПО

Точная структура затрат на создание программного обеспечения существенно зависит от процессов, используемых при разработке ПО, а также от типа разрабатываемого программного продукта. Если принять общую стоимость создания ПО за 100 единиц, то распределение стоимостей отдельных этапов производства может иметь такой вид, как на рис. 3.3.

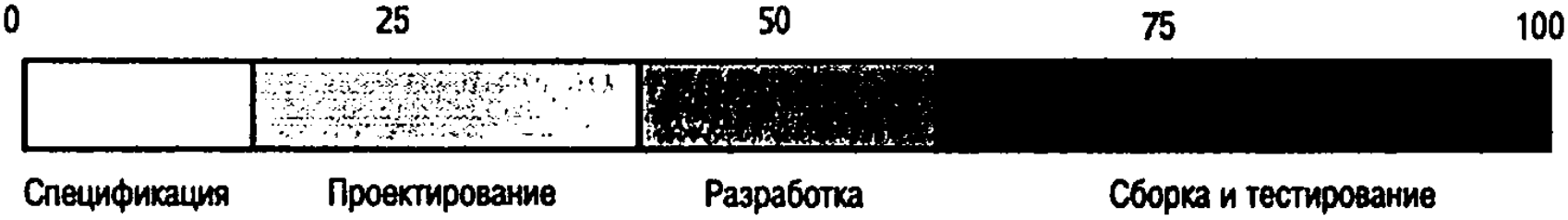


Рис. 3.3. Распределение стоимостей отдельных этапов производства ПО

Примерно такая структура затрат возможна тогда, когда затраты на создание спецификации, проектирование ПО, его разработку и сборку подсчитываются отдельно. Отметим, что часто стоимость этапа сборки и тестирования превышает стоимость этапа непосредственно разработки ПО. Например, на рис. 3.3 показана структура затрат, при которой на тестирование программной системы приходится примерно 40% общей стоимости затрат. Вместе с тем для некоторых критических систем эта статья расходов может превышать 50%.

При использовании эволюционного подхода к разработке ПО практически невозможно провести четкое разграничение между этапами создания спецификации, проектирования и разработки ПО. Поэтому структуру затрат, представленную на рис. 3.3, следует изменить так, как показано на рис. 3.4. Здесь оставлен отдельный этап разработки спецификации, поскольку общая спецификация высшего уровня создается еще до начала создания программного продукта. Создание спецификации нижнего уровня, проектирование, реализация, сборка и тестирование ПО при таком подходе выполняются параллельно на этапе разработки программной системы. Вместе с тем этот подход требует выполнения отдельного этапа тестирования системы после окончания начального этапа ее разработки.

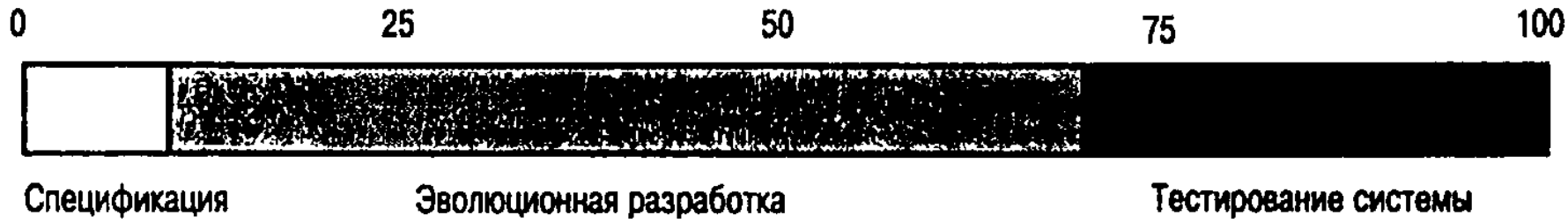
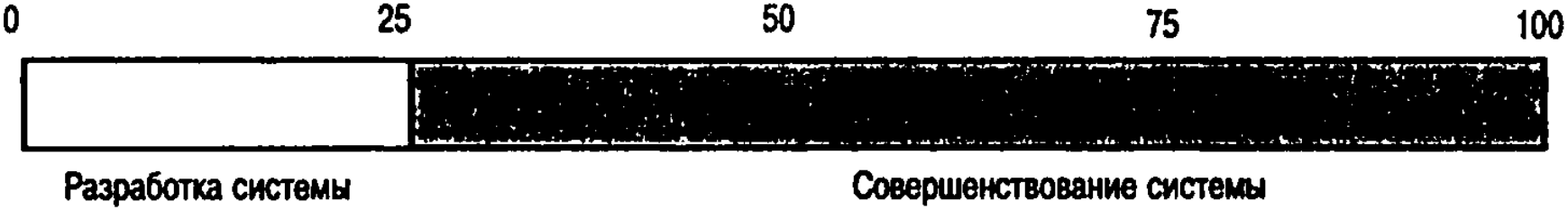


