Глава 8: Процесс разработки программного обеспечения

Из SWEBOK

Содержание

АКРОНИМЫ

BPMN	Нотация моделирования бизнес-процессов
КЕЙС	Компьютерная разработка программного обеспечения
CM	Управление конфигурацией
CMMИ	Модель зрелости интеграции
GQM	Цель-Вопрос-Метрика
IDEF0	Определение интеграции
LOE	Уровень усилий
ОДК	Классификация ортогональных дефектов
SDLC	Жизненный цикл разработки программного обеспечения
SPLC	Жизненный цикл программного продукта
UML	Единый язык моделирования

ВВЕДЕНИЕ

Инженерный процесс состоит из набора взаимосвязанных действий, которые преобразуют один или несколько входов в выходы, при этом потребляя ресурсы для выполнения преобразования. Многие процессы традиционных инженерных дисциплин (например, электрические, механические, гражданские, химические) связаны с преобразованием энергии и физических объектов из одной формы в другую, как в гидроэлектростанции, которая преобразует потенциальную энергию в электрическую, или в нефтеперерабатывающем заводе, который использует химические процессы для преобразования сырой нефти в бензин.

В этой области знаний (KA) процессы разработки программного обеспечения касаются рабочих действий, выполняемых инженерами-программистами для разработки, обслуживания и эксплуатации программного обеспечения, таких как требования, проектирование, создание, тестирование, управление конфигурацией и другие процессы разработки программного обеспечения. Для удобства чтения «процесс разработки программного обеспечения» будет называться «процессом программного обеспечения» в этом КА. Кроме того, обратите внимание, что «программный процесс» означает рабочие действия, а не процесс выполнения внедренного программного обеспечения.

Программные процессы определяются по ряду причин: для облегчения человеческого понимания, общения и координации; для помощи в управлении программными проектами; измерять и улучшать качество программных продуктов эффективным образом; поддерживать улучшение процесса; и обеспечить основу для автоматизированной поддержки выполнения процесса.

KA SWEBOK, тесно связанные с этим KA Процесса разработки программного обеспечения, включают управление разработкой программного обеспечения, модели и методы разработки программного обеспечения и качество программного обеспечения; Тема измерения и анализа первопричин, найденная в Engineering Foundations KA, также тесно связана. Управление программной инженерией связано с

адаптацией, адаптацией и реализацией программных процессов для конкретного программного проекта (см. Планирование процессов в КА Управления программной инженерией). Модели и методы поддерживают системный подход к разработке и модификации программного обеспечения.

КА качества программного обеспечения занимается планированием, обеспечением и контролем качества проектов и продуктов. Измерения и результаты измерений в Engineering Foundations КА необходимы для оценки и управления программными процессами.

РАЗБИВКА ТЕМ ПО ПРОЦЕССУ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Как показано на рис. 8.1, этот КА связан с определением процесса разработки программного обеспечения, жизненными циклами программного обеспечения, оценкой и улучшением процесса разработки программного обеспечения, измерением программного обеспечения и инструментами процесса разработки программного обеспечения.

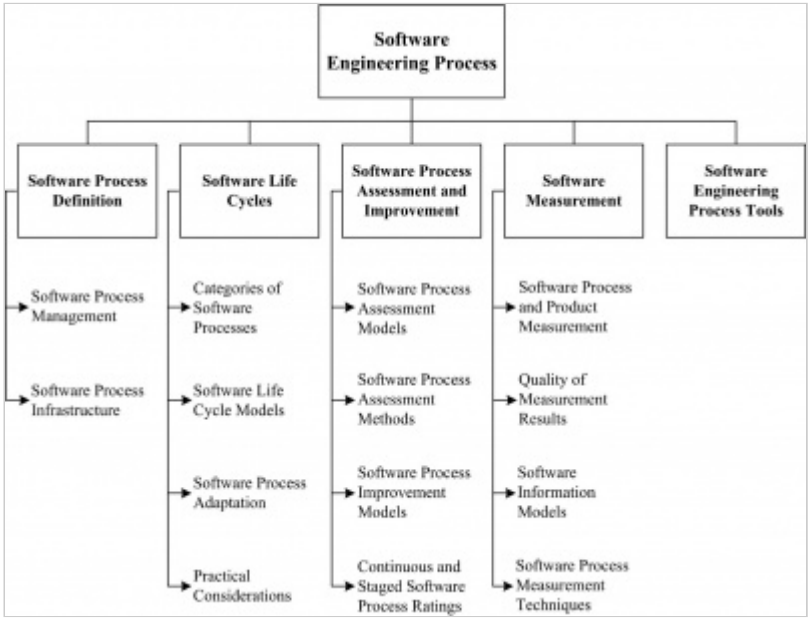

Рисунок 8.1: Разбивка тем для процесса разработки программного обеспечения КА

1 Определение программного процесса

[1 , c177] [2 , c295] [3 , c28-29, c36, c5]

Этот раздел касается определения программного процесса, управления программными процессами и инфраструктуры программных процессов.

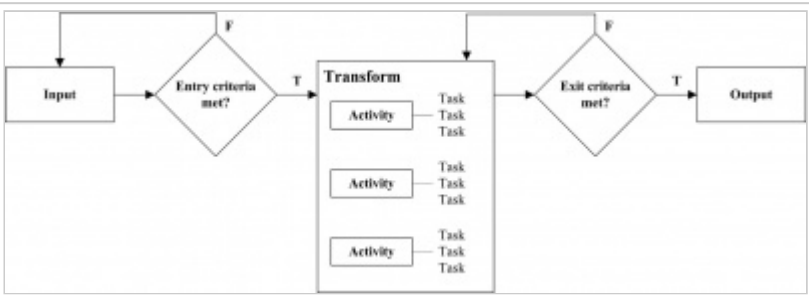

Рисунок 8.2: Элементы программного процесса

Как указано выше, программный процесс представляет собой набор взаимосвязанных действий и задач, которые преобразуют входные рабочие продукты в выходные рабочие продукты. Как минимум, описание программного процесса включает в себя необходимые входные данные, преобразующие рабочие действия и полученные выходные данные. Как показано на рис. 8.2, программный процесс может также включать критерии входа и выхода, а также декомпозицию рабочих операций на задачи, которые являются наименьшими единицами работы, подлежащими учету со стороны руководства. Входные данные процесса могут быть инициирующим событием или выходными данными другого процесса. Критерии входа должны быть удовлетворены до начала процесса. Прежде чем процесс может быть успешно завершен, должны быть выполнены все указанные условия, включая критерии приемки выходного рабочего продукта или рабочих продуктов.

