Лекции по программной инженерии

Лекция 2. Области знаний в программной инженерии. Основные термины и определения в этой области

Оглавление

2.1 Настольная книга программных инженеров любого профиля 3

2.2 Знания и требования к программным инженерам в области этики 9

2.2.1 Профессиональные и этические требования 9

2.2.2 Кодекс этики IEEE-CS/ACM 9

2.3. Программный продукт и его артефакты 11

2.3.1 Программный проект 13

2.3.2 Процесс проектирования 13

2.3.3 Этап внедрения 14

2.4 Методы и модели программной инженерии 14

2.1 Настольная книга программных инженеров любого профиля

2.1.1 История создания Свода знаний для программных инженеров

В 1958 всемирно известный статистик Джон Тьюкей (John Tukey) впервые ввел термин *software*– программное обеспечение.

Термин «Программная инженерия» был предложен в 1968 г. на конференции посвященной «Кризису ПО», возникшего в результате появления интегральных схем и катастрофического усложнения ПО.

В 1972 году IEEE 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 обеспечения. После 7 лет напряженной работы, в 1986 году IEEE выпустило IEEE Std 1002 “Taxonomy of Software Engineering Standards”.

Наконец, 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).

С 90-х годов XX века стало присваивать разработчикам программ звание инженера, в США подобное звание появилось в 1998 году.

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 свою очередь, IEEE и ACM, 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:

1. ***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][1];
2. ***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].

Последняя версия Свода знаний была выпущена в 2013 г. (SWEBOK V3)[2]. Естественно, программный инженер любой специализации должен быть хорошо знаком со всеми областями этих знаний. В России материалы по SWEBOK можно найти по адресу: https://web.archive.org/web/20100201155827/http://swebok.sorlik.ru/ .

Важно понимать, что 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.

Модель деятельности профессиональных сообществ и знаний в области программной инженерии по SWEBOK[3] представлена на рис. 3.1.

Одной из важнейших целей SWEBOK является **именно определение и систематизация** тех аспектов деятельности, которые составляют суть профессии *инженера-программиста*.

[1] http://software-testing.ru/files/se/3-software_engineering.pdf
[2] https://www.computer.org/web/swebok/v3
[3] https://www.computer.org/cms/professional-education/images/model-of-a-profession.jpg

