Программная инженерия

Сопровождение программного обеспечения (Software Maintenance)

Глава базируется на IEEE Guide to the Software Engineering Body of Knowledge[(1)] - SWEBOK®, 2004. Содержит перевод описания области знаний SWEBOK® “Software Maintenance”, с комментариями и замечаниями[(2)].

Сергей Орлик.

Программная инженерия

Сопровождение программного обеспечения (Software Maintenance)

Результат усилий по разработке программного обеспечения состоит в передачи в эксплуатацию программного продукта, удовлетворяющего требованиям пользователей. Соответственно, в процессе эксплуатации продукт будет изменяться или эволюционировать. Связано это с обнаружением при реальном использовании скрытых дефектов, изменениями в операционном окружении, необходимостью покрытия новых требований и т.п.

Фаза сопровождения в жизненном цикле, обычно, начинается сразу после приемки/передачи продукта и действует в течение периода гарантии или, чаще, технической поддержки. Однако, сама деятельность, связанная с сопровождением, начинается намного раньше.

Сопровождение программного обеспечения является составной частью жизненного цикла. К сожалению, так сложилось, что вопросам сопровождения уделяется существенно меньше внимания, чем другим фазам жизненного цикла. Исторически, в большинстве организаций, разработка программных систем явно в фаворе, по сравнению с деятельностью по сопровождению. Однако, такая ситуация постепенно начинает меняться (достаточно, по мнению автора, хотя бы взглянуть на частоту упоминаний ITIL* – IT Infrastructure Library, уделяющей особое внимание вопросам поддержки и сопровождения инфраструктуры информационных технологий). В большой степени, как отмечает SWEBOK, это связано с сокращением инвестиций организаций непосредственно в разработку программных систем, с целью увеличения сроков использования уже существующего и применяемого ПО. Конечно, с точки зрения автора, это не единственная причина. Скорее вопросы постоянно изменяющихся бизнес-потребностей, динамика бизнеса и желание повысить отдачу от уже эксплуатируемых систем приводит к усилению роли поддержки и сопровождения программного обеспечения и естественной интеграции такой деятельности в бизнес-процессы подразделений информационных технологий.

* ITIL, в частности, определяет три аспекта управления жизненным циклом приложений – *определение требований*, *проектирование и разработку*, и, наконец, *сопровождение*. Все это, в контексте программного обеспечения, относится к деятельности по управлению приложениями – Application Management в ITIL ICT Infrastructure Management (ICT - Information and Communications Technology).

mailto:sorlik@borland.ru
http://sorlik.blogspot.com

С точки зрения автора, если проблема 2000 года оказала особое влияние на изменение отношения к сопровождению на Западе, то расширение применения продуктов Open Source во всем мире и связанная с ним волна надежд на получение дешевого решения существующих задач привела к тому, что вопросы сопровождения вышли для многих организаций на первый план. Ситуация во многих ИТ-подразделениях показывает, что такие надежды оправдались только частично. Использование продуктов Open Source не стало дешевой альтернативой и, в ряде случаев, привело даже к большим проектным затратам именно в силу недостаточно проработанной политики эксплуатации и сопровождения построенных на их основе прикладных решений. Это ни в коем случае не значит, что волна Open Source “захлебнулась”. Это означает только, что игнорирование оценки стоимости сопровождения привело к превышению бюджетов, недостатку ресурсов и, в конце концов, частому провалу инициатив по использованию таких продуктов в корпоративной среде. Неготовность рассматривать жизненный цикл во времени как продолжающийся и после передачи системы в эксплуатацию, непроработанность соответствующих процедур корректировки продукта после его выпуска – основная беда и в бизнесе-среде, для которой программное обеспечение лишь инструмент, и в компаниях-интеграторах, “забывающих” о необходимости развития успеха после внедрения системы у заказчика, и у независимых поставщиков программных продуктов, которые, выпуская новую версию лучшего в своем классе продукта, начинают работать над новой версией, уделяя недостаточное внимание поддержке и обновлению уже существующих версий.