Программный процесс может включать в себя подпроцессы. Например, проверка требований к программному обеспечению — это процесс, используемый для определения того, обеспечат ли требования адекватную основу для разработки программного обеспечения; это подпроцесс процесса требований к программному обеспечению. Входными данными для проверки требований обычно являются спецификация требований к программному обеспечению и ресурсы, необходимые для

выполнения проверки (персонал, инструменты проверки, достаточное время). Задачи деятельности по проверке требований могут включать проверку требований, прототипирование и проверку модели. Эти задачи включают рабочие задания для отдельных лиц и групп. Результатом проверки требований обычно является проверенная спецификация требований к программному обеспечению, которая предоставляет входные данные для процессов проектирования и тестирования программного обеспечения. Проверка требований и другие подпроцессы процесса требований к программному обеспечению часто чередуются и повторяются различными способами; процесс требований к программному обеспечению и его подпроцессы могут быть запущены и завершены несколько раз во время разработки или модификации программного обеспечения.

Полное определение программного процесса может также включать роли и компетенции, ИТ-поддержку, методы и инструменты разработки программного обеспечения и рабочую среду, необходимые для выполнения процесса, а также подходы и меры (ключевые показатели эффективности), используемые для определения эффективности и эффективность выполнения процесса.

Кроме того, программный процесс может включать чередующиеся технические, совместные и административные действия.

Нотации для определения программных процессов включают текстовые списки составляющих действий и задач, описанных на естественном языке; диаграммы потоков данных; карты состояния; БПМН; ИДЕФ0; сети Петри; и диаграммы активности UML. Преобразующие задачи внутри процесса могут быть определены как процедуры; процедура может быть определена как упорядоченный набор шагов или, альтернативно, как контрольный список работы, которая должна быть выполнена при выполнении задачи.

Следует подчеркнуть, что не существует лучшего программного процесса или набора программных процессов. Программные процессы должны быть выбраны, адаптированы и применены соответствующим образом для каждого проекта и каждого организационного контекста. Идеального процесса или набора процессов не существует.

1.1 Управление программным процессом

[3 , c26.1] [4 , стр.453-454]

Две цели управления программными процессами заключаются в том, чтобы реализовать эффективность и действенность, являющиеся результатом систематического подхода к выполнению программных процессов и созданию рабочих продуктов — будь то на индивидуальном, проектном или организационном уровне — и внедрить новые или улучшенные процессы.

Процессы изменяются с расчетом на то, что новый или модифицированный процесс повысит эффективность и/или результативность процесса и качество результирующих рабочих продуктов. Переход на новый процесс, улучшение существующего процесса, организационные изменения и изменения инфраструктуры (внедрение технологий или изменение инструментов) тесно связаны между собой, поскольку все они обычно инициируются с целью улучшения стоимости, графика разработки или качества программного обеспечения. товары. Изменение процесса влияет не только на программный продукт; они часто приводят к организационным изменениям. Изменение процесса или введение нового процесса может иметь волновой эффект во всей организации. Например, изменения в инструментах и технологиях ИТ-инфраструктуры часто требуют изменений процессов.

Существующие процессы могут быть изменены, когда другие новые процессы развертываются в первый раз (например, введение инспекционной деятельности в рамках проекта разработки программного обеспечения, вероятно, повлияет на процесс тестирования программного обеспечения — см. Обзоры и аудиты в КА качества программного обеспечения и в тестировании программного обеспечения). КА). Эти ситуации также можно назвать «эволюцией процесса». Если изменения обширны, то, вероятно, потребуются изменения в организационной культуре и бизнес-модели, чтобы приспособиться к изменениям процессов.

1.2 Инфраструктура программного процесса

[2 , c.183, c.186] [4 , c.437-438]

Создание, внедрение и управление программными процессами и моделями жизненного цикла программного обеспечения часто происходит на уровне отдельных программных проектов. Однако систематическое применение программных процессов и моделей жизненного цикла программного

обеспечения в организации может принести пользу всей работе с программным обеспечением внутри организации, хотя и требует приверженности на организационном уровне. Инфраструктура программных процессов может предоставлять определения процессов, политики для интерпретации и применения процессов, а также описания процедур, которые будут использоваться для реализации процессов. Кроме того, инфраструктура программных процессов может обеспечивать финансирование, инструменты, обучение и сотрудников, на которых возложена ответственность за создание и поддержку инфраструктуры программных процессов.

Инфраструктура программных процессов варьируется в зависимости от размера и сложности организации, а также от проектов, осуществляемых внутри организации. Небольшие, простые организации и проекты имеют небольшие, простые потребности в инфраструктуре. Большие, сложные организации и проекты по необходимости имеют более крупную и сложную инфраструктуру программных процессов. В последнем случае могут быть созданы различные организационные подразделения (например, группа процессов разработки программного обеспечения или руководящий комитет) для наблюдения за внедрением и улучшением процессов программного обеспечения.

Распространенным заблуждением является то, что создание инфраструктуры программных процессов и внедрение повторяющихся программных процессов увеличат время и затраты на разработку и обслуживание программного обеспечения. Существуют затраты, связанные с внедрением или улучшением программного процесса; однако опыт показывает, что внедрение систематического улучшения программных процессов приводит к снижению затрат за счет повышения эффективности, отсутствия доработок и более надежного и доступного программного обеспечения. Таким образом, производительность процесса влияет на качество программного продукта.