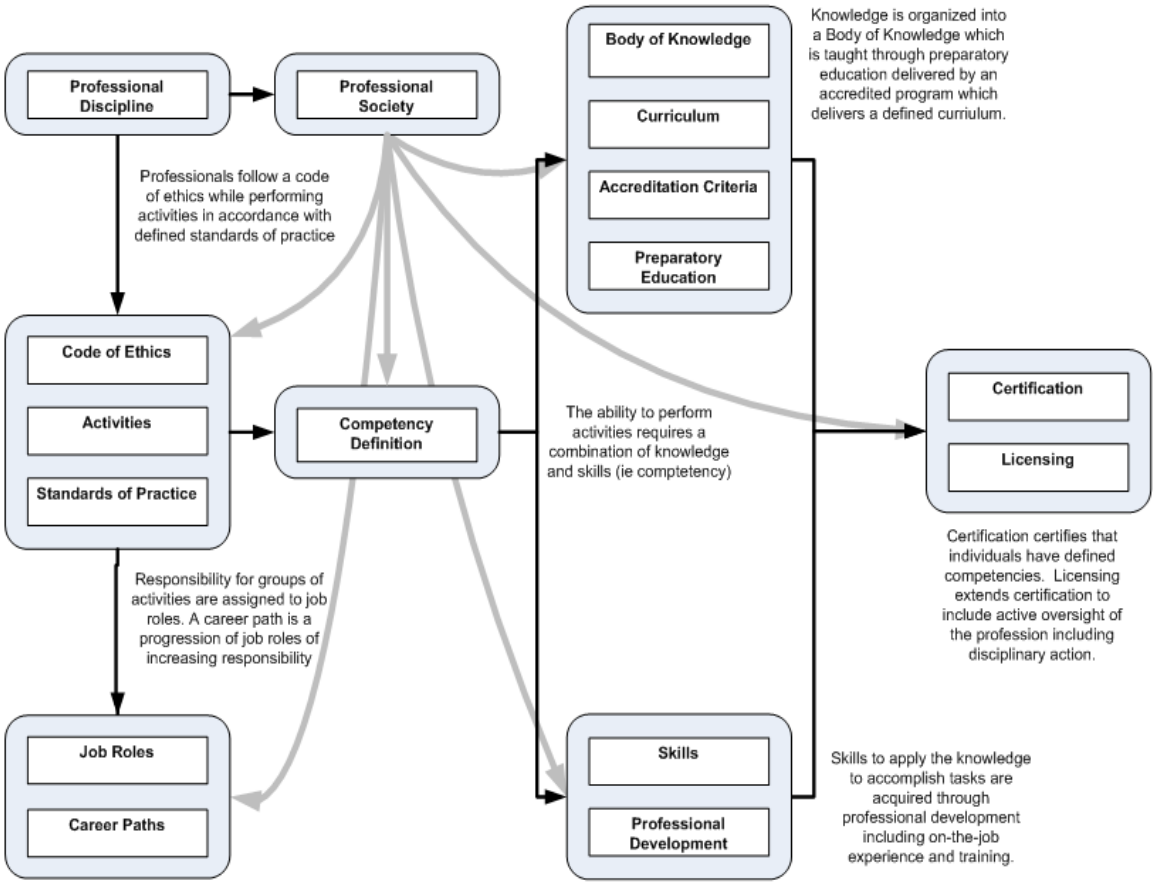

Рис. 2.1. Модель деятельности и знаний в области программной инженерии

2.1.2 Содержание Свода знаний программного инженера

Описание областей знаний в SWEBOK построено по иерархическому принципу, как результат структурной декомпозиции. Такое иерархическое построение обычно насчитывает два-три уровня детализации, принятых для идентификации тех или иных общепризнанных аспектов программной инженерии. При этом, структура декомпозиции областей знаний **детализирована только до того уровня, который необходим для понимания природы соответствующих тем и возможности нахождения источников компетенции и других справочных данных и материалов**.

В принципе, считается, что как таковой “свод знаний” по программной инженерии представлен не в обсуждаемом руководстве (SWEBOK), а в первоисточниках, как указанных в нем, так и представленных за его рамками. Это означает, что конкретизация знаний требует обращения к соответствующим стандартам и рекомендациям.

SWEBOK описывает 10 областей знаний:

1. ***Software requirements***– программные требования
2. ***Software design***– дизайн (архитектура)
3. ***Software construction***– конструирование программного обеспечения
4. ***Software testing***- тестирование

5. ***Software maintenance***– эксплуатация (поддержка) программного обеспечения
6. ***Software configuration management***– конфигурационное управление
7. ***Software engineering management***– управление в программной инженерии
8. ***Software engineering process***– процессы программной инженерии
9. ***Software engineering tools and methods*** – инструменты и методы
10. ***Software quality***– качество программного обеспечения

В дополнение к ним, SWEBOK также включает обзор смежных дисциплин, связь с которыми представлена как фундаментальная, важная и обоснованная для программной инженерии:

- ***Computer engineering***
- ***Computer science***
- ***Management***
- ***Mathematics***
- ***Project management***
- ***Quality management***
- ***Systems engineering***

Интересно рассмотреть и проанализировать отличие программной инженерии от Computer engineering (Компьютерной инженерии) и Systems Engineering (Системной инженерии). Это необходимо сделать как домашнее задание.