Рис. 3.4. Структура затрат при использовании эволюционного подхода к разработке ПО

В стоимость создания ПО также могут включаться затраты на его модернизацию после начала эксплуатации программного продукта. Для многих программных систем затраты на совершенствование системы могут превышать стоимость разработки в 3 или 4 раза (рис.3.5).



Структура затрат на создание заказного программного обеспечения (т.е. когда требования к системе устанавливаются заказчиком и разработка ПО выполняется по контракту) примерно такая же, как показано выше, но стоимость различных этапов создания программного продукта может значительно разниться. Это относится, в частности, к программам, разрабатываемым для персональных компьютеров. Как правило, такое программное обеспечение разрабатывается на основе эволюционного подхода с использованием уже готового эскиза спецификации. Поэтому стоимость разработки требований к ПО относительно низкая. Вместе с тем такие программные продукты предназначены для работы на разных компьютерных платформах, что существенно повышает затраты на тестирование систем. На рис. 3.6 показана типовая структура затрат на создание такого ПО.

Рис. 3.6. Структура затрат на создание заказного ПО

Стоимость модернизации общих программных продуктов (т.е. тех, которые продаются на открытом рынке программ) с трудом поддается оценке. Во многих случаях осуществляется небольшая формальная модернизация. Обычно с началом реализации созданного программного продукта начинается работа с его следующей версией. Но исходя из требований маркетинга предпочтительнее представить новую версию как новый (но совместимый со старой версией) программный продукт, а не как модифицированную версию того продукта, которую пользователь уже купил. Поэтому стоимость модернизации ПО обычно не подсчитывается отдельно, как это делается при модернизации заказных программных продуктов, а просто входит в стоимость разработки следующей версии программной системы.

Структура затрат на создание систем для электронной коммерции в Internet обычно отличается от того, что описано выше. В таких системах вместо создания программных модулей, управляющих информацией, обычно используют готовое программное обеспечение, а основные затраты приходятся на разработку пользовательских интерфейсов.

МЕТОДЫ ИНЖЕНЕРИИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Методы инженерии программного обеспечения впервые были представлены еще в 1970-х годах. Эти методы, названные функционально-модульными или функционально-ориентированными, связаны с определением основных функциональных компонентов программной системы и в свое время широко использовались. В 80-90-х годах к этим методам добавились объектно-ориентированные методы, предложенные Бучем (Booch) [54] и Рамбо (Rumbaugh) [302]. Эти методы, использующие разные подходы, ныне интегрированы в единый унифицированный метод, построенный на основе унифицированного языка моделирования UML (Unified Modeling Language).

Все упомянутые методы основаны на идее создания моделей системы, которые можно представить графически, и на использовании этих моделей в качестве спецификации системы или ее структуры.

Методы инженерии ПО обычно включают перечисленные в табл. 1 компоненты.