Сопровождение программного обеспечения в SWEBOK определяется как вся совокупность деятельности, необходимой для обеспечения эффективной (с точки зрения затрат) поддержки программных систем. Эти работы выполняются как перед вводом системы в эксплуатацию, так и после этого. Предварительные работы включают планирование деятельности по сопровождению системы, а также организацию перехода к ее полнофункциональному использованию. Если новая система должна заменить старую систему, предназначенную для решения тех же задач, просто на новом уровне эффективности, стоимости использования, новых функциональных возможностей, в этом случае важно обеспечить плавный переход со старой системы на новую, максимально естественный для пользователей. С этим связано не только планирование, например, переноса информации, хранимой в соответствующих базах данных, но и обучение пользователей, подготовка, настройка и проверка “боевой” конфигурации, определение последовательности операций, организация и обучение службы поддержки (help-desk) и т.п.

Область знаний “Сопровождение программного обеспечения” связана с другими аспектами программной инженерии. По-сути, описание этой области знаний непосредственно пересекается со всеми другими дисциплинами.

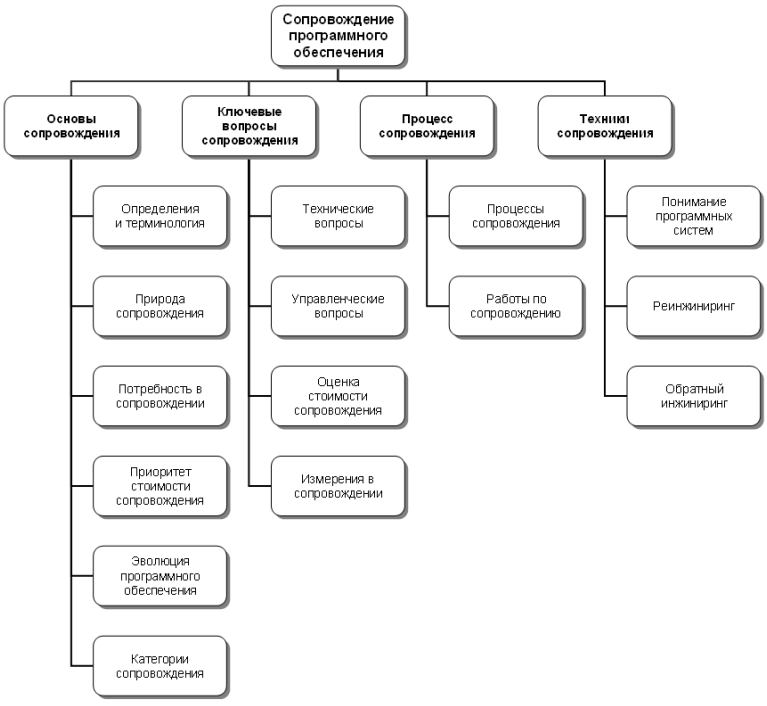

Рисунок 6. Область знаний “Сопровождение программного обеспечения” [SWEBOK, 2004, с.6-2, рис. 1]

mailto:sorlik@borland.ru
http://sorlik.blogspot.com

1. Основы сопровождения программного обеспечения (Software Maintenance Fundamentals)

Эта секция включает концепции и терминологию, формирующие основы понимания роли и содержания работ по сопровождению программных систем. Темы данной секции предоставляют соответствующие определения и описывают, почему именно существует потребность в сопровождении. Категории сопровождения критически важны для понимания сути обсуждаемых вопросов.

1.1 Определения и терминология (Definitions and Terminology)

Сопровождение программного обеспечения определяется стандартом IEEE Standard for Software Maintenance (IEEE 1219) как модификация программного продукта после передачи в эксплуатацию для устранения сбоев, улучшения показателей производительности и/или других характеристик (атрибутов) продукта, или адаптации продукта для использования в модифицированном окружении. Интересно, что данный стандарт также касается вопросов подготовки к сопровождению до передачи системы в эксплуатацию, однако, структурно это сделано на уровне соответствующего информационного приложения, включенного в стандарт.

В свою очередь, стандарт жизненного цикла 12207 (IEEE, ISO/IEC, ГОСТ Р ИСО/МЭК) позиционирует сопровождение как один из главных процессов жизненного цикла. Этот стандарт описывает сопровождение как процесс модификации программного продукта в части его кода и документации для решения возникающих проблем <при эксплуатации> или реализации потребностей в улучшениях <тех или иных характеристик продукта>. Задача состоит в модификации продукта при условии сохранения его целостности. Международный стандарт ISO/IEC 14764 (Standard for Software Engineering - Software Maintenance) определяет сопровождение программного обеспечения в тех же терминах, что и стандарт 12207, придавая особое значение работам по подготовке к деятельности по сопровождению до передачи системы в реальную эксплуатацию, например, вопросам планирования регламентов и операций по сопровождению.