SWEBOK не ассоциирован с той или иной моделью (например, жизненного цикла[4]) или методом. Хотя на первый взгляд первые пять областей знаний в SWEBOK представлены в традиционной последовательной (каскадной - waterfall) модели, это не более чем следование принятой последовательности освещения соответствующих тем. Остальные области и структура декомпозиции областей представлены в алфавитном порядке.

На рис. 2.2 (а,б) представлены области программной инженерии, непосредственно связанные с практикой проектирования ПО. На рис. 2.3 описаны области знаний по **вопросам управления проектом разработки ПО и методам обеспечения успешной реализации проекта**.

[4] О жизненном цикле ПО в следующих лекциях

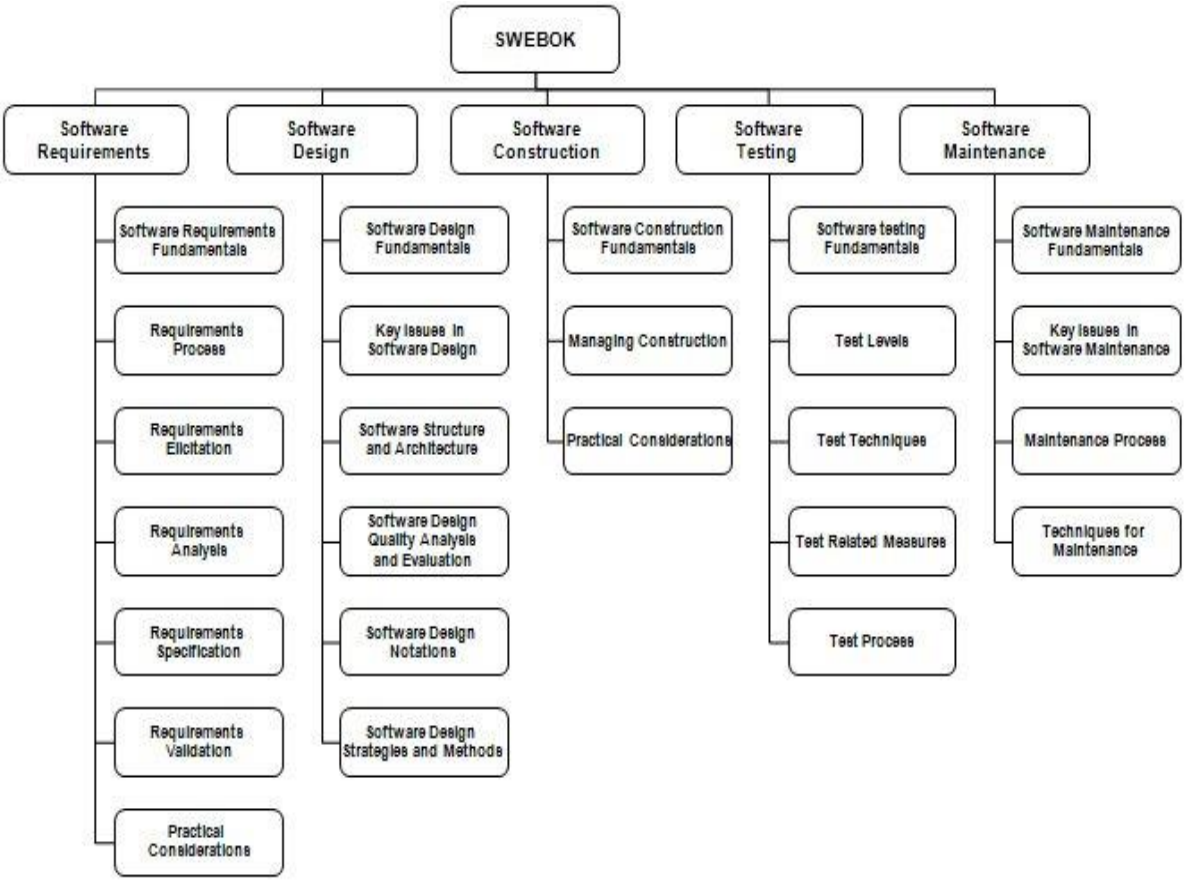

Рис. 2.2а. Содержание первых пяти областей знания SWEBOK (на английском языке)

