

# Программная инженерия

## Процесс программной инженерии (Software Engineering Process)

Глава базируется на IEEE Guide to the Software Engineering Body of Knowledge - SWEBOK®, 2004. Содержит перевод описания области знаний SWEBOK® "Software Engineering Process", с комментариями и замечаниями.

"Основы программной инженерии" разработаны на базе IEEE Guide to SWEBOK® 2004 в соответствии с IEEE SWEBOK 2004 Copyright and Reprint Permissions: "This document may be copied, in whole or in part, in any form or by any means, as is, or with alterations provided that (1) alterations are clearly marked as alterations and (2) this copyright notice is included unmodified in any copy."

Русский перевод SWEBOK 2004 с замечаниями и комментариями подготовлены [Сергеем Орликом](#) при участии [Юрия Булуя](#). Дополнительные главы написаны [Сергеем Орликом](#). Текст расширений SWEBOK отмечен цветом, отличным от перевода оригинального текста.

"Основы программной инженерии" Copyright © 2004-2010 [Сергей Орлик](#). Все права защищены.  
[SWEBOK](#) Copyright © 2004 by The Institute of Electrical and Electronics Engineers, Inc. All rights reserved.

Официальный сайт "Основ программной инженерии" (по SWEBOK) - <http://swbok.sorlik.ru>

## Программная инженерия

### Процесс программной инженерии (Software Engineering Process)

Программная инженерия .....	2
Процесс программной инженерии (Software Engineering Process).....	2
1. Реализация и изменение процесса (Process Implementation and Change).....	3
1.1 Инфраструктура процесса (Process Infrastructure).....	4
1.2 Цикл управления программным процессом (Software Process Management Cycle).....	5
1.3 Модели реализации и изменения процесса (Models for Process Implementation and Change) .....	5
1.4 Практические соображения (Practical Considerations).....	6
2. Определение процесса (Process Definition).....	6
2.1 Модели жизненного цикла программного обеспечения (Software Life Cycle Models) .....	6
2.2 Процессы жизненного цикла программного обеспечения (Software Life Cycle Processes) ..	7
2.3 Нотации определения процесса (Notations for Process Definitions) .....	7
2.4 Адаптация процесса (Process Adaptation).....	8
2.5 Автоматизация (Automation) .....	8
3. Оценка процесса (Process Assessment) .....	8
3.1 Модели оценки процесса (Process Assessment Models) .....	8
3.2 Методы оценки процесса (Process Assessment Methods).....	9
4. Измерения в отношении процессов и продуктов (Process and Product Measurement).....	10
4.1 Измерения в отношении процессов (Process Measurement) .....	10
4.2* Измерения в отношении программных продуктов (Software Product Measurement).....	11
4.3 Качество результатов измерений (Quality Of Measurement Results) .....	12
4.4 Информационные модели (Software Information Models).....	13
4.5 Техники количественной оценки процессов (Process Measurement Techniques).....	13

Область знаний “Процесс программной инженерии” (Software Engineering Process) может быть рассмотрена на двух уровнях. Первый уровень содержит техническую и управленческую деятельность на протяжении процессов жизненного цикла программного обеспечения, включающих приобретение, разработку, сопровождение и вывод из эксплуатации программных систем. Второй уровень – “мета-уровень”, связанный с определением, реализацией, оценкой, измерением, управлением, изменением и совершенствованием самих процессов жизненного цикла программного обеспечения. Первый уровень освещен в других областях знаний SWEBOK. Второй уровень рассматривается в данной области знаний.

Термин “процесс программной инженерии” (software engineering process) может интерпретироваться по-разному и это, соответственно, может приводить к определенной путанице.

- С одной стороны, учитывая специфику оригинального термина в английском языке, где (с точки зрения грамматики) может существовать термин *the software engineering process*, он будет подразумевать единственно правильный способ выполнения задач (performing tasks) программной инженерии. Такое предположение заведомо отбрасывается SWEBOK, так как “единственно правильного” процесса быть не может. Такие стандарты, как IEEE/ISO/ГОСТ 12207 говорят о *процессах* (во множественном числе - *processes*), подразумевая что программная инженерия содержит множество процессов, например, процесс разработки (Development Process) и процесс конфигурационного управления (Configuration Management Process).
- Вторая интерпретация связана с общим (general) обсуждением процессов, связанных с программной инженерией. Данная точка зрения отражена в названии этой области знаний и является одной из наиболее часто подразумеваемых при использовании термина “процесс программной инженерии”.
- Наконец, третье понимание данного термина может означать реальный набор действий, предпринимаемых в данной организации и рассматриваемый как единый процесс на

уровне организации. Такой подход также рассматривается в данной области знаний (та или иная интерпретация обычно зависит от контекста обсуждения).

Данная область знаний связана со всеми элементами управления процессами жизненного цикла программного обеспечения, в которых процедурные (управленческие) или технологические изменения применяются к совершенствованию процесса или продукта.

Процесс программной инженерии касается не только крупных организаций. Более того, связанные с данным процессом действия могут и должны применяться небольшими организациями, командами и отдельными специалистами.

Цель управления процессами программной инженерии состоит в реализации новых и лучших процессов в реальной практике конкретных специалистов, проектов или организации (отдельных ее групп подразделений или организации, в целом).

Данная область знаний не адресуется напрямую вопросам управления персоналом (human resources management, HRM). Эти темы исследуются, например, в *People CMM* (People Capability Maturity Model) и процессах *системной инженерии* (см. стандарты ISO 15288 “Systems Engineering - System Life Cycle Process” и IEEE 1220 “Standard for the Application and Management of the Systems Engineering Process”).

Также, необходимо понимать, что многие процессы программной инженерии порождаются и тесно связаны с другими дисциплинами, например, управлением (management), хотя иногда эти процессы и называют по-другому в контексте этих дисциплин.