Компонент	Описание	Пример
Описание модели системы	Описания моделей создаваемых систем и нотация, используемая для разработки этих моделей	**Модели объектов, модели потоков данных, модели конечных автоматов и т.п.**
Правила	Правила и ограничения, которые необходимо выполнять при разработке моделей систем	**Каждый элемент модели должен иметь уникальное имя**
Рекомендации	Эвристические советы и рекомендации, отражающие практический опыт применения данного метода	**Любой объект в модели не должен иметь более семи подчиненных ему объектов**
Руководство по применению метода	**Описание работ, которые необходимо выполнить для построения модели системы, а также рекомендации по организации этих работ**	**Атрибуты любого объекта должны быть документированы, прежде чем будут определены операции, связанные с этим объектом**

Не существует идеального и универсального метода — каждый метод имеет свою область применимости. Например, объектно-ориентированные методы часто применяются для создания интерактивных (диалоговых) программных систем, но практически не используются при разработке систем, работающих в режиме реального времени.

ОСНОВНЫЕ ПРОБЛЕМЫ, СТОЯЩИЕ ПЕРЕД СПЕЦИАЛИСТАМИ ПО ПРОГРАММНОМУ ОБЕСПЕЧЕНИЮ.

В XXI столетии специалисты по программному обеспечению столкнутся с описанными ниже проблемами.

Проблема наследования ранее созданного ПО. Многие большие программные системы, эксплуатируемые в настоящее время, созданы много лет назад, но до сих пор выполняют свои функции надлежащим образом. Проблема наследования означает поддержку и модернизацию таких систем, причем при минимальных финансовых и временных затратах.

Проблема все возрастающей разнородности программных систем. В настоящее время программное обеспечение должно быть способно работать в качестве систем, распределенных в компьютерных сетях, состоящих из компьютеров разных типов и использующих различные операционные системы. Проблема возрастающей разнородности программных систем состоит в том, что необходимо разрабатывать надежные программные системы, способные работать совместно с ПО разных типов.

Проблема, порожденная требованием уменьшения времени на создание ПО. Многие традиционные технологии создания качественного программного обеспечения требуют больших временных затрат. Вместе с тем сегодня запросы рынка ПО и требования к программным системам меняются очень быстро. Поэтому и ПО должно меняться с соответствующей скоростью. Проблема, порожденная требованием уменьшения времени на создание ПО, заключается в том, чтобы сократить время на разработку больших и сложных программных систем без снижения их качества.

Конечно, перечисленные проблемы связаны друг с другом. Например, возможна такая ситуация, когда необходимо быстро разработать на основе существующей системы ее сетевой вариант. Для решения таких проблем необходимы новые средства и технологии, которые вобрали бы в себя все лучшие методы современной инженерии программного обеспечения.

КОДЕКС ЭТИКИ И ПРАКТИЧЕСКОЙ ДЕЯТЕЛЬНОСТИ ИНЖЕНЕРИИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ.

Подобно любым другим профессионалам, специалисты по программному обеспечению должны согласиться, что к ним предъявляется более широкий круг требований, чем простая необходимость иметь тот или иной профессиональный уровень. Они работают в определенном правовом и социальном окружении. Область инженерии программного обеспечения, как и любая другая сфера человеческой деятельности, имеет ограничения в виде местных, национальных и международных законодательств. Поэтому специалисты по программному обеспечению должны принять на себя определенные этические и моральные обязательства, чтобы стать настоящими профессионалами.

ACM и IEEE совместно создали кодекс, соединяющий этические нормы и профессиональную практику.

Краткая версия этого кодекса содержит следующие десять принципов, которым должны следовать специалисты по программному обеспечению:
1) не использовать компьютер с целью повредить другим людям;
2) не создавать помех и не вмешиваться в работу других пользователей компьютерных сетей;
3) не пользоваться файлами, не предназначенными для свободного использования;
4) не использовать компьютер для воровства;
5) не использовать компьютер для распространения ложной информации;
6) не использовать ворованное программное обеспечение;
7) не присваивать чужую интеллектуальную собственность;
8) не использовать компьютерное оборудование или сетевые ресурсы без разрешения или соответствующей компенсации;
9) думать о возможных общественных последствиях программ, которые Вы пишите или систем, которые Вы разрабатываете;
10) использовать компьютер с самоограничениями, которые показывают Вашу предупредительность и уважение к другим людям.