2 жизненных цикла программного обеспечения

[1 , c2] [2 , c190]

В этом разделе рассматриваются категории программных процессов, модели жизненного цикла программного обеспечения, адаптация программных процессов и практические соображения. Жизненный цикл разработки программного обеспечения (SDLC) включает в себя программные процессы, используемые для определения и преобразования требований к программному обеспечению в поставляемый программный продукт. Жизненный цикл программного продукта (SPLC) включает в себя жизненный цикл разработки программного обеспечения, а также дополнительные программные процессы, которые обеспечивают развертывание, обслуживание, поддержку, развитие, вывод из эксплуатации и все другие процессы от начала до конца для программного продукта, включая управление конфигурацией программного обеспечения и программное обеспечение. процессы обеспечения качества, применяемые на протяжении всего жизненного цикла программного продукта. Жизненный цикл программного продукта может включать несколько жизненных циклов разработки программного обеспечения для развития и улучшения программного обеспечения.

Отдельные программные процессы не имеют временного порядка между собой. Временные отношения между программными процессами обеспечиваются моделью жизненного цикла программного обеспечения: либо SDLC, либо SPLC. Модели жизненного цикла обычно подчеркивают ключевые программные процессы в рамках модели, а также их временные и логические взаимозависимости и взаимосвязи. Подробные определения программных процессов в модели жизненного цикла могут быть предоставлены непосредственно или со ссылкой на другие документы.

В дополнение к передаче временных и логических взаимосвязей между программными процессами модель жизненного цикла разработки программного обеспечения (или модели, используемые в организации) включает в себя механизмы контроля для применения критериев входа и выхода (например, обзоры проекта, утверждения заказчиком, тестирование программного обеспечения, качество). пороги, демонстрации, консенсус команды). Выходные данные одного программного процесса часто служат входными данными для других (например, требования к программному обеспечению обеспечивают входные данные для процесса проектирования архитектуры программного обеспечения, а также процессов создания и тестирования программного обеспечения). Одновременное выполнение нескольких действий программного процесса может привести к общему результату (например, спецификации интерфейса для интерфейсов между несколькими программными компонентами, разработанными разными группами).

2.1 Категории программных процессов

[1 , предисловие] [2 , с 294-295] [3 , с22-с24]

[1 , предисловие] [2 , с.294-295] [3 , с22-с24]

Многие отдельные программные процессы были определены для использования в различных частях жизненного цикла разработки и сопровождения программного обеспечения. Эти процессы можно классифицировать следующим образом:

1. Первичные процессы включают программные процессы для разработки, эксплуатации и

обслуживание программного обеспечения.

1. Вспомогательные процессы применяются с перерывами или непрерывно на протяжении всего жизненного цикла программного продукта для поддержки основных процессов; они включают программные процессы, такие как управление конфигурацией, обеспечение качества, верификация и валидация.
2. Организационные процессы обеспечивают поддержку разработки программного обеспечения; они включают в себя обучение, анализ измерений процессов, управление инфраструктурой, управление портфелем и повторным использованием, совершенствование организационных процессов и управление моделями жизненного цикла программного обеспечения.
3. Межпроектные процессы, такие как повторное использование, линейка программных продуктов и разработка предметной области; они включают в себя более одного программного проекта в организации.

Программные процессы в дополнение к перечисленным выше включают следующее.

Процессы управления проектами включают процессы планирования и оценки, управления ресурсами, измерения и контроля, руководства, управления рисками, управления заинтересованными сторонами и координации основных, поддерживающих, организационных и межпроектных процессов разработки и обслуживания программного обеспечения.

Процессы программного обеспечения также разрабатываются для конкретных нужд, например, действия процессов, которые касаются характеристик качества программного обеспечения (см. KA качества программного обеспечения). Например, проблемы безопасности во время разработки программного обеспечения могут потребовать одного или нескольких программных процессов для защиты безопасности среды разработки и снижения риска злонамеренных действий. Программные процессы также могут быть разработаны для обеспечения адекватных оснований для установления уверенности в целостности программного обеспечения.

2.2 Модели жизненного цикла программного обеспечения

[1 , с2] [2 , с3.2] [3 , с2.1][5]

Неосязаемая и гибкая природа программного обеспечения допускает широкий спектр моделей жизненного цикла разработки программного обеспечения, начиная от линейных моделей, в которых этапы разработки программного обеспечения выполняются последовательно с обратной связью и итерацией по мере необходимости, за которыми следует интеграция, тестирование и поставка одного продукта. ; к итерационным моделям, в которых программное обеспечение разрабатывается поэтапно с увеличением функциональности в итеративных циклах; к гибким моделям, которые обычно включают частые демонстрации работающего программного обеспечения представителю клиента или пользователя, который направляет разработку программного обеспечения в короткие итерационные циклы, которые производят небольшие приращения работающего, поставляемого программного обеспечения. Инкрементные, итеративные и гибкие модели могут при желании доставлять ранние подмножества рабочего программного обеспечения в пользовательскую среду.

Линейные модели SDLC иногда называют прогнозирующими моделями жизненного цикла разработки программного обеспечения, а итеративные и гибкие SDLC — адаптивными моделями жизненного цикла разработки программного обеспечения. Следует отметить, что различные действия по техническому обслуживанию во время SPLC могут выполняться с использованием различных моделей SDLC, соответствующих действиям по техническому обслуживанию.