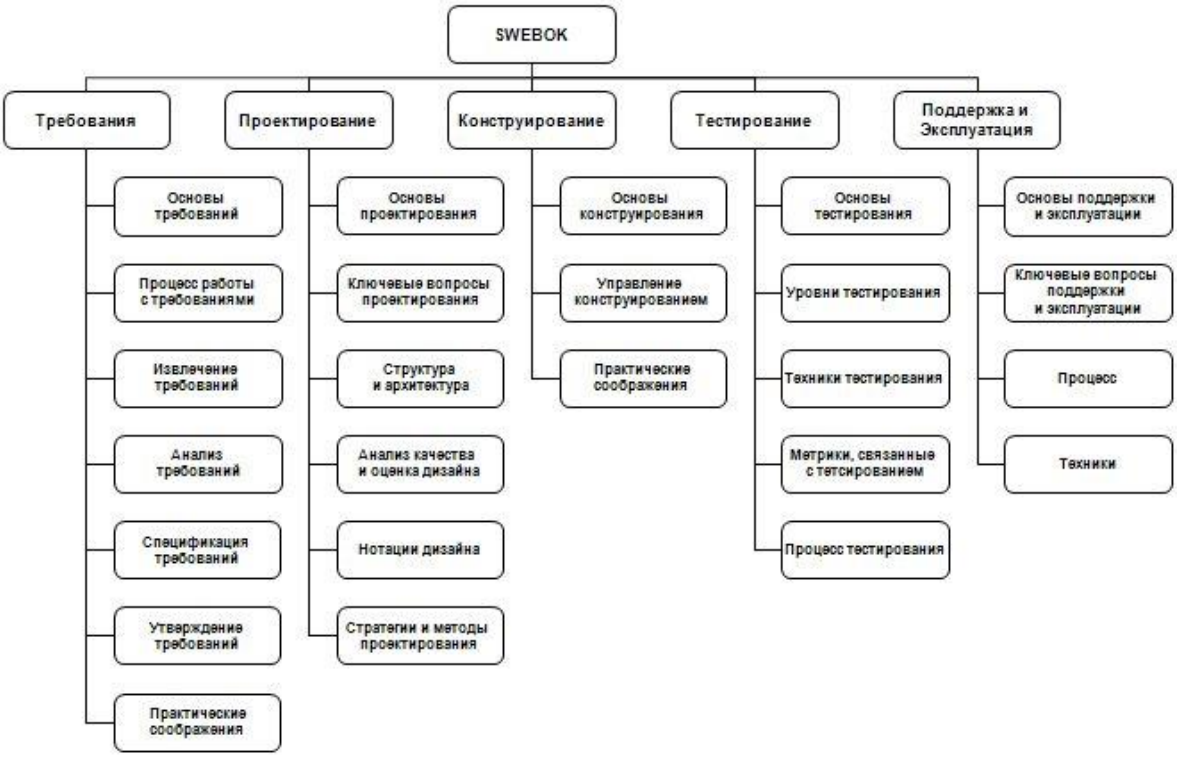

Рис. 2.2б. Содержание первых пяти областей знания SWEBOK (а- на английском языке (на русском языке)