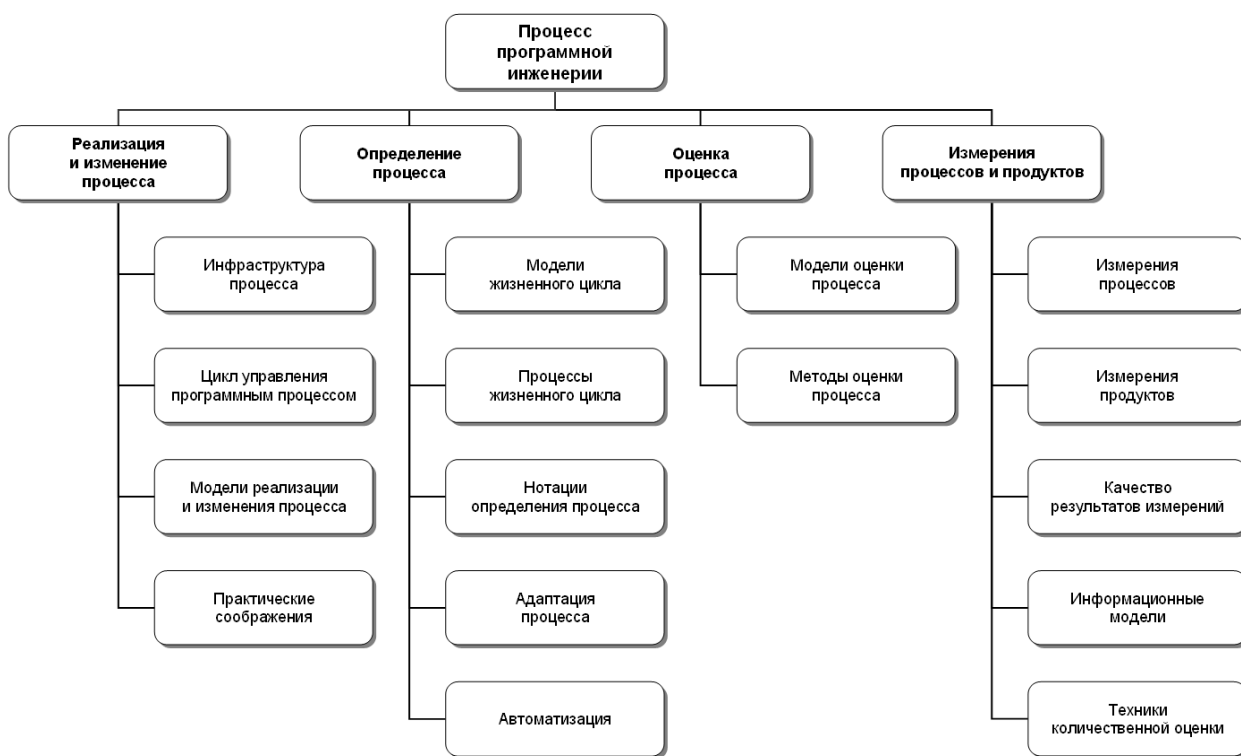


Рисунок 1. Область знаний “Процесс программной инженерии” [SWEBOK, 2004, с.9-2, рис. 1]

#### 1. Реализация и изменение процесса (Process Implementation and Change)

Данная секция фокусируется на организационных изменениях. Она описывает инфраструктуру, действия, модели и практические соображения по реализации процесса и его изменении.

Ниже рассматривается ситуация, в которой те или иные процессы реализуются впервые (например, процесс проведения инспекций в проекте или охват полного жизненного цикла программного обеспечения) и где изменяются уже существующие (используемые) процессы. Речь пойдет о том, что называют *эволюцией процесса* (*process evolution*). Существующие практики

рассматриваются в контексте часто необходимой модификации. Если требуемые модификации достаточно обширны, это приводит и к необходимости изменения организационной культуры.

#### 1.1 Инфраструктура процесса (*Process Infrastructure*)

Эта тема охватывает знания, связанные с инфраструктурой процесса программной инженерии и, в большой степени, базируется на стандартах IEEE/ISO/ГОСТ 12207 “Standard for Information Technology - Software Life Cycle Processes” и ISO 15504 “Information Technology - Software Process Assessment” (известен также как SPICE - Software Process Improvement and Capability dEtermination).

Для внедрения процессов жизненного цикла необходимо обладать соответствующей инфраструктурой, подразумевая, что ресурсы (компетентный персонал, инструменты, финансирование) – доступны, а ответственность – распределена <по членам проектной команды и/или организационной единицы, в терминах структуры компании или организации, например, отдела или группы>. Выполнение этих задач является хорошим индикатором того, что менеджмент <управленческий персонал проекта/организации> реально прилагает усилия по поддержке процесса программной инженерии. Как следствие таких усилий могут создаваться различные комитеты и другие специализированные организационные структуры и органы, в общем случае называемые *steering committee* – “управляющий комитет”, обладающий наблюдательными функциями в отношении усилий, направленных на мониторинг, контроль и выработку рекомендаций по поддержке и улучшению процесса программной инженерии. На основе таких функций будем в дальнейшем использовать термин “*наблюдательный орган*”, подчеркивая реальные задачи такой комиссии и возможность как формальной, так и неформальной его организации.

Наблюдательный орган является основой процессной инфраструктуры в проектной команде, подразделении или организации, в целом. Обычно, выделяют два типа инфраструктуры, применяемые на практике *Software Engineering Process Group* (SEPG, обычно, в русском языке для такой структуры используется приведенная англоязычная аббревиатура) и *Experience Factory* (EF, “фабрика опыта”).

##### 1.1.1 Software Engineering Process Group (SEPG)

SEPG создается как центральный орган, принимающий на себя работу по *process improvement - совершенствованию процесса(-ов)*. SEPG берет на себя ответственность по множеству вопросов, связанных с этой задачей в терминах инициирования <улучшений> и поддержки <существующего процесса и его постоянного совершенствования>.

Часто SEPG формируется из нескольких ведущих членов проектной команды (если, SEPG создается в рамках проекта) или на уровне подразделения или всей организации. При этом, в большинстве случаев, SEPG не включает “освобожденных” специалистов и, таким образом, ее члены всегда находятся в контексте реальных проблем, с которыми сталкиваются выполняя свои “основные” обязанности. Исключение, обычно, составляют SEPG, формируемые для достижения определенных организационных целей - приведения процессов в соответствие тем или иным требованиям и, в частности, для достижения того или иного уровня зрелости CMMI, обеспечения качества в рамках ISO или SixSigma и т.п. В этих случаях SEPG обычно возглавляется выделенным экспертом (или группой) в области постановки и совершенствования процессов.

