Классификация методологий, моделей и стандартов управления разработкой ПО

Выбор адекватных проектных методологий, моделей ЖЦ ПО, метрик проектов и других инструментальных средств управления ПО представляет собой достаточно сложную задачу для компаний-разработчиков ПО и проектных команд. На практике они не всегда представляют себе все существующие варианты выбора методов, моделей, стандартов и методологий, рассматривают их ограниченный набор, подходят к решению проблемы выбора без учета специфики компании и проекта.

Зачем нужны методологии?

Подходы к выбору методологий, моделей и стандартов зависят от предполагаемой цели использования. Чаще всего они используются следующим образом:

- Как эталон (желаемое состояние организации процессов, выполнения работ) «*эффективная методология – серебряная пуля*»;
- Для обоснования существующей практики управления проектами. Как доказательство (обоснование) правильности принятых решений
- Как руководство по достижению целей – *«методология»*
- Как набор требований (ограничений), предъявляемых внешней средой: клиентами, партнерами, собственниками, государственными организациями, профессиональными объединениями – *«стандарт»*
- Для объяснения (понимания) существующей практики. Используется терминология, концепции, идеи – *«модель»*
- Для обеспечения «эффективного» диалога между менеджментом, участниками проектных команд, клиентами и субподрядчиками – *«глоссарий»*
- Для определения (изменения) фокуса – центра приложения управленческих усилий, инвестиций в методологии.

Выбор методологий.

Менеджмент и участники проектных команд должны сформировать единое понимание в отношении целей и методов управления ИТ-проктами. Решение проблемы выбора требует двух типов компетенций:

- во-первых, компетенций в области методологий,
- во-вторых, компетенций в области подходов к выбору.

Классификации.

Полезным инструментом для решения задачи выбора является классификация. Для создания **классификации методов, моделей и стандартов** определены различные признаки *группировки* и выделены *группы* по этим признакам.

По характеру обоснования рекомендаций: концептуальные и эмпирические.

Концептуальные модели получены рационально-логическим методом, а эмпирические – чувственно-опытным. В основе концептуальных инструментальных средств лежат концепции теории менеджмента, такие как процессное управление и ре-инжиниринг бизнес-процессов, управление проектами, управление качеством. Универсальные концепции адаптируются к особенностям управления разработкой ПО, которую отличают проектный характер деятельности, технологическая гибкость процесса разработки, неопределенность требований в отношении ожидаемого результата и высокие риски.

Примером концептуальной модели является модель зрелости технологических процессов SEI (Capability Maturity Model, CMM). Концептуальной является методология PRINCE и рациональный унифицированный процесс (Rational Unified Process, RUP).

Эмпирические методологии разработаны на основе теоретического обобщения успешных практик ИТ-проектов. Примерами эмпирической модели служат SCRUM, XP, Crystal.

В зависимости от назначения: модели зрелости и процессные модели, проектные методологии и индивидуальные и групповые практики.

Модель зрелости CMMI, модель оценки процессов SPICE и стандарт ISO 9000 используются для управления ИТ-компанией (подразделением). Проектные методологии: MSF, SCRUM, XP, используются для управления ИТ-проектами разработки ПО. Методологии внедрения информационных систем используются для организации

проекта внедрения. Командные и индивидуальные практики (PSP, Personal Software process; TSP, Team Software process) служат для непрерывного повышения эффективности работы команд и индивидуальных разработчиков.

В зависимости от условий реализации проекта: прогнозируемые и адаптивные.

Прогнозируемые методологии основываются на предпосылке о возможности и целесообразности детального планирования будущего. Для ИТ-проекта формулируются требования к разрабатываемой системе, формируется план проекта и определяется потребность в ресурсах. Изменения в плане проекта и требованиях считаются нежелательными. Проектные методологии этого класса используют каскадную модель жизненного цикла.

Адаптивные методологии нацелены на преодоление ожидаемой неполноты требований и их постоянного изменения. В основе адаптивных методологий лежит итерационная модель жизненного цикла. Примером адаптивных методологий являются Scrum, Crystal, Extreme Programming. Адаптивные методологии учитывают психологические особенности процесса разработки ПО. Одним из значимых факторов успеха использования адаптивных методологий является высокая квалификация специалистов, в первую очередь – разработчиков.