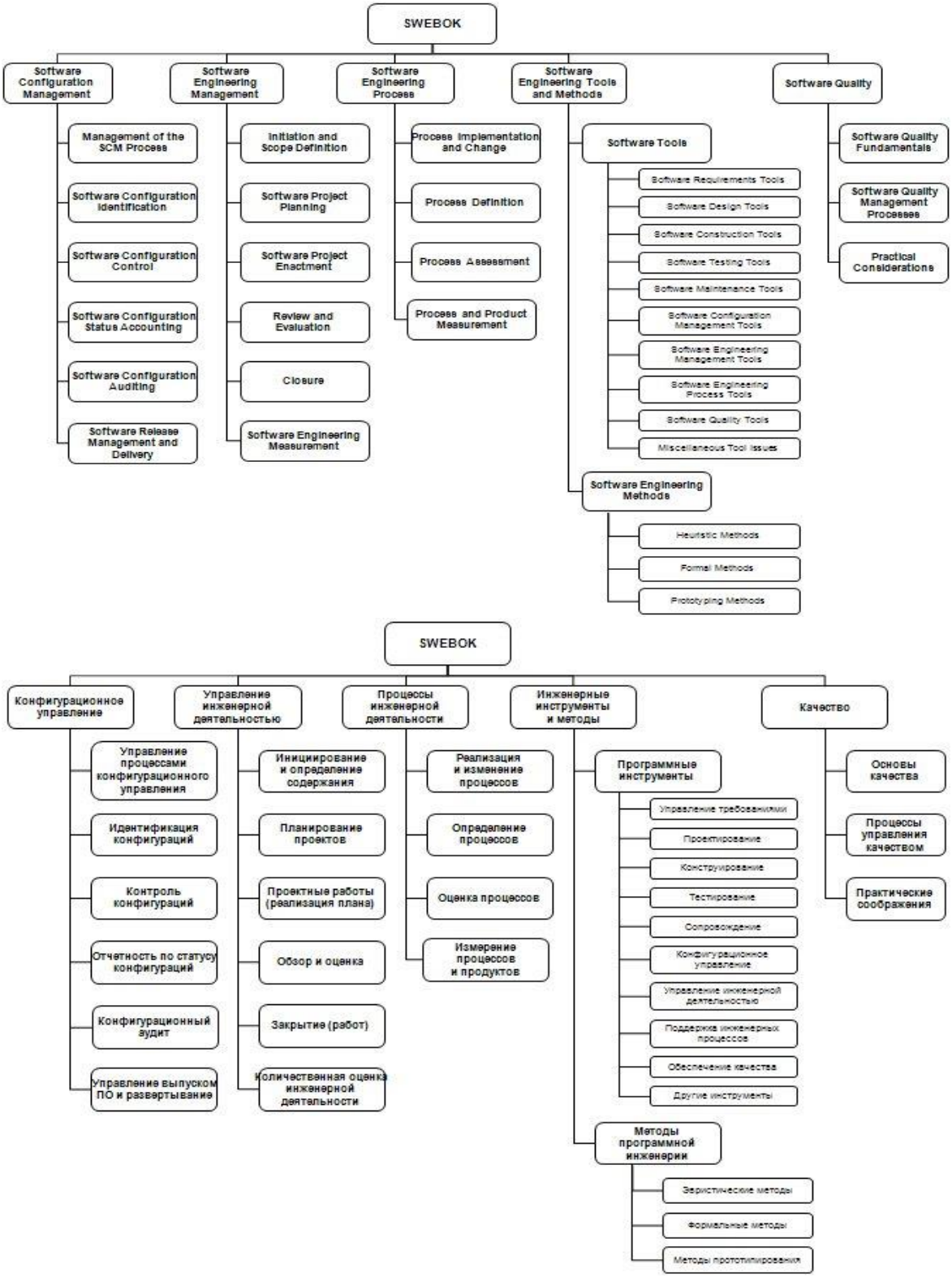

Рис.2.3. Области знаний SWEBOK, связанные с управлением проектом создания программного продукта и методам программной инженерии (а – на английском языке, б – на русском).

2.2 Знания и требования к программным инженерам в области этики

2.2.1 Профессиональные и этические требования

Развитие средств вычислительной техники, коммуникаций и программных систем (Internet, телекоммуникации, распределенные системы, IP телефония, компьютерные игры и обучающие программы, компьютерная безопасность) оказывает все большее воздействие на общество. Роль специалистов по программному обеспечению при этом все время возрастает. Они работают в определенном правовом и социальном окружении, находятся под действием, международных, национальных и местных законодательств.