1.2 Природа сопровождения (Nature of Maintenance)

Сопровождение поддерживает функционирование программного продукта на протяжении всего операционного жизненного цикла, то есть периода его эксплуатации. В процессе сопровождения фиксируются и отслеживаются запросы на модификацию (также называемые “запросами на изменения” – change requests, в частности, в контексте конфигурационного управления), оценивается влияние предлагаемых изменений, модифицируется код и другие активы (артефакты) продукта, проводится необходимое тестирование и, наконец, выпускается обновленная версия продукта. Кроме того, проводится обучение пользователей и обеспечивается их ежедневная поддержка при работе с текущей версией продукта. В SWEBOK отмечается, что сопровождение, с точки зрения операций отслеживания и контроля, обладает большим содержанием, чем разработка (в общем понимании). С точки зрения автора, объем и активность операций по контролю разработки в большой степени зависит от сложившихся практик, внутрикорпоративных регламентов и требований, а также применяемых методологий и концепции управления (в частности – проектного менеджмента). Так или иначе, *отслеживание и контроль – ключевые элементы деятельности по сопровождению программного обеспечения* (как и других ИТ-активов предприятия).

Стандарт 12207 определяет понятие “maintainer” - в соответствующем ГОСТ он именуется как “персонал сопровождения”, подразумевая организацию, выполняющая работы по сопровождению. SWEBOK использует данный термин, также, и в отношении лиц (individuals), проводящих определенные работы по сопровождению, в отличие, например, от разработчиков, занимающихся реализацией системы в программном коде.

Стандарт жизненного цикла 12207 также идентифицирует основные работы по сопровождению: реализация процесса <сопровождения>, анализ проблем и модификаций (изменений), реализаций модификаций, обзор (оценка)/принятие <решений> по сопровождению, миграция (с одной версии программного продукта на другую, с одного продукта на другой) и вывод системы из эксплуатации. Эти работы описываются далее в теме 3.2 “Работы по сопровождению” (Maintenance Activities).

Специалисты по сопровождению (персонал сопровождения) могут получать знания о программном продукте непосредственно от разработчиков. Взаимодействие с разработчиками и раннее

mailto:sorlik@borland.ru
http://sorlik.blogspot.com

привлечение этих специалистов помогает уменьшить усилия, необходимые для адекватного сопровождения программной системы. По мнению автора, передача знаний персоналу сопровождения, его обучение, должно начинаться не позднее начала опытной эксплуатации продукта. В противном случае, усилия на одновременную поддержку прикладной системы и обучение соответствующих специалистов не только превысит реально допустимые нормы загрузки персонала (как группы или службы сопровождения и техподдержки, так и разработчиков системы), но снизит эффективность поддержки пользователей на критически важном этапе первоначального использования новой системы. По опыту автора и результатам обсуждения этого вопроса с сотрудниками внутрикорпоративных ИТ-департаментов, обычно, в зависимости от сложности системы, пик нагрузки на службу сопровождения приходится в течении первых 2 - 6 недель, с момента передачи системы в реальную эксплуатацию, тем более, при отказе от использования старой системы или ее предыдущей версии. SWEBOK отмечает, что, в некоторых случаях, инженеры (создававшие систему) не могут быть привлечены к обучению и поддержке персонала сопровождения. Особенно часто это касается тиражируемых или “коробочных” систем. Это создает дополнительные трудности для специалистов, обеспечивающих сопровождение. В то же время, инженеры, занимающиеся технической поддержкой (несколько боле узкий круг в команде сопровождения, включающей менеджеров, администраторов и других специалистов), должны (в зависимости от типа продукта) иметь доступ к активам проекта (например, описанию его внутренней архитектуры), включая код, документацию и т.п. Именно таким образом начинает формироваться информационная инфраструктура службы технической поддержки и сопровождения у производителей программных продуктов.