##### 1.1.2 Experience Factory (EF)

Концепция “фабрики опыта” отделяет проектную организацию (например, организационную структуру, отвечающую за разработку программного обеспечения – ИТ-подразделение, группу разработки или проектную команду) от организации, отвечающей за улучшение процесса. Проектная организация, в этом случае, фокусируется на разработке и сопровождении программного обеспечения, а EF – занята совершенствованием процесса программной инженерии.

Основной задачей EF является *институализация* (внедрение в повседневную практику) коллективного опыта и полученных уроков в масштабах организации на основе разработки, обновления и внедрения в проектную организацию “пакетов опыта” – *experience packages*

(например, руководств, моделей, курсов обучения и т.п.), <типовых> “активов процесса” – *process assets*. Проектная организация предлагает на рассмотрение EF свои продукты, планы, использовавшиеся при разработке, а также данные, собранные в процессе разработки и эксплуатации.

Сложно провести четкую грань между SEPG и EF. Скорее, можно говорить о создании SEPG в форме “фабрики опыта” в крупных ИТ-подразделениях, например, международных компаний, или достаточно крупных организациях, основной деятельностью которых является создание программного обеспечения. В этом случае SEPG проводит пилотное внедрение усовершенствованных или новых процессов в рамках одного или нескольких выбранных проектов и, затем, распространяет этот опыт во всей организации. Так или иначе, отдача от SEPG/EF обычно заметна в *проектно-ориентированных* или *проектных организациях*, чья деятельность построена в форме управления портфелем проектов (более подробную информацию о проектно-ориентированных организациях можно, например, найти в PMI PMBOK и других материалах Project Management Institute). В общем случае, говоря об инфраструктуре процессов, обычно используют именно термин SEPG для обоих типов организации команд, фокусирующихся на процессе разработки программных систем.

### 1.2 Цикл управления программным процессом (*Software Process Management Cycle*)

Управление процессами в области программного обеспечения состоит из четырех действий, представленных в рамках итеративного цикла. Это позволяет получать и анализировать отклики на постоянной основе и, <более оперативно> совершенствовать процесс. Вот эти четыре действия, предлагаемые SWEBOK:

- *Establish Process Infrastructure* – *создание инфраструктуры процесса*. Задачи – обеспечить согласие и поддержку заинтересованных лиц (в первую очередь, менеджмента) в работах по реализации и изменению процесса; получить возможность развернуть соответствующую инфраструктуру процесса, выделив необходимые ресурсы и обеспечив распределение обязанностей (ответственности).
- *Planning* – *планирование*. Задача (цель) – понять (*сформулировать*) текущие бизнес-цели и потребности в процессе, необходимые отдельным специалистам, проекту и/или организации, в целом; идентифицировать сильные и слабые стороны (*см. концепцию SWOT-анализа в различных источниках*) <существующего процесса и планируемых на данной итерации нововведений и/или изменений> и разработать план реализации и изменения процесса.
- *Process Implementation and Change* – *реализация и изменение процесса*. Задача (цель) – выполнение разработанного плана по внедрению нового и/или модифицированного процесса (включая, например, если это необходимо, развертывание новых инструментов или проведение тренингов). В результате заданный процесс должен быть внедрен в практику.
- *Process Evaluation* – *оценка процесса*. Задача (цель) – понять, насколько хорошо процесс реализован, получены или нет ожидаемые преимущества от его внедрения. Результат анализа становится “входом” для следующей итерации.

### 1.3 Модели реализации и изменения процесса (*Models for Process Implementation and Change*)

Существует две распространенные модели внедрения процесса – Quality Improvement Paradigm – QIP (Software Engineering Laboratory, Software Process Improvement Guidebook, NASA/GSFC, Technical Report SEL-95-102, April 1996, <http://sel.gsfc.nasa.gov/website/documents/online-doc/95-102.pdf>) и разработанная в Институте программной инженерии Университета Карнеги-Меллон SEI CMU модель IDEAL (Initiating – Diagnosing – Establishing – Acting – Learning). Во всех случаях оценка может проводиться по качественным и/или количественным показателям.

На сегодняшний день наиболее проработанными и распространенными стандартами оценки и совершенствования процесса программной инженерии являются CMMI (де факто стандарт) и SCAMPI (разработанная в SEI CMU стандартная методика оценки совершенствования процессов – Standard CMMI Appraisal Method for Process Improvement), а также в ISO/IEC 15504 (де юро



стандарт), также известном как SPICE (Software Process Improvement and Capability Determination) и разработанным для аттестации зрелости процессов.

#### *1.4 Практические соображения (Practical Considerations)*

Реализация и изменение процесса является составной частью организационных изменений. В большинстве успешных случаев усилия, направленные на организационные изменения рассматриваются как самостоятельный проект со своими (соответствующими) правами, планами, ресурсами и т.п.

Обычно составляются соответствующие руководства (guidelines) по реализации и изменению процесса, включая разработку плана действий (action plan), проводятся тренинги, согласуется поддержка менеджмента (желательно, высшего управленческого звена), отбираются пилотные проекты, в которых впервые будут задействованы соответствующие процессы и инструменты и т.п. Такие рекомендации можно найти во многих источниках, в том числе, и в указанных в оригинальной версии SWEBOK. Также, можно найти множество отчетов и исследований по факторам успеха, значимым для внедрения и изменения процесса ([например, многие из таких исследований связаны с моделью CMMI и представлены на сайте SEI CMU <http://sei.cmu.edu/>](#)).

SWEBOK также отмечает роль “агентов” изменений, как лиц, часто создающих предпосылки, инициирующих изменения, а также специалистов, постоянно реализующих изменения в своей практике. Естественно, что реализация и изменение процесса может рассматриваться как консалтинг. Он может быть внутренний (например, проводимый силами специалистов SEPG) или внешний (с привлечением экспертов из других подразделений и организаций, часто специализирующихся в данной области так же, как мы видим консультантов и внешних управляющих в области проектного менеджмента). Практика показывает, что достаточно успешным является подход, предполагающий совместную работу внешних и внутренних консультантов SEPG, так как в этом случае легче отойти от сложившихся внутри организации шаблонов восприятия и обеспечить свежий взгляд на возможности и потенциальную отдачу в совершенствовании процесса, конечно, с учетом опыта вовлеченных в эти работы специалистов.