Ясно, что программисты, как и специалисты других профессий, должны быть честными и порядочными людьми. Но вместе с тем, программисты не могут руководствоваться **только моральными нормами или юридическими ограничениями, т.к. они обычно бывают связаны более тонкими профессиональными обязательствами**:

- **Конфиденциальность** – программные специалисты должны уважать конфиденциальность в отношении своих работодателей или заказчиков независимо от того, подписывалось ли ими соответствующее соглашение.
- **Компетентность** – программный специалист не должен завышать свой истинный уровень компетентности и не должен сознательно браться за работу, которая этому уровню не соответствует.
- **Защита интеллектуальной собственности** – специалист должен соблюдать законодательство и принципы защиты интеллектуальной собственности при использовании чужой интеллектуальной собственности. Кроме того, он должен защищать интеллектуальную собственность работодателя и клиента. Внимание: создаваемая им интеллектуальная собственность является собственностью работодателя или клиента.
- **Злоупотребление компьютером** – программный специалист не должны злоупотреблять компьютерными ресурсами работодателя или заказчика; под злоупотреблениями понимается широкий спектр — от игр в компьютерные игрушки на рабочем месте до распространения вирусов и т.п.

Подробнее можно прочитать на сайте http://school-collection.iv-edu.ru/dlrstore/39141622-5991-11da-8314-0800200c9a66/inzenery.htm

2.2.2 Кодекс этики IEEE-CS/ACM

В разработке таких этических обязательств ведущую роль играют профессиональные сообщества. Такие общества, как

- ACM – Association for Computing Machinery - Ассоциация по вычислительной технике,

- IEEE – Institute of Electrical and Electronic Engineers – Институт инженеров по электротехнике и электронике
- CS - British Computer Society – Британское компьютерное общество

совместно разработали и опубликовали IEEE-CS/ACM Software Engineering Code of Ethics and Professional Practices – Кодекс этики и профессиональной практики программной инженерии.

Члены этих организация принимают обязательство следовать этому кодексу в момент вступления в организацию

Кодекс содержит восемь Принципов, связанных с поведением и решениями, принимаемыми профессиональными программистами, включая практиков, преподавателей, менеджеров и руководителей высшего звена

Кодекс распространяется также на студентов и «подмастерьев», изучающих данную профессию

Кодекс имеет краткую и полную версии

2.2.2.1 Кодекс этики - Преамбула

Краткая версия кодекса

- суммирует стремления кодекса на высоком уровне абстракции.
- полная версия показывает, как эти стремления отражаются на деятельности профессиональных программистов.
- без высших принципов детали кодекса станут казуистическими и нудными;
- без деталей стремления останутся возвышенными, но пустыми;
- вместе же они образуют целостный кодекс.

Программные инженеры должны добиваться, чтобы анализ, спецификация, проектирование, разработка, тестирование и сопровождение программного обеспечения стали полезной и уважаемой профессией. В соответствии с их приверженностью к процветанию, безопасности и благополучию общества, программные инженеры будут руководствоваться следующими **Восемью Принципами**

2.2.2.2 Кодекс этики: 8 принципов

1. ОБЩЕСТВО

- Программные инженеры будут действовать соответственно общественным интересам.

2. КЛИЕНТ И РАБОТОДАТЕЛЬ

- Программные инженеры будут действовать в интересах клиентов и работодателя, соответственно общественным интересам.

3. ПРОДУКТ

– Программные инженеры будут добиваться, чтобы произведенные ими продукты и их модификации соответствовал высочайшим профессиональным стандартам.