Отличительной чертой различных моделей жизненного цикла разработки программного обеспечения является способ управления требованиями к программному обеспечению. Модели линейной разработки обычно разрабатывают полный набор требований к программному обеспечению, насколько это возможно, во время инициации и планирования проекта. Затем требования к программному

обеспечению строго контролируются. Изменения требований к программному обеспечению основаны на запросах на изменение, которые обрабатываются комиссией по контролю за изменениями (см. раздел Запрос, оценка и утверждение изменений программного обеспечения на панели по контролю за изменениями в KA Управления конфигурацией программного обеспечения). Инкрементная модель создает последовательные инкременты работающего, доставляемого программного обеспечения на основе разделения требований к программному обеспечению, которые должны быть реализованы в каждом из инкрементов. Требования к программному обеспечению могут строго контролироваться, как в линейной модели. или может быть некоторая гибкость в пересмотре требований к программному обеспечению по мере развития программного продукта. Гибкие модели могут изначально определять объем продукта и высокоуровневые функции; однако гибкие модели предназначены для облегчения эволюции требований к программному обеспечению в ходе проекта.

Следует подчеркнуть, что континуум SDLC от линейного к гибкому — это не тонкая прямая линия. В конкретную модель могут быть включены элементы различных подходов; например, инкрементальная модель жизненного цикла разработки программного обеспечения может включать в себя последовательные требования к программному обеспечению и этапы проектирования, но допускает значительную гибкость при пересмотре требований к программному обеспечению и архитектуры во время создания программного обеспечения.

2.3 Адаптация программного процесса

[1 , c2.7] [2 , c51]

Предопределенные SDLC, SPLC и отдельные программные процессы часто необходимо адаптировать (или «адаптировать») для лучшего удовлетворения местных потребностей. Организационный контекст, инновации в технологии, размер проекта, критичность продукта, нормативные требования, отраслевая практика и корпоративная культура могут определять необходимые адаптации. Адаптация отдельных программных процессов и моделей жизненного цикла программного обеспечения (разработки и продукта) может заключаться в добавлении дополнительных деталей к программным процессам, действиям, задачам и процедурам для решения критических проблем. Он может состоять в использовании альтернативного набора действий, которые достигают цели и результатов программного процесса. Адаптация может также включать исключение из модели разработки или жизненного цикла продукта программных процессов или действий, которые явно неприменимы к объему выполняемой работы.

2.4 Практические соображения

[2 , стр. 188-190]

На практике программные процессы и действия часто чередуются, перекрываются и применяются одновременно. Модели жизненного цикла программного обеспечения, которые определяют отдельные программные процессы со строго заданными критериями входа и выхода, а также заданными границами и интерфейсами, следует рассматривать как идеализацию, которую необходимо адаптировать для отражения реалий разработки и сопровождения программного обеспечения в рамках организационного контекста и бизнес-среды.

Еще одно практическое соображение: программные процессы (такие как управление конфигурацией, создание и тестирование) могут быть адаптированы для облегчения эксплуатации, поддержки, обслуживания, миграции и вывода из эксплуатации программного обеспечения.

Дополнительные факторы, которые следует учитывать при определении и адаптации модели жизненного цикла программного обеспечения, включают требуемое соответствие стандартам, директивам и политикам; требования клиентов; критичность программного продукта; и организационная зрелость и компетентность. Другие факторы включают характер работы (например, модификация существующего программного обеспечения или разработка нового) и область применения (например, аэрокосмическая промышленность или управление гостиницей).

3 Оценка и улучшение программного процесса

[2 , c188, c194] [3 , c26] [4 , c397, c15]

В этом разделе рассматриваются модели оценки программных процессов, методы оценки программных процессов, модели улучшения программных процессов, а также непрерывные и поэтапные рейтинги процессов. Оценки программных процессов используются для оценки формы и содержания программного процесса, которые могут быть определены стандартизированным набором критериев. В некоторых случаях вместо оценки процесса используются термины «оценка процесса» и «оценка возможностей». Оценки возможностей обычно выполняются приобретателем (или потенциальным приобретателем) или внешним агентом от имени приобретателя (или потенциального приобретателя). Результаты используются в качестве индикатора того, приемлемы ли для покупателя программные процессы, используемые поставщиком (или потенциальным поставщиком).

Оценки процессов выполняются на уровне целых организаций, организационных единиц внутри организаций и отдельных проектов. Оценка может включать в себя такие вопросы, как определение того, соблюдаются ли критерии входа и выхода программного процесса, рассмотрение факторов риска и управления рисками или выявление извлеченных уроков. Оценка процесса осуществляется с использованием как модели оценки, так и метода оценки. Модель может обеспечить норму сравнительного сравнения проектов внутри организации и между организациями.

Аудит процесса отличается от оценки процесса. Оценки выполняются для определения уровней возможностей или зрелости, а также для выявления программных процессов, которые необходимо улучшить. Аудиты обычно проводятся для подтверждения соответствия политикам и стандартам. Аудиты предоставляют руководству представление о фактических операциях, выполняемых в организации, чтобы можно было принимать точные и значимые решения по вопросам, влияющим на проект разработки, деятельность по обслуживанию или тему, связанную с программным обеспечением.

Факторы успеха для оценки и улучшения процессов разработки программного обеспечения в организациях, занимающихся разработкой программного обеспечения, включают спонсорство руководства, планирование, обучение, опытных и способных лидеров, приверженность команды, управление ожиданиями, использование агентов изменений, а также пилотные проекты и эксперименты с инструментами. Дополнительные факторы включают независимость оценщика и своевременность оценки.

3.1 Модели оценки программных процессов

[2 , c4.5, c4.6] [3 , c26.5] [4 , c44-48]

Модели оценки программных процессов обычно включают критерии оценки программных процессов, которые рассматриваются как передовые методы. Эти методы могут относиться только к процессам разработки программного обеспечения или могут также включать такие темы, как сопровождение программного обеспечения, управление программными проектами, системная инженерия или управление человеческими ресурсами.

3.2 Методы оценки программного процесса

[1 , c.322-331] [3 , c.26.3] [4 , c.44-48, c.16.4][6]

Метод оценки программного процесса может быть качественным или количественным. Качественные оценки основываются на суждениях экспертов; количественные оценки присваивают программным процессам числовые оценки на основе анализа объективных свидетельств, указывающих на достижение целей и результатов определенного программного процесса. Например, количественная оценка процесса проверки программного обеспечения может быть выполнена путем изучения выполненных процедурных шагов и полученных результатов, а также данных об обнаруженных дефектах и времени, необходимом для обнаружения и устранения дефектов, по сравнению с тестированием программного обеспечения.