По характеру знаний и фокусу: инженерные, управленческие и интегрированные.

Инженерные инструменты (software engineering) основываются на технологических принципах и направлены на совершенствование конечных продуктов, таких как программный код, тестовые примеры, прототипы, документация. Инженерные инструментальные необходимы квалифицированному разработчику.

Управленческие инструментальные средства (software project management and process management) основываются на принципах теории управления (менеджмента), в их основе лежат такие концепции, как тотальное управление качеством, управление проектами, управление знаниями.

Интегрированные инструментальные средства объединяют инженерные концепции и концепции управления.

Проектные методологии находятся в центре теории управления разработкой ПО, поэтому для них можно предложить расширенный набор групп на основе различных характеристик проекта.

В зависимости от проектных рисков (предложено А. Коберном). В зависимости от характера потерь в случае неудовлетворительной работы ПО выделяются проекты с различными видами рисков:

- «потеря комфорта»
- «потеря денег»
- «потеря больших денег и бизнеса»
- «потеря жизни»

В зависимости от технологии проекта (для проектных методологий): универсальные, структурные, объектные и метрологии для сервисно-ориентированной архитектуры (SOA).

На Рис. 1. представлена предлагаемая нами классификация инструментальных средств управления разработкой ПО. В центр классификации помещены две группы методологий: *методологии управления разработкой ПО и методологии внедрения информационных систем* (ИС). Методологии содержат рекомендации по использованию отдельных инструментов: метрик, технических стандартов, языков графического моделирования.

Методологии включают в себя:

описание рекомендуемой модели жизненного цикла разработки (внедрения) ПО, модель команды проекта и ролей, а также используемых методик, техник.

В зависимости от лежащей в основе методологии управления проектом разработки ПО модели жизненного цикла, проектные методологии варьируют от классической каскадной до итеративных методологий.

Методологий внедрения информационных систем представляют собой набор методологий, разработанных специально для внедрения той или иной информационной системы. В некоторых случая для одной системы может существовать несколько альтернативных методологий внедрения. Методологии управления ИТ-проектами формируются на основе теории управления проектами. Модели управления ИТ-компаниями опираются на концепции тотального управления качеством и процессного управления.

В левой части классификации представлены универсальные *концепции менеджмента*: всеобщее управление качеством (TQM), процессное управление и реинжиниринг безнес-процессов (BPR) и управление

проектами (PM), а также различные вычислительные техники. Самостоятельную группу образуют управленческие стандарты, среди которых наиболее известен стандарт ISO 9000.

Универсальные концепции менеджмента аккумулируют опыт и лучшие управленческие практики, которые стали основой методологий совершенствования деятельности компаний-разработчиков, таких как модели зрелости (CMM/CMMI), стандарты оценки и улучшения процессов (SPICE), и TickIT. Эти модели и стандарты регламентируют организационно-управленческую и технологическую среду, в условиях которой применяются проектные методологии.

Самостоятельный кластер представляют собой *индивидуальные техники*, среди которых выделяется собственный процесс разработки (PSP). В основе индивидуальных техник также лежат концепции управления.

Вычислительные *техники* представляют собой алгоритмы оценки, анализа и оптимизации, используемые в управлении компанией и отдельными проектами. В правой части представлены *метрики* и *языки моделирования*. Метрики служат для получения фактических и плановых количественных оценок для процессов, проектов и продуктов. Использование метрик является косвенным признаком применения концепций управления качеством в разработке ПО и зрелости организации процессов разработки.

Языки графического моделирования используются для создания понятных и согласованных требований и проектных решений. Универсальный язык моделирования (UML) позволяет однозначно транслировать графические нотации в проектный код, а также порождать графические описания на основе программного кода. Развитие графических нотаций, как составной части автоматизированного проектирования, оказывает влияние на проектные методологии.