Кроме того, можно увидеть организационные изменения в контексте внедрения тех или иных технологий (в SWEBOK используется термин technology transfer). При этом, эти технологии могут касаться как непосредственно самого программного обеспечения, так и связаны с самим процессом ([например, технологии моделирования](#)).

Существует два распространенных подхода к оценке реализации и изменения процесса. Они состоят в оценке самого процесса и в оценке результатов процесса (process outcomes), соответственно.

## **2. Определение процесса (Process Definition)**

Определение процесса может быть процедурой, рекомендацией или стандартом. Процессы жизненного цикла программного обеспечения четко определяются по разным причинам, в частности, с целью повышения качества получаемого продукта, улучшения коммуникаций и улучшения понимания различных аспектов программной инженерии отдельными специалистами, поддержки совершенствования процессов, поддержки управления процессами, обеспечения автоматизации процессов и т.п. Используемые типы описаний процессов, часто, зависят (как минимум, частично) от целей определения процессов.

Также необходимо отметить, что проектный и организационный контексты помогают определить наиболее подходящие определения процессов. Важными факторами при определении процесса являются природа работ (например, разработка или сопровождение), прикладная область (application domain), модель жизненного цикла и зрелость самой организации.

### *2.1 Модели жизненного цикла программного обеспечения (Software Life Cycle Models)*

Модели жизненного цикла задают высокоуровневое определение фаз (стадий) разработки программного обеспечения. Их целью *не* является предоставление детального определения, но концентрируется на ключевых работах и их взаимосвязях. Примерами таких моделей\* являются водопадная (каскадная - waterfall), модель прототипирования, эволюционной разработки,

инкрементальная/итеративная, спиральная и т.п. Существуют различные сравнения и критерии выбора моделей, ссылки на некоторые из которых, в частности, даны в оригинальной версии SWEBOK.

\* базовые модели жизненного цикла рассматриваются далее в отдельной главе.

## 2.2 Процессы жизненного цикла программного обеспечения (Software Life Cycle Processes)

Определения процессов жизненного цикла обычно являются более детальными, чем модели. Однако, определения процессов не описывают порядка их выполнения во времени (за это как раз и отвечают модели, *прим. автора*). Это означает, что, в принципе, процессы жизненного цикла программного обеспечения могут быть “выстроены” (во времени) соответственно любой модели жизненного цикла. Основным источником знаний по процессам является стандарт *IEEE/ISO/ГОСТ 12207 “Information Technology – Software Lifecycle Processes”* (основные элементы структуры этого стандарта рассматривается за рамками перевода и комментариев SWEBOK в самостоятельной главе, посвященной жизненному циклу).

В рамках данных понятий жизненного цикла - “модель” и “процессы”, возможно говорить, что *совокупность модели, процессов и практик определяет метод/методологию <поддержки жизненного цикла>*.

Стандарт *IEEE 1074 “Standard for Developing Software Life Cycle Processes”* предоставляет список процессов и действий по разработке и сопровождению программного обеспечения, а также *список действий по поддержке самого жизненного цикла*, который может быть отображен на процессы и организован таким же образом, как и любая модель жизненного цикла. Кроме того, этот стандарт идентифицирует и связывает другие стандарты IEEE с действиями по поддержке процессов жизненного цикла. В принципе, стандарт IEEE 1074 может быть использован для построения процессов, соответствующих любой модели жизненного цикла.

SWEBOK отмечает два стандарта, связанных с процессами сопровождения программного обеспечения – IEEE 1219 “Standard for Software Maintenance” и ISO 14764 “Standard for Software Engineering -Software Maintenance” (см. область знаний SWEBOK “Сопровождение программного обеспечения”).

Другие важные стандарты, предоставляющие определение процессов, включают:

- IEEE 1540: Standard for Software Risk Management – управление рисками программного обеспечения
- IEEE 1517: Standard for Software Reuse Processes – процессы повторного использования программного обеспечения
- ISO/IEC 15939: Standard for Software Measurement Process – процесс измерений в области программного обеспечения

В ряде ситуаций процессы программной инженерии определяться принимая во внимание организационные процессы управления качеством. ISO 9001 формулирует требования к процессам управления качеством, а ISO 9003 интерпретирует эти требования в отношении организаций, занимающихся разработкой программного обеспечения (ISO/IEC 90003:2004, Software and Systems Engineering - Guidelines for the Application of ISO9001:2000 to Computer Software).

Некоторые процессы жизненного цикла придают особое значение быстрому вводу в эксплуатацию (rapid delivery) программных систем и глубокой вовлеченности пользователей <в процесс разработки>. Такие процессы называют *agile*-методами – быстрыми, живыми, подвижными. К ним относится, например, экстремальное программирование – eXtreme Programming (XP, см. работы Кента Бека – Kent Beck). Отличительной особенностью этих методов является гибкость в вопросах планирования, когда план проекта активно корректируется по мере продвижения к цели проекта. Другие процессы уделяют специальное внимание принятию решений на основе оценки рисков (см. работы Барри Бозма – Barry W. Boehm).

## 2.3 Нотации определения процесса (Notations for Process Definitions)

Процессы могут определяться на различных уровнях абстракции. В свою очередь, могут быть определены и различные элементы процессов – действия, продукты (артефакты) и ресурсы. При этом, могут использоваться детальные фреймворки, структурирующие типы информации, требуемой для определения процессов.

Существует ряд нотаций, используемых для определения процессов. Ключевое отличие между ними заключается в типах информации, которая определяется, контролируется и используется тем или иным фреймворком. Инженеры должны иметь представление о следующих подходах: *диаграммах потоков данных* (data flow diagrams), в терминах целей процессов и получаемых на их выходе результатов (outcomes) (см. стандарт ISO 15504 “Information Technology - Software Process Assessment” - “SPICE”), как наборе процессов и их декомпозиции в работы и задачи, определенный на естественном языке (см. стандарт IEEE/ISO/ГОСТ 12207), *диаграммах переходов и состояний* (statechart), SADT, IDEF0 и многих других.