Типичный метод оценки процесса разработки программного обеспечения включает планирование, установление фактов (путем сбора данных с помощью анкет, интервью и наблюдения за методами работы), сбор и проверку данных о процессе, а также анализ и отчетность. Оценка процесса может основываться на субъективном качественном суждении оценщика или на объективном наличии или отсутствии определенных артефактов, записей и других свидетельств.

Действия, выполняемые во время оценки процесса разработки программного обеспечения, и распределение усилий для действий по оценке различаются в зависимости от цели оценки процесса разработки программного обеспечения. Оценки программных процессов могут проводиться для

разработки рейтингов возможностей, используемых для выработки рекомендаций по улучшению процессов, или могут проводиться для получения рейтинга зрелости процесса, чтобы претендовать на контракт или награду.

Качество результатов оценки зависит от метода оценки процесса разработки программного обеспечения, целостности и качества полученных данных, возможностей и объективности команды по оценке, а также доказательств, изученных в ходе оценки. Цель оценки процесса разработки программного обеспечения — получить представление, которое позволит установить текущее состояние процесса или процессов и обеспечить основу для улучшения процесса; выполнение оценки процесса разработки программного обеспечения путем следования контрольному списку на предмет соответствия без получения информации мало что дает.

3.3 Модели улучшения процессов разработки программного обеспечения

[2 , стр.187-188] [3 , с26.5] [4 , с2.7]

В моделях улучшения процессов разработки программного обеспечения особое внимание уделяется итеративным циклам непрерывного улучшения. Цикл улучшения программного процесса обычно включает в себя подпроцессы измерения, анализа и изменения. Модель «Планируй-Делай-Проверяй-Действуй» — это хорошо известный итеративный подход к улучшению процессов разработки программного обеспечения. Мероприятия по улучшению включают определение желаемых улучшений и установление приоритетов (планирование); внедрение улучшения, включая управление изменениями и обучение (выполнение); оценка улучшения по сравнению с предыдущими или примерными результатами процесса и затратами (проверка); и внесение дальнейших модификаций (действующих). Модель улучшения процессов Plan-Do-Check-Act может применяться, например, для улучшения программных процессов, которые улучшают предотвращение дефектов.

3.4 Рейтинги непрерывных и поэтапных программных процессов

[1 , с.28-34] [3 , с26.5] [4 , с.39-45]

Возможности программных процессов и зрелость программных процессов обычно оцениваются с использованием пяти или шести уровней, чтобы охарактеризовать возможности или зрелость программных процессов, используемых в организации.

Непрерывная рейтинговая система включает присвоение рейтинга каждому интересующему программному процессу; поэтапная рейтинговая система устанавливается путем присвоения одного и того же рейтинга зрелости всем процессам программного обеспечения в рамках определенного уровня процессов. Представление непрерывных и поэтапных уровней процесса представлено в таблице 8.1. Непрерывные модели обычно используют рейтинг уровня 0; постановочные модели обычно этого не делают.

Таблица 8.1. Уровни оценки программного процесса

Уровень	Непрерывное представление уровней возможностей	Поэтапное представление уровней зрелости
0	Неполный	
1	Выполненный	Исходный
2	Удалось	Удалось
3	Определенный	Определенный
4		Количественно управляемый
5		Оптимизация

В таблице 8.1 уровень 0 указывает на то, что программный процесс выполнен не полностью или не может быть выполнен. На уровне 1 выполняется программный процесс (оценка возможностей) или выполняются программные процессы в группе уровня зрелости 1, но на разовой, неформальной основе. На уровне 2 программный процесс (рейтинг возможностей) или процессы на уровне зрелости 2 выполняются таким образом, что обеспечивает менеджменту видимость промежуточных рабочих

продуктов и может осуществлять некоторый контроль над переходами между процессами. На уровне 3 один программный процесс или процессы в группе уровня зрелости 3 плюс процесс или процессы в группе уровня зрелости 2 четко определены (возможно, в организационных политиках и процедурах) и повторяются в разных проектах. Уровень 3 возможностей или зрелости процесса обеспечивает основу для улучшения процесса в организации, поскольку процесс выполняется (или процессы) аналогичным образом. Это позволяет единым образом собирать данные о производительности для нескольких проектов. На уровне зрелости 4 можно применять и использовать количественные показатели для оценки процесса; можно использовать статистический анализ. На уровне зрелости 5 применяются механизмы непрерывного улучшения процессов.

Непрерывные и поэтапные представления могут использоваться для определения порядка, в котором должны быть улучшены программные процессы. В непрерывном представлении различные уровни возможностей для различных программных процессов служат ориентиром для определения порядка, в котором программные процессы будут улучшаться. В поэтапном представлении достижение целей набора программных процессов в пределах уровня зрелости достигается для этого уровня зрелости, что обеспечивает основу для улучшения всех программных процессов на следующем более высоком уровне.

4 Программное измерение

[3 , c26.2] [4 , c18.1.1]

В этом разделе рассматриваются измерения программных процессов и продуктов, качество результатов измерений, информационные модели программного обеспечения и методы измерения программных процессов (см. Измерение в Основах проектирования).

Перед внедрением нового процесса или модификацией текущего процесса необходимо получить результаты измерений для текущей ситуации, чтобы обеспечить основу для сравнения текущей ситуации и новой ситуации. Например, прежде чем внедрять процесс проверки программного обеспечения, необходимо измерить усилия, необходимые для устранения дефектов, обнаруженных в ходе тестирования. После начального пускового периода после введения процесса проверки объединенные усилия по проверке и тестированию можно сравнить с предыдущим объемом усилий, необходимых только для тестирования. Аналогичные соображения применимы, если процесс изменен.