В управлении проектами разработки ПО существует набор стандартных (типовых) задач, которые могут решаться одинаково, вне зависимости от технических и организационных особенностей проекта и выбранной для него методологии. Для решения этих задач предназначены *методики*. Методика служат для решения одной задачи и включают в себя описание области применения, алгоритмы использования, описание необходимых исходных данных. Методики могут предусматривать использование определенных метрик, языков моделирования и стандартов.

Методики содержат полные рекомендации по решению отдельных проектных задач, таких как управление рисками, или оценка трудоемкости проекта. Примерами методик служат SEI Risk Evaluation Method или COCOMO.

В нижней части классификации представлены группы стандартов, регламентирующих различные аспекты разработки ПО. Стандарты разрабатываются международными и государственными организациями по стандартизации, отраслевыми комитетами, исследовательскими институтами, крупными компаниями, такие как ISO (International Organization for Standardization), SEI (Software Engineering Institute), DoD (Department of Defense USA), IEEE (Институт Электронной и Электротехнической Инженерии, Institute of Electrical and Electronics Engineers), МЭК (Международная Электротехническая комиссия), а так ИТ-компании: Bell, Hewlett Packard, Sun Microsystems, IBA, Oracle, Microsoft и др. Стандарты, регламентирующие требования к процессам разработки и выходным продуктам, дополняют методологии управления разработкой ПО.

Объектами стандартизации в сфере ИТ являются:

- Конструкторская документация (состав, структура, требования к оформлению);
- Стандарты кодирования и оформления программных текстов;
- Терминология и определения;
- Модели процессов;
- Модели жизненного цикла;
- Требования к безопасности хранения и передачи информации и способы ее обеспечения;
- Качество программного обеспечения, характеристики качества, методы получения данных по качеству;
- Графические и нотации и инструменты формализованного описания требований и технических решений;
- Форматы хранения данных, обмена и передачи данных.

Модели и стандарты, регламентирующие процессы и жизненный цикл, лежат в основе системы менеджмента качества компании, используются при разработке проектных планов.