Практика показывает, что инженеры по технической поддержке производителя программного обеспечения (не только “коробочного”, но и создаваемого и настраиваемого интеграторами, обладающими собственными программными решениями) должны не просто иметь доступ ко всем ключевым активам проекта (код, документация, спецификации требований, внутренние модели и т.п.), но в их обязанности входит создание “патчей” (patch – “заплата”), исправлений ошибок и, в особых случаях, такие изменения, до выпуска новой версии продукта, создаются с привлечением непосредственно разработчиков продукта (групп и подразделений R&D – Research and Development). При этом, разработчики продукта информируются о найденных ошибках и, в случае нахождения соответствующих решений специалистами технической поддержки, такие решения передаются разработчикам с тем, чтобы те либо включили такие изменения в новую версию программного продукта (безусловно, в случае успешного прохождения всех необходимых тестов), либо нашли более адекватное решение в контексте новой функциональности либо тех изменений, которые включены в новую версию продукта. В обязанности инженеров службы сопровождения, в общем случае, входит: проверка пользовательского сценария, приводящего к сбою; идентификация причин сбоя, т.е локализация ошибки/причин ее появления; предоставление соответствующих исправлений или, при невозможности создания таковых на данном этапе либо в заданные сроки – предоставление обходного пути решения проблемы для достижения требуемых бизнес-задач (такие обходные пути, обычно, называют “workaround”); журналирование всех работ и операций; помещение описания проблемы и ее решения в базу знаний службы сопровождения; передача всей информации разработчикам; своевременное информирование пользователя о статусе запроса и некоторые другие работы, содержание которых может варьироваться, в зависимости от регламентов и корпоративных стандартов в конкретной организации, либо параметров контракта на сопровождение и техническую поддержку, если таковой есть.

1.3 Потребность в сопровождении (Need for Maintenance)

Сопровождение необходимо для обеспечения того, чтобы программный продукт на протяжении всего периода эксплуатации удовлетворяет требованиям пользователей. Деятельность по сопровождению применима для программного обеспечения, созданного с использованием любой модели жизненного цикла и методологии разработки. На первый взгляд, это утверждение SWEBOK может показаться тривиальным. Однако, обратитесь к своему опыту разработки и использования различного программного обеспечения. Наверняка, Вы найдете случаи из собственной практики или практики коллег, когда столь очевидное утверждение хорошо бы донести до разработчика того или иного программного продукта. Изменения программной системы могут быть обусловлены как действиями по корректировке ее поведения или несвязанные с необходимостью корректировки (подразумевая уже не исправление ошибок, а, например, повышение производительности или расширение функциональности).

mailto:sorlik@borland.ru
http://sorlik.blogspot.com

В общем случае, работы по сопровождению должны проводиться для решения следующих задач:

- устранение сбоев
- улучшение дизайна
- реализация расширений <функциональных возможностей>
- создание интерфейсов взаимодействия с другими (внешними) системами
- адаптация (например, портирование) для возможности работы на другой аппаратной платформе (или обновленной платформе), применения новых системных возможностей, функционирования в среде обновленной телекоммуникационной инфраструктуры и т.п.
- миграции унаследованного (legacy) программного обеспечения
- вывода программного обеспечения из эксплуатации

Деятельность персонала сопровождения включает четыре ключевых аспекта:
поддержка контроля (управляемости) программного обеспечения в течение всего цикла эксплуатации:

- поддержка модификаций программных систем
- совершенствование существующих функций*
- предотвращение падения производительности программной системы до неприемлемого уровня

Автор убежден, что говоря о предотвращении деградации производительности, мы должны понимать, что это, при всем желании совершенствования системы, может делаться и за счет обновления мощности аппаратной части и/или соответствующей телекоммуникационной инфраструктуры, если это более обосновано, чем модификация самой программной системы. На самом деле это вопрос того, что окажется дешевле (и менее рискованно), т.е. связано с затратами/ стоимостью соответствующих работ, оборудования и поддержки обновленного системного окружения (что, к сожалению, часто также не учитывается даже при более-менее сложившейся практике сопровождения).

* судя по всему, SWEBOK в данном случае подразумевает функции не программного обеспечения, а процессов сопровождения и функции персонала сопровождения, как таковые.

1.4 Приоритет стоимости сопровождения (Majority of Maintenance Costs)

Работы по сопровождению потребляют если не большую (как отмечает SWEBOK), то значительную часть финансовых ресурсов жизненного цикла программного обеспечения. Общее понимание сопровождения подразумевает лишь устранение сбоев. Однако, исследования и опросы на протяжении многих лет показывают, что более 80% усилий по сопровождению связаны не столько устранением сбоев, сколько с другими работами, не связанными с исправлением дефектов. Многие менеджеры по сопровождению объединяют в отчетности вопросы расширения функциональности и исправления ошибок в поддерживаемых программных системах. Такое смешение качественно различных работ приводит к неправильному представлению о реальной, на самом деле, не столь высокой стоимости сопровождения в части устранения дефектов. Понимание различных категорий работ в рамках деятельности по сопровождению помогает понять структуру реальных затрат. Кроме того, понимание факторов, влияющих на возможности сопровождения системы, помогают не только сохранять необходимый уровень затрат, но и снижать их.