4. СУЖДЕНИЕ

– Программные инженеры будут добиваться честности и независимости в своих профессиональных суждениях

5. МЕНЕДЖМЕНТ

– Менеджеры и лидеры программных инженеров будут руководствоваться этическим подходом к руководству разработкой и сопровождением ПО, а также будут продвигать и развивать этот подход

6. ПРОФЕССИЯ

– Программные инженеры будут улучшать целостность и репутацию своей профессии соответственно с интересами общества

4. КОЛЛЕГИ

– Программные инженеры будут честными по отношению к своим коллегам и будут всячески их поддерживать

8. ЛИЧНОСТЬ

– Программные инженеры в течение всей своей жизни будут учиться практике своей профессии и будут продвигать этический подход к практике своей профессии

Полная версия кодекса: IEEE-CS/ACM Software Engineering Ethics and Professional Practices (http://club.shelek.ru/viewart.php?id=277 ,

2.3. Программный продукт и его артефакты

Ниже приводятся лишь некоторые основные понятия программной инженерии, имеющие отношения к наиболее общим («верхним») уровням проектирования ПО на разных этапах проектирования.

При разработке программного обеспечения под **программным продуктом** понимают совокупность созданного программного приложения и все его артефакты[5], появившиеся при работе над этим приложением.

[5] **Артефакт** (лат. *artefactum* от *arte* — искусственно + *factus* — сделанный) в обычном понимании — любой искусственно созданный объект, продукт человеческой деятельности.

Программный продукт – результат реализации программного проекта, обладающий заявленной функциональностью и потребительскими характеристиками.

Артефакт программного продукта – неотъемлемая часть результата и процесса выполнения программного проекта, реализованная в виде документации, программного кода (исходного или скомпилированного) или его части (например, модуля).

Примеры артефактов, соответствующих составляющих приложения приведены в табл.2.1.

Таблица 2.1 - Артефакты программного продукта

Составляющие программного продукта и процесса	**Артефакты**
Требования	Спецификация требований к программному продукту
Программная архитектура	Проектная модель
Детальная проектирование	Исходный и объектный код
Реализация	
Тестирование	Тестовые процедуры и тестовые варианты
Внедрение и сопровождение	Документация по программному обеспечению

Выделяют две категории программных продуктов.

1. Общего назначения (коробочное программное обеспечение), создаваемое для распространения на открытом рынке;
2. Программные продукты, выполненные по заказу конкретного потребителя или целевой категории. Они обладают узконаправленной функциональностью.

Правовые аспекты защиты программных продуктов – авторское право, патентная защита, закон о производственных секретах, лицензионное соглашение и контракты.

Качество программного обеспечения (программного продукта) – совокупность наиболее важных характеристик программного продукта (например, надёжность), а также характеристик самого процесса разработки (например, количество дефектов на тысячу строк кода), измеряемых количественно по тем или иным методикам, на основе которых можно сделать заключение о соответствие продукта и/или процесса заранее определённым показателям.

Прототип – программный продукт, по ряду ключевых на данный момент характеристик близкий к разрабатываемому. Прототип предназначен для демонстрации для демонстрации результатов заказчику с целью определения его мнения относительно интерфейса или части реализованной функциональности. Как правило прототипы используют для своевременной корректировки

требований к программному обеспечению к программному продукту в процессе разработки.

2.3.1 Программный проект

Проект – протяжённое во времени предприятие, направленное на создание уникальных продуктов, услуг или достижение иных результатов.

IT-проект (проект с позиций программной инженерии) – это совокупность действий, необходимая для создания артефактов программного продукта. Любой проект включает в себя взаимодействие с заказчиком, написание документации, кодирование и тестирование. А также – процесс реализации комплекса мероприятий, направленных на создание программного продукта.

Требования к программному обеспечению – документ, отражающий, что должно делать разрабатываемое программное обеспечение.

Исходный код – программный код, написанный на каком-либо языке программирования.

Объектный код – переведённый на машинный язык исходный код, ещё не подверженный особой упаковке, характерной для конкретной операционной системы.

Двоичный код – переведённый на машинный язык, понятный ЭВМ исходный код программы.