4.1 Программный процесс и измерение продукта

[1 , c6.3, стр.273] [3 , c26.2, стр.638]

Измерение программного процесса и продукта связано с определением эффективности и действенности программного процесса, действия или задачи. Эффективность программного процесса, *действия* или задачи — это отношение фактически потребляемых ресурсов к ожидаемым или желательным ресурсам, которые будут потребляться при выполнении программного процесса, действия или задачи (см. Эффективность в области экономики программной инженерии КА). Усилия (или эквивалентные затраты) являются основной мерой ресурсов для большинства программных процессов, действий и задач; он измеряется в таких единицах, как человеко-часы, человеко-дни, человеко-недели или человеко-месяцы усилий, или в эквивалентных денежных единицах, таких как евро или доллары.

Эффективность — это отношение фактического результата к ожидаемому результату, полученному программным процессом, действием или задачей; например, от фактического количества дефектов, обнаруженных и исправленных во время тестирования программного обеспечения, до ожидаемого количества дефектов, которые необходимо обнаружить и исправить, возможно, на основе исторических данных для аналогичных проектов (см. Эффективность в области экономики программной инженерии КА). Обратите внимание, что измерение эффективности программного процесса требует измерения соответствующих атрибутов продукта; например, измерение дефектов программного обеспечения, обнаруженных и исправленных во время тестирования программного обеспечения.

Необходимо соблюдать осторожность при измерении атрибутов продукта с целью определения эффективности процесса. Например, количество дефектов, обнаруженных и исправленных в ходе тестирования, может не соответствовать ожидаемому количеству дефектов и, таким образом, обеспечить обманчиво низкую оценку эффективности либо потому, что тестируемое программное обеспечение

имеет лучшее, чем обычно, качество, либо, возможно, из-за внедрения недавно представленной вышестоящей программы. Процесс проверки позволил сократить оставшееся количество дефектов в программном обеспечении.

Показатели продукта, которые могут быть важны для определения эффективности программных процессов, включают сложность продукта, общее количество дефектов, плотность дефектов и качество требований, проектной документации и других связанных рабочих продуктов.

Также обратите внимание, что эффективность и результативность являются независимыми понятиями. Эффективный программный процесс может быть неэффективным для достижения желаемого результата программного процесса; например, объем усилий, затрачиваемых на поиск и исправление программных дефектов, может быть очень большим и привести к низкой эффективности по сравнению с ожиданиями.

Эффективный процесс может быть неэффективен для достижения желаемого преобразования входных рабочих продуктов в выходные рабочие продукты; например, невозможность найти и исправить достаточное количество дефектов программного обеспечения в процессе тестирования. Причины низкой эффективности и/или низкой результативности способа выполнения программного процесса, действия или задачи могут включать одну или несколько из следующих проблем: недостаточные входные рабочие продукты, неопытный персонал, отсутствие адекватных инструментов и инфраструктуры, изучение нового процесс, сложный продукт или незнакомая область продукта. На эффективность и результативность выполнения программных процессов также влияют (положительно или отрицательно) такие факторы, как текучесть кадров, связанные с программным обеспечением, изменения в расписании, новый представитель заказчика или новая организационная политика.

В разработке программного обеспечения производительность при выполнении процесса, действия или задачи представляет собой отношение произведенной продукции к потребляемым ресурсам; например, количество обнаруженных и устраненных дефектов программного обеспечения, деленное на человеко-часы усилий (см. Производительность в области экономики программной инженерии KA). Точное измерение производительности должно включать общие усилия, затраченные на выполнение критериев выхода программного процесса, действия или задачи; например, усилия, необходимые для исправления дефектов, обнаруженных во время тестирования программного обеспечения, должны быть включены в производительность разработки программного обеспечения.

Расчет производительности должен учитывать контекст, в котором выполняется работа. Например, усилия по исправлению обнаруженных дефектов будут включены в расчет производительности команды разработчиков программного обеспечения, если члены команды исправят обнаруженные ими дефекты — как при модульном тестировании разработчиками программного обеспечения или в межфункциональной гибкой команде. Или расчет производительности может включать либо усилия разработчиков программного обеспечения, либо усилия независимой группы тестирования, в зависимости от того, кто устраняет дефекты, обнаруженные независимыми тестировщиками. Обратите внимание, что этот пример относится к усилиям команд разработчиков или групп тестировщиков, а не отдельных лиц. Производительность программного обеспечения, рассчитанная на уровне отдельных лиц, может ввести в заблуждение из-за множества факторов, влияющих на индивидуальную производительность инженеров-программистов.

Стандартизированные определения и правила подсчета для измерения программных процессов и рабочих продуктов необходимы для предоставления стандартизированных результатов измерений по проектам внутри организации, для наполнения репозитория исторических данных, которые можно анализировать для выявления программных процессов, которые необходимо улучшить, и для построения прогнозные модели на основе накопленных данных. В приведенном выше примере определения дефектов программного обеспечения и человеко-часов усилий по тестированию, а также правила подсчета дефектов и усилий потребуются для получения удовлетворительных результатов измерения.

Важна степень институционализации программного процесса; Неспособность институционализировать программный процесс может объяснить, почему «хорошие» программные процессы не всегда дают ожидаемые результаты. Программные процессы могут быть институционализированы путем принятия в рамках местного организационного подразделения или более крупных подразделений предприятия.

4.2 Качество результатов измерений

[4 , c3.4-3.7]

Качество результатов измерения процессов и продуктов в первую очередь определяется надежностью и достоверностью результатов измерений. Измерения, которые не удовлетворяют этим критериям качества, могут привести к неверным интерпретациям и ошибочным инициативам по улучшению программных процессов. Другие желательные свойства программных измерений включают простоту сбора, анализа и представления, а также сильную корреляцию между причиной и следствием.

Тема Измерение программной инженерии в КА Управления программной инженерии описывает процесс реализации программы измерения программного обеспечения.