Существуют как технические, так и другие (например, организационные, являющиеся, по мнению автора, наиболее сильно влияющими на объем затрат) факторы, оказывающие влияние на стоимость сопровождения, в целом:

- тип приложения
- новизна программного обеспечения
- наличие и квалификация персонала по сопровождению
- длительность использования программной системы
- характеристики и специфика аппаратной части (а также телекоммуникационной инфраструктуры)
- качество дизайна (например, модульность или масштабируемость), кода, документации и соответствующих работ по тестированию системы

mailto:sorlik@borland.ru
http://sorlik.blogspot.com

1.5 Эволюция программного обеспечения (Evolution of Software)

В 1969 году Леман (см. рекомендуемую литературу к данной секции SWEBOK) впервые связал деятельность по сопровождению и вопросы эволюции программного обеспечения. Результаты более чем 20-ти летних исследований во главе с Леманом привели к формулированию ряда важных положений, ключевое из которых утверждает, что деятельность по сопровождению, по-сути, представляет собой эволюционную разработку программных систем. Принятию тех или иных решений в процессе сопровождения, помогает понимание того, что происходит с программной системой в процессе ее эксплуатации. Существующее (особенно, корпоративное) программное обеспечение никогда не бывает полностью завершенным и продолжает эволюционировать в течение всего срока эксплуатации. В процессе эволюционирования, программная система становится все более сложной до тех пор, пока не предпринимаются специальные усилия (в том числе, в рамках специального проекта по модификации) по уменьшению его сложности.

В то же самое время, если можно выделить тенденции развития программной системы и ее поведение достаточно стабильно, его эволюционирование можно измерить. Последние годы делаются попытки разработать соответствующие модели оценки усилий по сопровождению. В результате уже создаются определенные средства (численные и инструментальные) управления работами по сопровождению, ряд из которых приводится в ссылках к данной секции SWEBOK.

1.6 Категории сопровождения (Categories of Maintenance)

Многие источники, в частности, стандарт IEEE 1216, определяют три категории работ по сопровождению: корректировка, адаптация и совершенствование. Такая классификация была обновлена в стандарте ISO/IEC 14764 Standard for Software Engineering - Software Maintenance введением четвертой составляющей. Таким образом, сегодня говорят о четырех категориях сопровождения:

- Корректирующее сопровождение (corrective maintenance): “реактивная” модификация программного продукта, выполняемая уже после передачи в эксплуатацию для устранения сбоев;
- Адаптирующее сопровождение (adaptive maintenance): модификация программного продукта на этапе эксплуатации для обеспечения продолжения его использования с заданной эффективностью (с точки зрения удовлетворения потребностей пользователей) в изменившемся или находящемся в процессе изменения окружении; в первую очередь, подразумевается изменение бизнес-окружения, порождающее новые требования к системе;
- Совершенствующее сопровождение (perfective maintenance): модификация программного продукта на этапе эксплуатации для повышения характеристик производительности и удобства сопровождения;
- Профилактическое сопровождение (preventive maintenance): модификация программного продукта на этапе эксплуатации для идентификации и предотвращения скрытых дефектов до того, когда они приведут к реальным сбоям.

ISO/IEC 14764 (Standard for Software Engineering - Software Maintenance) классифицирует адаптивное и совершенствующее сопровождение как работы по расширению <функциональности> продукта. Этот стандарт также объединяет корректирующую и профилактическую деятельность в общую категорию работ по корректировке системы. Профилактическое сопровождение (новейшая категория работ по сопровождению) наиболее часто проводится для программных систем, связанных с вопросами безопасности <людей>.

	Корректирующие работы	Работы по расширение
“Проактивный” подход	Профилактическое сопровождение	Совершенствующее сопровождение
“Реактивный” подход	Корректирующее сопровождение	Адаптирующее сопровождение

Таблица 1. Категории сопровождения программного обеспечения.