Хотя SWEBOK приводит расширенный список диаграмм/нотаций, вероятно, в силу своей “консервативности”, не упоминает, например, activity-диаграммы UML, хотя они могут использоваться в практике для описания бизнес-процессов, в частности, и для описания процессов программной инженерии. Ряд нотаций разработан и используется в рамках конкретных (частных) фреймворков/методологий, например, RUP. Кроме того, существует успешный опыт по использованию достаточно нотации BPMN – *Business Process Management Notation* для описания процессов программной инженерии. Спецификация BPMN определяет графическое представление бизнес-процессов в форме *диаграмм бизнес-процессов* – *Business Process Diagram (BPD)*. Первый стандарт BPMN был выпущен 3 мая 2004 года консорциумом *The Business Process Management Initiative – BPMI.org* (<http://www.bpmi.org>). Предоставляя развитые выразительные средства для определения процессов как комплекса взаимосвязанных действий, событий и артефактов, сгруппированных по участникам, BPMN позволяет достаточно легко сформировать в рамках одной диаграммы BPD цельный взгляд на процессы.

#### 2.4 Адаптация процесса (Process Adaptation)

Важно отметить, что предопределенные процессы, даже стандартизированные, должны адаптироваться в соответствии с локальными (конкретными) потребностями, например, организационным контекстом, размером проекта, регулирующих требованиях, промышленных практиках и корпоративной культурой. Ряд стандартов, в первую очередь, IEEE/ISO/ГОСТ 12207 и ISO 15504, содержат механизмы и рекомендации по процессу адаптации и его совершенствованию.

#### 2.5 Автоматизация (Automation)

Автоматизированные средства либо поддерживают сами работы по определению процессов (например, позволяя описывать процессы с использованием тех или иных диаграмм и нотаций) и/или предоставляют соответствующие руководства по определению процессов (например, RUP, EUP или MSF). В случаях, когда проводится процесс анализа, некоторые инструменты обеспечивают различные формы симуляции моделируемых (определяемых) процессов.

### 3. Оценка процесса (Process Assessment)

Оценка процесса (process assessment) проводится с использованием соответствующих *моделей оценки* (assessment models) и *методов оценки* (assessment methods). Во многих случаях вместо термина “assessment” используется термин “appraisal” (*подразумеваемая сама процедура оценки, например, CMMI Appraisal*). В свою очередь, термин “appraisal” заменяют на “capability evaluation”, когда говорят об *оценке способностей/потенциальных возможностей*, например, с целью заключения контракта/договора подряда на проведение соответствующих работ.

Оценка процесса(-ов) может проводиться как неформально, подразумевая внутрикорпоративные инициативы по повышению качества, и формально (то есть с получением аттестационного документа), в том числе, с привлечением внешних специалистов по оценке и, часто, с целью подтверждения соответствующего качества/уровня зрелости процессов.

#### 3.1 Модели оценки процесса (Process Assessment Models)



Модель оценки задает, что именно признается лучшими практиками оценки. Эти практики могут касаться только “технических” работ программной инженерии ([например, проектирования или кодирования](#)), а могут иметь отношение и к вопросам управления, системной инженерии, управления персоналом и т.п.

Стандарт ISO/IEC 15504 (SPICE) определяет типовую (exemplar) модель оценки и требования соответствия к другим моделям. В каждом конкретном случае используется та или иная существующая модель – CMM-SW, CMMI, Bootstrap\*. Также имеются и другие модели, например, ISO 9000-3 (теперь именуемый как ISO 90003) “Software and Systems Engineering - Guidelines for the Application of ISO9001:2000 to Computer Software”, являющаяся приложением общей модели качества ISO 9001 “Quality Management Systems - Requirements” к программной инженерии. Кроме того, существуют частные модели, охватывающие, например, только вопросы документирования, проектирования и т.п.

\* В 1990 году стартовала европейская инициатива European Strategic Program for Information Technology (ESPRIT), целью которой было внедрение современных программных технологий в Европе. В основу инициативы легли работы Уотса Хампфри (Watts S. Humphrey) – идеолога CMMI, PSP (People Software Process) и многих других современных концепций программной инженерии как дисциплины. Результат этой инициативы был назван “Bootstrap” – “самонастройка”. В то время, как модель CMM – Capability Maturity Model (в частности, CMM-SW – CMM for Software), а потом и CMMI, разрабатывались с учетом потребностей крупных государственных структур США (в первую очередь, министерства обороны) и их подрядчиков, модель Bootstrap изначально была ориентирована на малые и средние коммерческие компании, с определенным акцентом на индивидуальные практики.

Также и в системной инженерии существуют модели зрелости, применимые в отношении программного обеспечения, когда программы являются частью системы.

SWEBOK отмечает, что ряд моделей применим к небольшим организациям.

Существуют две основных архитектуры моделей оценки: *непрерывная* (continuous) и *этапная* (staged). Отличия между ними заключаются во взгляде на порядок оценки процессов. Выбор соответствующей архитектуры и модели оценки в конкретной организации должен базироваться на ее целях и потребностях ([например, необходимости совершенствования тех или иных процессных областей или официального подтверждения внешним ассессором достижения организацией четвертого уровня зрелости всего процесса программной инженерии по CMMI](#)).

### 3.2 Методы оценки процесса (Process Assessment Methods)

Для надлежащего проведения оценки соответствующие методы <оценки> позволяют получить количественные параметры, характеризующие возможности оцениваемого процесса (или зрелости организации, в целом).

Например, метод CBA-IPi (CMM-Based Appraisals for Internal Process Improvement) фокусируется на совершенствовании процесса внутри организации, а метод SCE (Software Capability Evaluation) касается процессов у подрядчиков\*. Требования в обоих типах методов отражают те лучшие практики оценки, которые описаны в стандарте ISO 15504. Эти методы были разработаны для модели CMM-SW. С выходом CMMI (интегрированной модели, объединяющей различные модели CMM), соответственно, получило развитие новое семейство методов - SCAMPI (Standard CMMI Appraisal Method for Process Improvement). Деятельность, выполняемая в процессе оценки, распределение усилий по соответствующим работам и общая атмосфера оценки ([касающаяся, в первую очередь, степени формализации, сопутствующей оценке](#)) могут серьезно отличаться, в зависимости от того, направлена ли оценка на совершенствование процессов или проводится в контексте контракта/договора подряда.