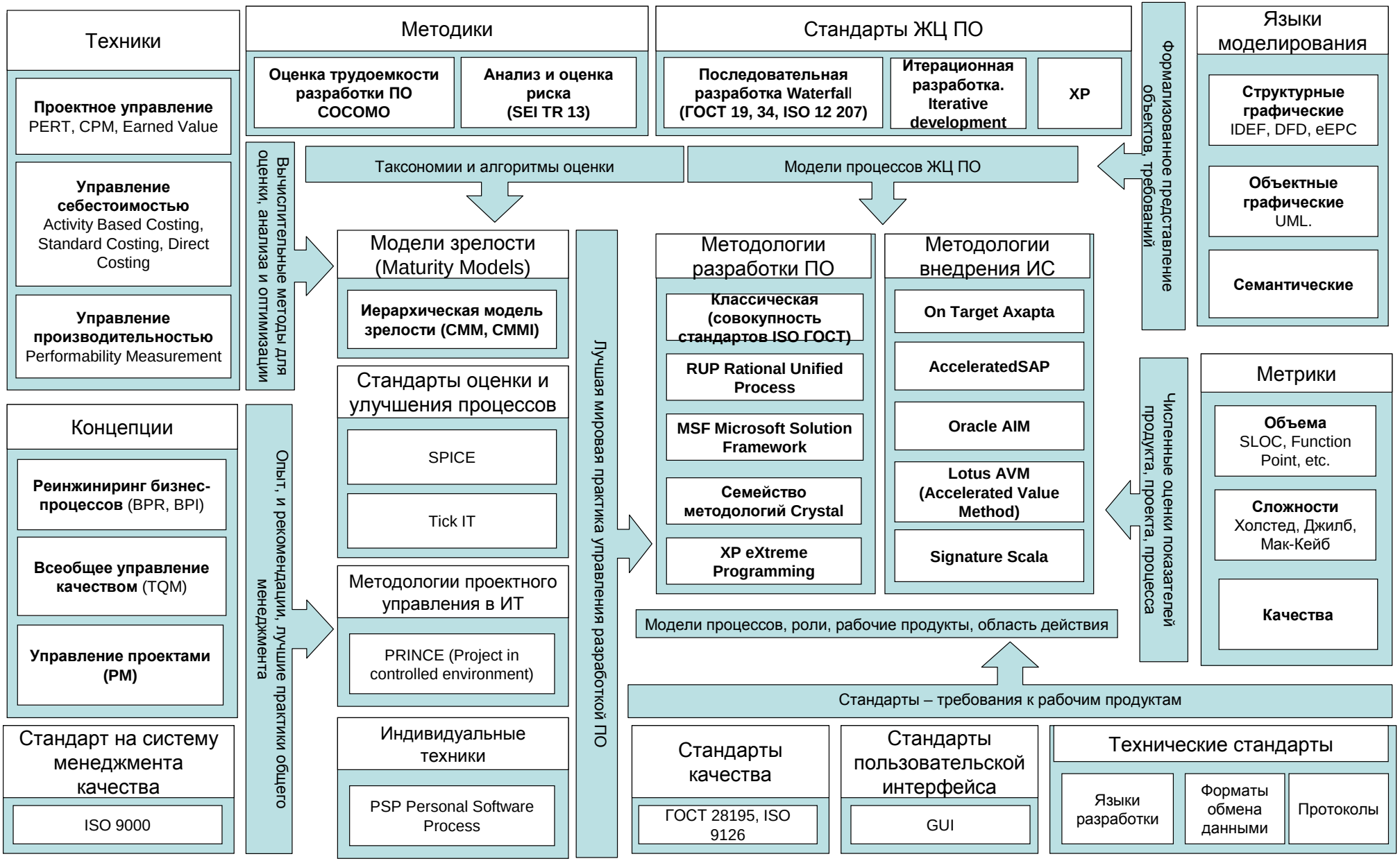

Рис. 1. Классификация методологий, методов, моделей и стандартов разработки программного обеспечения

Методологии. В результате анализ методологий управления разработкой ПО были выявлены две группы методологий, которые отличаются целью их использования, историей создания и назначением.

Первая группа представляет собой методологии, целью которых является успешное выполнение отдельного проекта. К этой группе относится большинство проектных методологий, и практически все адаптивные методологии. Логика успешного функционирования организации выглядит как рост компетенций, создание технических активов за счет последовательного выполнения успешных проектов.

Вторая группа включает в себя методологии, обеспечивающие устойчивое функционирование компании-разработчика ПО, нацелены на последовательное развитие компетенций. К этой группе относятся модели зрелости (CMM, CMMI). Логика успешного функционирования предполагает создание, контроль и непрерывное улучшение способностей организации к выполнению проектов, и, как следствие, успешное выполнение проектов.

Две выделенные группы отличаются не только целью использования, но и историей создания и развития, а также практикой использования.

Проектные методологии представляют собой ядро теории управления разработкой ПО. К существующей классификации в зависимости от используемой в ней модели жизненного цикла (водопадные (каскадные) и итерационные методологии) добавилась более общая классификация на прогнозируемы и адаптивные методологии. К адаптивным относятся все методологии, которые удовлетворяют требованиям, сформулированным в Манифесте адаптивной разработки.

Прогнозируемые (предикативные) методологии фокусируются на детальном планировании будущего. Известны запланированные задачи и ресурсы на весь срок проекта. Команда с трудом реагирует на возможные изменения. План оптимизирован исходя из состава работ и существующих требований. Изменение требований может привести к существенному изменению плана, а также дизайна проекта. Часто создается специальный комитет по «управлению изменениями» (change control board) чтобы в проекте учитывались только самые важные требования (ГОСТ 19, 32).

Адаптивные методологии нацелены на преодоление ожидаемой неполноты требований и их постоянного изменения. Когда меняются требования, команда разработчиков тоже меняется. Команда, участвующая в адаптивной разработке, с трудом может предсказать будущее проекта. Существует точный план лишь на ближайшее время. Более удаленные во времени планы существуют лишь как декларации о целях проекта, ожидаемых затратах и результатах. Среди адаптивных методологий: (Scrum, Crystal, Extreme Programming, Adaptive Software Development, DSDM, Feature Driven Development, Lean software development).