2. Ключевые вопросы сопровождения программного обеспечения (Key Issues in Software Maintenance)

Для обеспечения эффективного сопровождения программных систем необходимо решать целый комплекс вопросов и проблем, связанных с соответствующими работами. Необходимо понимать, что процесс сопровождения предъявляет уникальные технические и управленческие требования к персоналу, занимающемуся сопровождениям и, в первую очередь, специалистам-инженерам. Попытка найти дефект в продукте, содержащем 500 тысяч строк кода, написанных другими инженерами – яркий пример сложностей, с которыми приходится сталкиваться инженерам по сопровождению. Другой пример, уже организационный, постоянная борьба за ресурсы с разработчиками (с моей точки зрения, это чаще всего проявляется в вопросах отвлечения разработчиков от текущей работы для помощи в решении проблем сопровождения, а также в конкуренции за приоритеты финансирование разработки новой системы или сопровождения существующей). Одновременное планирование перспективной версии системы, реализация следующей версии и подготовка критических патчей для текущей версии – еще один классический пример проблем, с которыми приходится сталкиваться в процессе эксплуатации программного обеспечения.

Данная секция представляет некоторые технические и управленческие вопросы, связанные с сопровождением программных систем. Эти вопросы и проблемы сгруппированы в набор тем:

- Технические вопросы
- Управленческие вопросы
- Оценка стоимости
- Измерения

2.1 Технические вопросы (Technical Issues)

2.1.1 Ограниченное понимание (Limited understanding)

Ограниченное понимание подразумевает как быстро инженер по сопровождению может понять где необходимо внести исправления или изменения в код системы, которую он не разрабатывал. Исследования показывают, что от 40 до 60 процентов усилий по сопровождению тратится на анализ и понимание сопровождаемого программного обеспечения. Формирование целостного взгляда о системе представляет большое значение для инженеров. Этот процесс более сложен в случае анализа текстового представления системы – ее исходного кода, особенно, когда процесс эволюции системы от сборки к сборки, от релиза к релизу, в нем никак не отмечен, не документирован и когда разработчики не могут объяснить историю и структуру изменений, что, к сожалению, случается достаточно часто.

Практика автора показывает, что для объектно-ориентированных программ качественно упрощает задачу понимания кода использование UML-инструментария, способного на основе кода восстановить не только модель классов, но и их взаимодействия в форме диаграмм классов (class diagram), коммуникаций или сотрудничества (collaboration в UML1.x, переименованная в communication в UML 2.0) и, особенно, последовательностей (sequence diagram), демонстрирующая структуру взаимных вызовов во времени. Если соответствующий инструментарий предоставляет одновременную визуализацию кода и диаграммы и обеспечивает взаимную синхронизацию их с точки зрения навигации (выбор метода в любой из представленных диаграмм автоматически позиционирует соответствующим образом редактор кода и, наоборот) – такие средства автоматизации могут качественно сократить время, необходимое для формирования представления о системе, иногда – даже не в разы, а на порядок (конечно, при достаточном уровне знания используемых технологий со стороны инженера по сопровождению). Если к этому добавить документированность (и доступность соответствующих активов –спецификаций, моделей) архитектуры и ключевых технологических решений со стороны разработчиков системы – обсуждаемый вопрос, конечно, не становится тривиальным, однако, превращается во вполне решаемую задачу. Вообще говоря, использование соответствующих средств автоматизации построения моделей по коду (задача обратного инжиниринга – rerverse engineering) является обоснованной практикой изучения любой системы или фреймворка. Я убежден, весь мой опыт показывает, что при достаточной квалификации инженера, формирование общего архитектурного представления о системе (или фреймворке), понимания того, какие технологические и структурные

mailto:sorlik@borland.ru
http://sorlik.blogspot.com

подходы и шаблоны использовались при ее построении, позволяет решать возникающие вопросы корректировки кода и расширения функциональности системы, не нарушая общие принципы ее построения, естественным образом обеспечивая ее эволюцию, без ущерба ее целостности. При таком понимании, даже не заглядывая в код системы или фреймворка, инженер способен с очень большой вероятностью предположить возможные причины сбоя, а, в общем случае, и любых аспектов поведения системы. Тема обратного инжиниринга освещается SWEBOK как самостоятельная техника сопровождения (4.3), однако, автору показалось важным особо акцентировать на ней внимание именно в этой части обсуждения вопросов сопровождения.