4.3 Информационные модели программного обеспечения

[1 , c.310-311] [3 , c.712-713] [4 , c.19.2]

Информационные модели программного обеспечения позволяют моделировать, анализировать и прогнозировать атрибуты программных процессов и программных продуктов, чтобы давать ответы на соответствующие вопросы и достигать целей улучшения процессов и продуктов. Необходимые данные могут быть собраны и сохранены в репозитории; данные могут быть проанализированы и модели могут быть построены. Проверка и уточнение информационных моделей программного обеспечения происходят во время проектов программного обеспечения и после завершения проектов, чтобы гарантировать, что уровень точности достаточен, а их ограничения известны и понятны. Информационные модели программного обеспечения также могут быть разработаны для контекстов, отличных от программных проектов; например, информационная модель программного обеспечения может быть разработана для процессов, которые применяются во всей организации,

Построение информационной модели программного обеспечения на основе анализа включает в себя разработку, калибровку и оценку модели. Информационная модель программного обеспечения разрабатывается путем установления гипотетического преобразования входных переменных в желаемые выходные данные; например, размер и сложность продукта могут быть преобразованы в предполагаемые усилия, необходимые для разработки программного продукта, с использованием уравнения регрессии, полученного на основе наблюдаемых данных из прошлых проектов. Модель калибруется путем настройки параметров модели в соответствии с наблюдаемыми результатами прошлых проектов; например, показатель степени в модели нелинейной регрессии может быть изменен путем применения уравнения регрессии к другому набору прошлых проектов, отличных от проектов, использованных для разработки модели. Модель оценивается путем сравнения вычисленных результатов с фактическими результатами для другого набора аналогичных данных.

1. результаты, рассчитанные для другого набора данных, сильно отличаются от фактических результатов для этого набора данных, и в этом случае производная модель неприменима для нового набора данных и не должна применяться для анализа или прогнозирования.

для будущих проектов;

1. результаты, рассчитанные для нового набора данных, близки к фактическим результатам для этого набора данных, и в этом случае в параметры модели вносятся незначительные корректировки для улучшения согласованности;
2. результаты, рассчитанные для нового набора данных и последующих наборов данных, очень близки, и никаких корректировок модели не требуется.

Непрерывная оценка модели может указывать на необходимость корректировок с течением времени по мере изменения контекста, в котором применяется модель.

Метод Цели/Вопросы/Метрики (GQM) изначально предназначался для проведения измерений, но его также можно использовать для проведения анализа и улучшения программных процессов.

Его можно использовать для руководства построением информационной модели программного обеспечения на основе анализа; результаты, полученные с помощью информационной модели программного обеспечения, можно использовать для улучшения процессов.

Следующий пример иллюстрирует применение метода GQM:

- Цель: сократить среднее время обработки запроса на изменение на 10 % в течение шести месяцев.
- Вопрос 1-1: Каково базовое время обработки запроса на изменение?

- Метрика 1-1-1: Среднее время обработки запроса на изменение на дату начала
- Метрика 1-1-2: стандартное отклонение времени обработки запроса на изменение на дату начала
- Вопрос 1-2: Каково текущее время обработки запроса на изменение?
- Метрика 1-2-1: Среднее время обработки запроса на изменение в настоящее время
- Метрика 1-2-2: стандартное отклонение времени обработки запроса на изменение в настоящее время

4.4 Методы измерения программных процессов

[1 , c8]

Методы измерения программных процессов используются для сбора данных о процессах и данных о рабочих продуктах, преобразования данных в полезную информацию и анализа информации для определения операций процесса, которые являются кандидатами на улучшение. В некоторых случаях могут потребоваться новые программные процессы.

Методы измерения процессов также предоставляют информацию, необходимую для измерения результатов инициатив по улучшению процессов. Методы измерения процессов могут использоваться для сбора как количественных, так и качественных данных.

4.4.1 Количественные методы измерения процессов

[4 , c5.1, c5.7, c9.8]

Цель методов количественного измерения процессов состоит в сборе, преобразовании и анализе количественных данных о процессах и рабочих продуктах, которые можно использовать для определения областей, в которых необходимы улучшения процессов, и для оценки результатов инициатив по улучшению процессов. Количественные методы измерения процессов используются для сбора и анализа данных в числовой форме, к которым могут быть применены математические и статистические методы.

Количественные данные процесса могут собираться как побочный продукт программных процессов. Например, количество дефектов, обнаруженных во время тестирования программного обеспечения, и количество затраченных человеко-часов могут быть собраны путем прямого измерения, а производительность обнаружения дефектов может быть получена путем подсчета дефектов, обнаруженных в расчете на один человеко-час. Базовые инструменты контроля качества могут использоваться для анализа количественных данных измерения процесса (например, контрольные листы, диаграммы Парето, гистограммы, диаграммы рассеяния, рабочие диаграммы, контрольные диаграммы и диаграммы причинно-следственных связей) (см. Фонды КА). Кроме того, можно использовать различные статистические методы, начиная от расчета медиан и средних и заканчивая методами многомерного анализа (см. Статистический анализ в Основах инженерии). Данные, собранные с использованием количественных методов измерения процессов, также могут использоваться в качестве входных данных для имитационных моделей (см. Моделирование, прототипирование и моделирование в Основах инженерного дела КА); эти модели можно использовать для оценки влияния различных подходов к улучшению процесса разработки программного обеспечения.

Классификация ортогональных дефектов (ODC) может использоваться для анализа количественных данных измерений процесса. ODC можно использовать для группировки обнаруженных дефектов по категориям и связывания дефектов в каждой категории с программным процессом или программными процессами, в которых возникла группа дефектов (см. Характеристика дефектов в КА качества программного обеспечения). Дефекты интерфейса программного обеспечения, например, могут возникнуть в результате неадекватного процесса разработки программного обеспечения; улучшение процесса проектирования программного обеспечения уменьшит количество дефектов интерфейса программного обеспечения. ODC может предоставить количественные данные для применения анализа первопричин.