\* SEI разделяет общее понятие appraisal на assessment и evaluation. Assessment – внутренняя деятельность в организации, направленная на оценку и совершенствование собственного процесса в рамках всей организации. Evaluation подразумевает аудит и мониторинг процессов поставщика (подрядчика, исполнителя) со стороны заказчика, в первую очередь, в процессе самого выполнения работ, т.е. уже после заключения контракта/договора подряда. CBA-IPi относится к общей категории методов Software Process Assessment (SPA) как части работ по

совершенствованию процессов – Software Process Improvements (SPI). SPI как деятельность по совершенствованию процесса(-ов) сегодня считается достаточно общим термином, используемым за рамками CMM/CMMI, например, в контексте таких фреймворков, как ISO 15504 или TQM (Total Quality Management).

Существует определенная критика моделей, методов (да и самой идеи) оценки. Такая критика, обычно, основана на эмпирической природе оценки. Однако, по прошествии определенного периода времени, после публикации таких критических материалов, опыт и практика индустрии сформировали достаточно четкие доказательства (в том числе, собрав необходимые статистические данные – см. отчеты SEI CMU по результатам внедрения и использования CMMI) обоснованности современных принципов, моделей и методов оценки.

#### 4. Измерения в отношении процессов и продуктов (Process and Product Measurement)

В силу того, что приложение количественных оценок к программной инженерии может быть достаточно сложным, в частности, в терминах моделирования или методов анализа, существует ряд фундаментальных аспектов измерений в программной инженерии, лежащих в основе многих более детальных измерений и процессов анализа. Более того, результаты усилий по совершенствованию процессов и продуктов, могут быть оценены только в том случае, если установлены количественные характеристики заданных параметров <процессов и продуктов> для заданных вех или, более точно (так как измерения, все же, выходят за рамки вех конкретных проектов, если, конечно, сама SPI-деятельность не позиционировать как проект, что, конечно, возможно), так называемых “базовых линий” (baseline\*).

\* термин baseline обычно используется в контексте управления изменениями, требованиями и, часто, конфигурациями, в целом, для именования временных “срезов” всего комплекса соответствующих активов.

Измерения могут проводиться для поддержки инициирования реализации и изменения процессов и для оценки результатов таких работ. Также, измерения могут выполняться и в отношении самих продуктов.

Ключевые понятия, термины и методы измерений в приложении к программному обеспечению определены в стандарте ISO 15939 “Software Engineering - Software Measurement Process” и международном словаре метрологии ISO. ISO 15939 также определяет стандартный процесс для измерения характеристик процессов и продуктов.

Необходимо отметить, что в литературе встречаются некоторые терминологические отличия, например, термин “метрика” (*metric*) часто используется вместо термина “измерение” (*measure*)\*\*.

\*\* В данном переводе эти термины используются взаимозаменяемым образом, за исключением случаев, когда контекст обсуждения предполагает разделение процесса измерения – “измерения”, как такового, и критериев/параметров или результатов измерения – “метрик”.

##### 4.1 Измерения в отношении процессов (Process Measurement)

Используемый здесь термин “process measurement” – “измерения в отношении процесса” подразумевает сбор, анализ и интерпретацию количественной информации о процессе. Измерения используются для идентификации сильных и слабых сторон процесса (strengths and weaknesses) и для оценки процесса после того, как он реализован и/или изменен.

Также, проведение количественной оценки процесса может преследовать и другие цели. Например, соответствующие измерения полезны для управления программными проектами. В обсуждаемом здесь контексте, фокус измерений в отношении процессов направлен на оценку реализации и/или изменения процесса.

Рисунок 2 иллюстрирует важность предположений, делаемых в большинстве программных проектов, в которых процесс непосредственно влияет на результаты проекта. Соответствующий контекст воздействует на связь между процессом и его результатом. Другими словами, это означает, что связь процесс-результат процесса находится в зависимости от контексте.

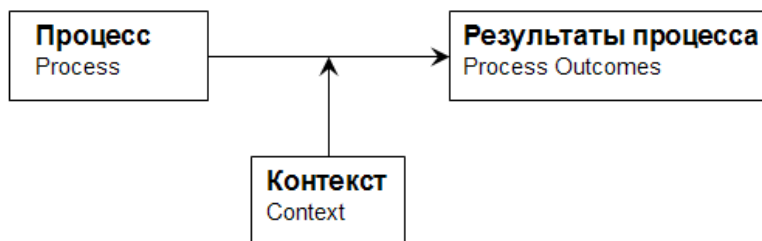


Рисунок 2. Связь между процессом и его результатами (или “выходом процесса” – process outcome).

Далеко не каждый процесс обладает положительным влиянием на получаемые результаты. Например, внедрение инспекции <получаемого> программного обеспечения может сократить усилия и стоимость по тестированию, но может и увеличить время, если каждая инспекция приводит к нарушению расписания просто в силу продолжительности соответствующих инспекционных действий. Поэтому, рекомендуется использовать множество метрических показателей (метрик), по которым оценивается процесс и его результат(ы), безусловно, в контексте значимых для бизнеса характеристик.

Хотя определенные усилия могут направляться на решение вопросов использования соответствующего инструментария, главный ресурс, который нуждается в управлении – персонал. Как результат, наиболее важные метрики касаются оценки продуктивности команд и процессов (например, может использоваться оценка функциональных точек, производимых на единицу трудозатрат\* – “person-effort”), а также, ассоциированного с этим уровня опыта в программной инженерии, в целом, и в отдельных технологиях, в частности.

\* трудозатраты чаще всего определяются как “человеко-час”, “человеко-день” или “человеко-месяц”; здесь полезно обратиться к юбилейному второму изданию классического труда Фреда Брукса “Мифический человеко-месяц” [Брукс, 1995].