2.3.2 Процесс проектирования

Архитектура программного обеспечения – модель ПО на самом высоком уровне, формализованная теми или иными средствами

Проектирование программного обеспечения – создание модели программного обеспечения с применением тех или иных средств, языков и стандартов, уровень детализации которой находится между архитектурой программного обеспечения и потребительскими характеристиками.

Детальное проектирование - создание подробной модели разрабатываемой программной системы с применением псевдокода, блок-схем и др., достаточной для написания для написания программного кода.

Инспектирование – коллективное исследование артефактов проекта, направленное на выявление дефектов, осуществляемое, как правило, лицами (разработчиками ил их группами), не участвующими в создании инспектируемых артефактов.

Тестирование – комплекс мероприятий, направленный на выявление дефектов в готовом программном обеспечении и/или его составляющих.

Управление конфигурациями программного обеспечения – поддержание соответствия версий всех артефактов создаваемого программного продукта в процессе разработки.

2.3.3 Этап внедрения

Сопровождение программного обеспечения – комплекс мероприятий, направленный на выявление дефектов в готовом программном обеспечении и/или его составляющих.

2.4 Методы и модели программной инженерии

Метод программной инженерии — это структурный подход к созданию ПО, который способствует производству высококачественного продукта эффективным в экономическом аспекте способом. В этом определении есть две основные составляющие: (а) создание высококачественного продукта и (б) экономически эффективным способом.

Иными словами, метод – это то, что обеспечивает решение основной задачи программной инженерии: создание качественного продукта при заданных ресурсах времени, бюджета, оборудования, людей.

Начиная с 70-х годов создано достаточно много методов разработки ПО. Наиболее известны:

- Метод структурного анализа и проектирования Том ДеМарко (1978),
- Метод сущность-связь проектирования информационных систем Чен (1976)
- Метод объектно-ориентированного анализа Буч (1994), Рамбо (1991).

Метод программной индустрии основан на идее создания моделей ПО с поэтапным преобразованием этих моделей в программу – окончательную модель решаемой задачи.

Так, на **этапе спецификаций создается модель** – описание требований, которая далее преобразуется в модель проекта ПО, проект – в программный код. При этом важно, чтобы модели метода представлялись графически с помощью некоторого языка представления моделей.

Методы должны включать в себя следующие компоненты:

- Описание моделей системы и нотация, используемая для описания этих моделей (например, объектные модели, конечно-автоматные модели и т.д.)
- Правила и ограничения, которые надо выполнять при разработке моделей (например, каждый объект должен иметь одинаковое имя)
- Рекомендации — **эвристики**[6], характеризующие хорошие приемы проектирования в данном методе (скажем, рекомендация о том, что ни у **одного объекта не должно быть больше семи подобъектов**)
- Руководство по применению метода — описание последовательности работ (действий), которые надо выполнить для построения моделей (все

[6] **Эвристика** (*от древнегреческого ευρίσκω – «отыскиваю», «открываю»*) – совокупность логических приемов, методов и правил, облегчающих и упрощающих решение познавательных, конструктивных, практических задач. Эвристика – это момент открытия нового, а также методы, которые используются в процессе этого открытия. Эвристикой еще называют науку, которая имеет дело с изучением творческой деятельности. В педагогике под этой категорией подразумевается метод обучения.

атрибуты должны быть задокументированы до определения операций, связанных с этим объектом)

Нет идеальных методов, все они применимы только для тех или иных случаев.

Нет абсолютных методов – применяемые на практике методы могут включать элементы различных подходов. **Выбор метода составляет задачу специалиста по программной инженерии.**

Модель в программной инженерии – отображение в той или иной нотации (форме), понятной квалифицированному специалисту, задачи и результата того или иного этапа процесса разработки программного проекта. Например, различают следующие модели:

- Модель прецедентов (требований);
- Модель классов;
- Модель сущность-связь.