2.1.2 Тестирование (Testing)

Стоимость повторения полного набора тестов для основных модулей системы может быть существенным как по времени, так и по стоимости. Для сопровождения системы особо значимым является выборочное регрессионное тестирование (см. область знаний Software Testing, тему 2.2.6 Регрессионное тестирование) системы или его компонент для проверки того, что внесенные изменения для привели к непреднамеренному изменению поведения программного обеспечения. Вопрос состоит в том, что часто сложно найти время для необходимого тестирования. Не меньшей проблемой является и координации в проведении тестов различными членами группы сопровождения, занимающимися решением различных задач. Если же система выполняет критичные <для бизнеса> функции, временный вывод системы из эксплуатации (как говорят, перевод системы в offline) для выполнения тестов часто оказывается просто невозможен.

Таким образом, с точки зрения автора, одним из ключевых вопросов сопровождения является организация работ по тестированию модификаций эксплуатируемых систем, вплоть до предварительного планирования и разработки регламентов, в соответствии с которыми, например, основываясь на оценке критичности запросов на изменения (как дефектов, так и важных расширений – будь то новая функциональность или необходимое расширение интеграционных возможностей), затрагиваемых модулях, персоналом сопровождения будут проводиться стандартные процедуры. К таким процедурам, наравне с журналированием запросов и проводимых работ, могут и, скорее, должны относиться: анализ влияния <изменений> (impact analysis – см. ниже), оценка рисков, тестирование (различными методами, в различном объеме), выпуск предварительных версий патчей/обновлений в ограниченное использование (если это позволяет спецификация системы), использование “клона” системы (развертывание ее на идентичном оборудовании в идентичной конфигурации) и т.п.

2.1.3 Анализ влияния (Impact analysis)

Анализ влияния описывает как проводить (в частности, с точки зрения эффективности затрат) полный анализ возможных последствий и влияний изменений, вносимых в существующую систему. Персонал сопровождения должен обладать необходимыми знаниями о специфике системы (в идеальном случае, иметь полное представление о системе на уровне ее разработчиков) – ее содержании и структуре. Инженеры используют эти знания для выполнения работ по анализу влияния, идентифицируя все системы* и программные продукты, на которые могут повлиять изменения, вносимые в обслуживаемую программную систему. При этом, должны быть определены риски, связанные с внесением обсуждаемых изменений.

* Как мы видим из описания данных работ в SWEBOK, речь идет не только о компонентах системы, но и о ее окружении, включая другие системы, функционирующие в том же операционном/системном окружении.
Запросы на изменения** (change requests - CR), иногда упоминаемые как запросы на модификацию (modification request - MR), часто также называемые отчетами о проблемах (problem report - PR), должны анализироваться и трансформироваться в термины программной системы. Эти шаги выполняются после того, как соответствующий запрос на изменение начинает обрабатываться в рамках *процесса управления изменениями* или, как принято называть, *конфигурационного управления*, и фиксируется в системе конфигурационного управления (см. область знаний Configuration Management).

** Обычно запросы на изменения разделяют на две категории – “пожелания” (suggestions), относящиеся к расширению системы, и “отчеты об ошибках” (defect или bug report), направляемые пользователями в службу сопровождения или инженерами по тестированию разработчикам.

Цели анализа влияния могут быть сформулированы следующим образом:

mailto:sorlik@borland.ru
http://sorlik.blogspot.com

- Определение содержания изменений для задания работ по планированию и реализации
- Получение максимально возможной оценки ресурсов, необходимых для проведения соответствующих работ
- Анализ стоимости и выгоды от внесения запрошенных изменений (обычно касается пожеланий, запросов на расширение системы)
- Обсуждение сложности вопросов, связанных с внесением соответствующих изменений

Сложность решения вопроса, поставленного соответствующим запросом на изменения, часто является основным фактором определения того, когда и как будет решена проблема. Инженеры идентифицирую компоненты, в которые необходимо внести изменения. Обычно рассматривается несколько вариантов решения проблемы и вырабатывается (также, обязательно, фиксируются в соответствующей системе обработки запросов на изменения) наиболее оптимальный путь ее решения.