Результаты процесса могут, например, оцениваться в отношении качества продукта (как число сбоев на тысячу строк кода – KLOC, Kilo-Lines of Code или на функциональную точку – FP, Function Point), сопровождаемость (усилия, необходимые для реализации определенного типа изменений), продуктивность (LOC, Lines Of Code или FP за человеко-месяц), время вывода продукции на рынок (time-to-market) или степень удовлетворенности потребителей (по измерениям результатов опросов пользователей). Метод оценки связи между процессом и его “выходом” (результатом) зависит от конкретного контекста, например, масштабов (размеров) организации.

В общем случае, мы фокусируемся на результатах процесса. Однако, для достижения заданных результатов (например, в терминах более высокого качества, лучшей сопровождаемости, большей удовлетворенности пользователей) мы должны внедрить соответствующие процессы.

Конечно, не только процессы непосредственно влияют на результат <проекта>. Существуют и другие факторы, играющие не менее важную роль, например, возможности инструментов и потенциал, знания и опыт специалистов. Когда оценивается влияние изменения процессов, такие факторы должны учитываться (например, попытка внедрения развитых процессов программной инженерии при отсутствии ресурсов или в неподготовленной организационной среде практически наверняка приведет к краху таких инициатив). Кроме того, важна степень институализации процессов (process instiutalization или process fidelity – следование заданным процессам как повседневная практика работы). Фактор институализации в большинстве случаев объясняет, почему “хорошие” процессы не приводят к желаемому результату, когда процессы полностью или частично остаются лишь на бумаге.

#### 4.2\* Измерения в отношении программных продуктов (Software Product Measurement)

Измерения в отношении программного продукта включают количественную оценку его размера, структуры и качества.

\* данная тема рассматриваемой области знаний “потеряла” нумерацию в при верстке оригинального варианте SWEBOK 2004, поэтому, далее, нумерация тем смещена.

#### 4.2.1 Оценка размера (Size measurement)

Размер программного продукта чаще всего оценивается по его “длине” (например, в количестве строк кода в модулях, страниц спецификаций требований и других документов) или функциональности (например, по количеству функциональных точек в спецификации). Принципы количественной оценки “функционального” размера программного обеспечения описываются в стандартах:

- IEEE 14143-1 “Information Technology - Software Measurement - Functional Size Measurement - Part 1: Definitions of Concepts”
- ISO 19761 “Software Engineering - Cosmic FPP - A Functional Size Measurement Method”
- ISO 20926 “Software Engineering - IFPUG 4.1 Unadjusted Functional Size Measurement Method-Counting Practices Manual”
- ISO 20968 “Software Engineering-MK II Function Point Analysis – Counting Practices Manual”

#### 4.2.2 Оценка структуры (Structure measurement)

Существует широкий спектр метрик, которые можно использовать для оценки структуры программного обеспечения – в отношении высокоуровневого и низкоуровневого дизайна, а также артефактов кода для анализа потоков работ, потоков данных, вложенности <вызовов>, структур контроля, модульности, взаимодействия и т.п.

#### 4.2.3 Оценка качества (Quality measurement)

Качество, как многомерная характеристика программного обеспечения, наиболее сложно для количественной оценки, в отличие от других характеристик, описанных выше. Более того, некоторые параметры качества требуют, в большей степени, применения качественной, а не количественной оценки. Детальное обсуждение вопросов оценки качества представлено в области знаний SWEBOK “Software Quality” (тема 3.4). ISO разработала соответствующие модели качества программного обеспечения и связанных с ним метрик (см. стандарт ISO 9126 “Software Engineering - Product Quality”, части 1-4).

#### 4.3 Качество результатов измерений (Quality Of Measurement Results)

Качество результатов измерений (точность, воспроизводимость, повторяемость, изменяемость, случайность ошибок измерений) является основой программ проведения количественных оценок для получения эффективных и ограниченных (**ограниченного количества значимых**) результатов. Ключевые характеристики результатов измерений и связанного с ними качества инструментов измерения (**в первую очередь, обоснованности используемого математического аппарата**) определены в международном словаре метрологии ISO (International vocabulary on metrology).

Теория количественной оценки устанавливает основу для возможных измерений. Измерения (и соответствующие типы “размерностей” или “шкал”) описаны в этой теории как систематическое определение численных величин для представления свойств объектов.

SWEBOK подчеркивает важность определения масштабов измерений и понимания каждого типа “размерности” (**как мы увидим далее, под этим термином могут подразумеваться определенные категории метрических показателей - метрик**) с учетом связи с последующим выбором методов анализа данных. Выразительная сторона размерностей связана с классификацией метрик. Для этого теория количественных оценок предлагает последовательность наложения все более детальных ограничений для выделения соответствующих (и все более специализированных) групп метрик. Если метрические показатели используются только для отметки объектов с целью классификации (**например, в простейшем случае, бинарной классификации - “да/нет”, “удовлетворяет/не удовлетворяет”**), такие значения называют *номинальными* (nominal). Если значения определяются для ранжирования (ranking) объектов (например, “хороший”, “лучший чем”, “наилучший”), эти показатели называют *порядковыми* (ordinal). Если величины метрических показателей определяются относительно заданных единиц измерений, такие показатели называют *интервальными* (interval). Наконец, встречаются *пропорциональные* (ratio) показатели (основывающиеся на оценке взаимного отношения различных значений показателей, каждое из которых измеряется разницей между величиной показателя и нулем).

Хотелось бы обратить внимание на описание концепции и использования метрических показателей в специальной главе “Метрические показатели, применяемые при оценке размера программ” русского перевода книги “Управление программными проектами” [Фатрелл, Шафер и Шафер, 2003, глава 21, с.692-748].

#### 4.4 Информационные модели (*Software Information Models*)

По мере сбора данных и наполнения ими репозитория измерений, становится возможно построить соответствующие информационные модели на основе собранных данных и имеющихся знаний.

Эти модели применяются для анализа, классификации и предсказания <характеристик и поведения измеряемых объектов>. Оценка моделей необходима для обеспечения достаточной степени точности и понимания их ограничений. Также необходимо отметить важность работ, направленных на уточнение моделей как в процессе ведения проекта, так и после его завершения.