Статистический контроль процесса можно использовать для отслеживания стабильности или отсутствия стабильности процесса с помощью контрольных диаграмм.

4.4.2 Качественные методы измерения процессов

[1 , c6.4]

Методы качественного измерения процессов, включая интервью, анкетирование и экспертную оценку, можно использовать для усиления методов количественного измерения процессов. Методы группового консенсуса, включая метод Дельфи, могут использоваться для достижения консенсуса среди групп заинтересованных сторон.

5 инструментов процесса разработки программного обеспечения

[1 , c8.7]

Инструменты программных процессов поддерживают многие нотации, используемые для определения, реализации и управления отдельными программными процессами и моделями жизненного цикла программного обеспечения. Они включают редакторы для таких нотаций, как диаграммы потоков данных, диаграммы состояний, диаграммы BPMN, IDEF0, сети Петри и диаграммы деятельности UML. В некоторых случаях программные средства обработки позволяют проводить различные виды анализа и моделирования (например, моделирование дискретных событий). Кроме того, могут быть полезны бизнес-инструменты общего назначения, такие как электронные таблицы.

Инструменты автоматизированной разработки программного обеспечения (CASE) могут усилить использование интегрированных процессов, поддержать выполнение определений процессов и предоставить людям рекомендации по выполнению четко определенных процессов. Простые инструменты, такие как текстовые процессоры и электронные таблицы, можно использовать для подготовки текстовых описаний процессов, действий и задач; эти инструменты также поддерживают прослеживаемость входных и выходных данных нескольких программных процессов (таких как анализ потребностей заинтересованных сторон, спецификация требований к программному обеспечению, архитектура программного обеспечения и детальное проектирование программного обеспечения), а также результатов программных процессов, таких как документация, программные компоненты, тестовые сценарии. и отчеты о проблемах.

Большинство областей знаний в этом Руководстве описывают специализированные инструменты, которые можно использовать для управления процессами в рамках этого KA. В частности, см. KA по управлению конфигурацией программного обеспечения для обсуждения инструментов управления конфигурацией программного обеспечения, которые можно использовать для управления процессами создания, интеграции и выпуска программных продуктов. Другие инструменты, например, для управления требованиями и тестирования, описаны в соответствующих KA.

Программные средства обработки процессов могут поддерживать проекты, в которых участвуют географически рассредоточенные (виртуальные) команды. Все чаще инструменты обработки программного обеспечения доступны через средства облачных вычислений, а также через специализированные инфраструктуры.

Панель управления проектом или информационная панель могут отображать выбранные атрибуты процессов и продуктов для программных проектов и указывать измерения, которые находятся в контрольных пределах, и те, которые требуют корректирующих действий.

ДАЛЬНЕЙШИЕ ЧТЕНИЯ

Программное дополнение к Руководству по Своду знаний по управлению проектами® *(SWX)* [5].

SWX обеспечивает адаптации и расширения общих практик управления проектами, описанных в Руководстве PMBOK® по управлению программными проектами. Основным вкладом этого расширения в Руководство PMBOK® является описание процессов, применимых для управления адаптивными программными проектами жизненного цикла.

Д. Гибсон, Д. Голденсон и К. Кост, «Результаты улучшения процессов на основе CMMI» [6].

В этом техническом отчете обобщаются общедоступные эмпирические данные о результатах производительности, которые могут быть получены в результате улучшения процессов на основе CMMI. Отчет содержит серию кратких описаний случаев, которые были созданы в сотрудничестве с представителями 10 организаций, которые добились заметных количественных результатов деятельности благодаря своим усилиям по улучшению на основе CMMI.

CMMI® для разработки, версия 1.3 [7].

CMMI® for Development, версия 1.3, содержит интегрированный набор руководящих принципов для разработки и улучшения продуктов и услуг. Эти рекомендации включают передовой опыт разработки и улучшения продуктов и услуг для удовлетворения потребностей клиентов и конечных пользователей.

ISO/IEC 15504-1:2004 Информационные технологии. Оценка процессов. Часть 1. Понятия и словарь *[8].*

Этот стандарт, широко известный как SPICE (Улучшение процесса разработки программного обеспечения и определение возможностей), состоит из нескольких частей. Часть 1 содержит концепции и словарь для процессов разработки программного обеспечения и связанных с ними функций управления бизнесом. Другие части стандарта 15504 определяют требования и процедуры для проведения оценки процессов.

ИСПОЛЬЗОВАННАЯ ЛИТЕРАТУРА

[1] RE Fairley, *Управление и руководство программными проектами* , Wiley-IEEE Computer Society Press, 2009.

[2] Дж. В. Мур, *Дорожная карта разработки программного обеспечения: руководство* , основанное на стандартах, издательство Wiley-IEEE Computer Society Press, 2006.

[3] И. Соммервиль, *Разработка программного обеспечения* , 9-е изд., Addison-Wesley, 2011.

[4] С. Х. Кан, *Метрики и модели в разработке качества программного обеспечения* , 2-е изд., Addison-Wesley, 2002.

[5] Институт управления проектами и компьютерное общество IEEE, *Расширение программного обеспечения к руководству PMBOK®, пятое издание, изд.:* Институт управления проектами, 2013 г.

[6] Д. Гибсон, Д. Голденсон и К. Кост, «Результаты улучшения процессов на основе CMMI», Институт программной инженерии, 2006 г., http://resources.sei.cmu.edu/library/asset-view .cfm?assetID=8065 ..

[7] Команда разработчиков CMMI, «CMMI для разработки, версия 1.3», Институт программной инженерии, 2010 г., http://resources.sei.cmu.edu/library/asset-view.cfm?assetID=9661 .

[8] *ISO/IEC 15504-1:2004 Информационные* технологии — Оценка процессов — Часть 1: Понятия и словарь *, ISO/IEC, 2004.*

- Последнее изменение этой страницы: 25 августа 2015 г., 17:14.