Предлагаемая классификация структурирует дальнейшую работу по определению подходов к выбору методов моделей и стандартов в следующих направлениях:

- Определение влияния концепций управления (управления качеством, проектами, рисками и процессами) на методологию управления разработкой ПО;
- Структурный сравнительный анализ методов, моделей, стандартов и методологий.
- Анализ должен подтвердить или опровергнуть следующие гипотезы:
- Существуют группы инструментальных средств, которые объединяют методологии, имеющие сходное назначение, похожую историю создания.
- Методологии одной группы будут иметь сходную структуру и содержание.
- Методологии одной группы требуют примерно равного уровня усилий для внедрения в рабочую практику.
- В зависимости от «уровня зрелости» организации должна использоваться та или иная группа инструментальных средств.
- Методологии одной группы опираются сходный набор концепции управления.
- Методологии содержат «строительные блоки» - обособленные элементы, которые можно использовать для создания адаптированных методологий для решения уникальных задач.
- В основе каждой методологии лежит набор «принципов», которые могут быть истинными или ложными для проекта и (или) организации.

1. Standish Group Chaos Report http://www.standishgroup.com/sample_research/chaos_1994_1.php
2. Alexander C., Ishikawa S., Silverstein M., Jacobson M., Fiksdahl-King I., Angel S.. A Pattern Language. Oxford University Press, New York, 1977
3. Boehm B. COCOMO II Model Definition Manual, Center for Software Engineering, 1998 - 37p
4. Booch G., Rumbaugh, Jacobson I. The Unified Modeling Language User Guide. – Addison-Wesley, 1999 – 482 p.
5. Capability Maturity Model® Integration (CMMI SM), Version 1.1 Continuous Representation CMU/SEI-2002-TR-011 ESC-TR-2002-011 CMMI Product Team Copyright 2002 by Carnegie Mellon University. 725 p
6. Guide to SWBok http://www.swebok.org/
7. Humphrey W. S. The Personal Process in Software Engineering, A personal Commitment to Software Quality. — SEI, 1994. —29 p.
8. ISO/IEC 19501:2005 Information technology -- Open Distributed Processing -- Unified Modeling Language (UML) Version 1.4.2
9. Manifesto for Agile Software Development http://agilemanifesto.org/
10. Paulk M. C., Cutis B., Chriss M. B., Weber Ch. V. Capability Maturity Model for Software, Version 1.1 CMU SEI, 1993. — 533 p.
11. Porter M. Competitive Strategy: Techniques for Analyzing Industries and Competitors New York: free Press, 1980
12. Rob Thomsett. Double Dummi Spit and other Estimation Games. American Programmer, June 1996.
13. Royce, Winston W., Managing the Development of Large Scale Software Systems, Proceedings of IEEE WESCON, pp. 1--9, August 1970.
14. Sisti F., Joseph S. Software Risk Evaluation Method, Version 1.0, SEI 1994. — 141 p.
15. Schwaber Ken Agile Project Management with Scrum, Microsoft Press 2004 – 163 p.
16. Бек К. Экстремальное программирование: планирование.- СПб: Питер, 2003.- 143 с.
17. Брукс Ф. Мифический человеко-месяц или как создаются программные системы. – Пер. с англ. – СПб.: Символ-Плюс, 1999. – 304 с.
18. Крачтен Филипп. Введение в Rational Unified Process. Вильямс, 2002 г. - 240 стр.
19. Коберн А. Каждому проекту своя методология. Humans and Technology Technical Report, TR 99.04, Oct.1999 7691 Dell Rd, Salt Lake City, UT 84121 USA
20. Коберн А. Быстрая разработка программного обеспечения. М: Лори, 2002 г. – 314 с.
21. Ройс У. Управление проектами по созданию программного обеспечения. Унифицированный подход. М.: Лори, 2002. – 434 с.
22. Йордон Э. Путь камикадзе. Как разработчику программного обеспечения выжить в безнадежном проекте. Издательство «Лори», 2001 – 256 с.
23. Фаулер М. Новые методологии программирования http://www.silicontaiga.ru/home.asp?artId=4889