##### 4.4.1 Построение модели (*Model building*)

Построение модели включает калибрование и оценку модели. Ориентированный на цель подход к измерениям наполняет процесс построения модели необходимым содержанием, то есть модель конструируется для ответа на значимые вопросы и достижения целей совершенствования создаваемого программного обеспечения. На этот процесс также оказывают влияние неявные ограничения используемых метрических показателей и связанных с ними методов анализа. Модель калибруется и оценивается на основании уже накопленных результатов наблюдений (например, по недавно выполненным проектам или проектам, аналогичным данному по используемым технологиям и т.п.) и сравнения ее эффективности с точки зрения соответствия прогнозов реальным данным.

##### 4.4.2 Внедрение модели (*Model implementation*)

Внедрение модели включает интерпретацию и уточнение моделей. Откалиброванные модели применяются в отношении процесса, их результаты интерпретируются и оцениваются в контексте процесса/проекта, после чего модели уточняются в тех аспектах, где это необходимо.

#### 4.5 Техники количественной оценки процессов (*Process Measurement Techniques*)

Определенные техники измерения процесса могут использоваться для анализа процессов программной инженерии и идентификации их преимуществ и недостатков (сильных и слабых сторон). Такие техники применяются во многих случаях для инициирования или оценки влияния (последствий) внедрения или изменения процессов.

Качество результатов измерений, в терминах точности, повторяемости и воспроизводимости, связано с инструментальной составляющей и используемой концепцией оценки и точкой зрения в отношении измерений (например, когда оценивающее лицо – ассессор - выставляет оценки по конкретным процессам).

Техники измерения процесса классифицируются по двум типам: аналитическая и эталонная (*benchmarking*). Эти два типа используются вместе, так как основываются на различных типах информации.

##### 4.5.1 Аналитические техники (*Analytical techniques*)

Аналитические техники характеризуются, как зависящие от “количественных свидетельств того, где необходимы усовершенствования и где инициативы по совершенствованию оказались успешны”. Аналитический тип, иллюстрируемый, например, подходом QIP (*Quality Improvement Paradigm*) состоит из цикла “понимание-проверка-приложение”. Техники, представленные ниже, приведены в качестве других примеров аналитического подхода к измерениям и отражают достаточно типичную практику реализации такого <аналитического> взгляда на проведение количественной оценки. Будут или нет использоваться эти техники в практике конкретной организации зависит, как минимум, от зрелости ее организационной культуры и используемых процессов.



- *Экспериментальные исследования (Experimental Studies)*. Проводятся в специально подготовленном “окружении” для оценки <нового или измененного> процесса. Обычно новый (или измененный) процесс сравнивается с существующим для определения того, в какой степени “старый” процесс дает лучшие результаты, по сравнению с новым процессом.

Другой тип экспериментальных исследований – “симуляция” процесса (моделирование его поведения и результатов, *прим. автора*). Этот тип исследований может использоваться для анализа поведения процесса, выяснения потенциальных возможностей усовершенствования процесса, предсказания результатов процесса (для того случая, если существующий процесс изменяется определенным образом) и контроля выполнения процесса. В качестве первичных данных для симуляции процесса, обычно, используются данные текущего (существующего) процесса.

- *Обзор (оценка) определения процесса (Process Definition Review)* подразумевает, каким образом оценивается определение процесса для идентификации его недостатков и потенциальных аспектов совершенствования. Один из легких способов анализа процесса – сравнение его с существующими стандартами (например, IEEE/ISO/ГОСТ 12207). При таком подходе метрические показатели обычно не собираются, или, в случае их наличия, играют лишь “поддерживающую” (второстепенную) роль. Специалисты, выполняющие анализ определения процесса, используют свои знания, опыт и другие возможности для принятия решения какие изменения процесса могут потенциально привести к желаемому результату в отношении “выходов” процесса (получаемого программного продукта или его отдельных элементов). Наблюдения (observations) за выполнением процесса также могут дать дополнительные данные, позволяющие идентифицировать возможные пути совершенствования процесса.
- *Ортогональная классификация дефектов (Orthogonal Defect Classification)* – техника, которая может быть использована для связывания (отображения) сбоев с их потенциальными причинами. В данном контексте может быть полезен для детального ознакомления стандарт IEEE 1044 “Standard for the Classification of Software Anomalies”, классифицирующий возможные сбои (аномалии) в работе программного обеспечения.
- *Анализ причин (Root Case Analysis)* является еще одной популярной техникой, часто используемой на практике. Эта техника предполагает “спуск” от обнаруженного сбоя к идентификации его причины, изменяя сам процесс (или, по аналогии, код программного обеспечения, если бы речь шла о поиске дефекта, приводящего к сбою) до тех пор, пока сбой не исчезнет и реструктурируя процесс с тем, чтобы обнаруженная проблема не повторялась в будущем.

Описанная выше ортогональная классификация дефектов может использоваться для определения категорий различных сбоев и, соответственно, путей обнаружения их причин. Такая классификация добавляет количественные показатели к технике анализа причин.

- *Статистический контроль процесса (Statistical Process Control, SPC)* – эффективный путь для определения стабильности (или отсутствия стабильности) процесса.
- Индивидуальный программный процесс (Personal Software Process, PSP) определяет серию возможных улучшений в индивидуальной практике разработки программного обеспечения. Предполагает движение “снизу-вверх”, включая сбор персональных данных и их интерпретацию для повышения индивидуальной продуктивности специалистов.

Хотя SWEBOK это и не упоминает, однако, существует и развитие PSP – Team Software Process (TSP), направленный на аспекты повышения качества командной работы, включая совершенствование взаимодействия между членами проектной команды.

#### 4.5.2 Эталонные техники (Benchmarking techniques)

Этот тип техник основывается на идентификации “совершенной” организации процесса и связанных с ней практиках и инструментах. Предполагается, что если менее опытная команда (организация, компания) применяет успешные подходы более опытной организации, принимаемой

в качестве эталона, менее опытная команда также станет “совершенной”, то есть улучшит свои процессы до уровня данного успешного примера. Данная техника уделяет специальное внимание оценке зрелости организации и/или потенциальных возможностей ее процессов ([ресурсов, культуры, бизнес-практик и т.п.](#)).

В определенной степени, CMMI (и аналогичные модели в области управления проектами, например, PMI OPM3 и менеджмента качества, например, Six Sigma) предоставляют обоснованный и подтвержденный базис для использования эталонной техники.