С точки зрения автора, оптимальность пути не всегда означает наиболее ”красивое” технологическое решение. Иногда это может быть временное решение, может быть даже нарушающее архитектурные шаблоны системы, однако, обоснованное с точки зрения сроков и стоимости его реализации. В то же самое время, результаты анализа направляются разработчикам системы, обычно работающим над следующей версией, для включения соответствующего изменения уже в рамках принятого стиля кодирования, соглашений, архитектурных шаблонов и т.п. Безусловно, такой путь многим может показаться просто неэтичным, с точки зрения “настоящего” инженерного подхода. Однако, если разработчики готовят следующую версию системы, затрагивая модуль, модифицируемый службой сопровождения, с точки зрения бизнес-решений, “некрасивый”, но быстрый путь достижения требуемого поведения системы, в большинстве случаев, будет выглядеть более обоснованным, чем принятие на себя персоналом сопровождения функций разработчиков системы. Иногда, если требуемое изменение не столь критично, чтобы решение было предоставлено “вчера” (хотя пользователи, практически всегда, именно так характеризуют свои запросы в терминах приоритета), логичным выглядит откладывание проведения соответствующих модификаций и передача этих работ непосредственно разработчикам. Как это часто можно услышать – “будет доступно в следующем релизе”. Ничего не напоминает? Но, экономически, это часто бывает более чем оправдано.

Если программное обеспечение изначально разрабатывалось с учетом дальнейшей поддержки, это может существенно облегчить анализ влияний, как одной из ключевых работ по сопровождению.

2.1.4 Возможность сопровождения (Maintainability)

Возможность сопровождения или сопровождаемость программной системы определяется, например, глоссарием IEEE (стандарт 610.12-90 Standard Glossary for Software Engineering Terminology, обновление 2002 года) как легкость сопровождения, расширения, адаптации и корректировки для удовлетворения заданных требований. Стандарт ISO/IEC 9126-01 (Software Engineering – Product Quality – Part 1: Quality Model, 2001 г.) определяет возможность сопровождения как одну из характеристик качества.

Для уменьшения стоимости дальнейшего сопровождения, на протяжении всего процесса разработки необходимо специфицировать, оценивать и контролировать характеристики, влияющие на возможность сопровождения. Если такие работы проводятся регулярно, это облегчает дальнейшее сопровождение, повышая его сопровождаемость (в частности, как характеристику качества). Часто этого сложно добиться, потому, что, к сожалению, такого рода характеристики игнорируются при разработки. Разработчики заняты другими запланированными работами и также часто пренебрегают требованиями, предъявляемыми к сопровождаемости системы.

Одной из ключевых проблем сопровождения является отсутствие системной документации, мешающее формированию понимания системы и, как следствие, невозможности адекватного анализа влияния. Эта и другие проблемы могут быть решены при использовании систематического подхода к построению зрелых процессов, применению соответствующих техник и автоматизации необходимых задач по поддержке жизненного цикла с помощью специализированных инструментальных средств.

2.2 Управленческие вопросы (Management Issues)

mailto:sorlik@borland.ru
http://sorlik.blogspot.com

2.2.1 Согласование с организационными целями (Alignment with organizational objectivies)

Организационные цели описывают как продемонстрировать возврат инвестиций от деятельности по сопровождению программного обеспечения. Обычно, разработка ведется на проектной основе, с определенными временными и бюджетными ограничениями. Главный акцент, при этом, делается на выпуске системы, отвечающей потребностям пользователей, в заданные сроки и в рамках бюджета. В отличие от этого, сопровождение системы преследует цели максимального продления срока эксплуатации программного обеспечения. Такой подход может основываться на необходимости обновления и расширения программного обеспечения, как отклика на изменяющиеся потребности пользователей. При этом, оценка возврата инвестиций становится более сложной и приводит к формированию точки зрения старшего менеджмента, что деятельность по сопровождению потребляет значительную часть ресурсов без явно выраженной и количественно определяемой отдачи для организации.

2.2.2 Проблемы кадрового обеспечения* (Staffing)

Данная тема касается вопросов привлечения и удержания квалифицированного персонала по сопровождению. Часто, работа по сопровождению не выглядит привлекательной, инженеры по поддержке воспринимаются как специалисты “второго класса” (в SWEBOK используется устойчивое выражение “second-class citizens”), что приводит к безусловному падению духа коллектива, отвечающего за поддержку систем.

По мнению автора, это серьезный вызов для менеджеров, отвечающих за вопросы сопровождения и, вообще говоря, является классической задачей общего менеджмента. Решение этой задачи, в первую очередь, находится в руках старшего менеджмента, формирующего соответствующий стиль отношений между функциональными и вспомогательными подразделениями. На более высоком уровне, для организаций и бизнесов - потребителей информационных технологий, эта задача связана с внутрикорпоративными департаментами автоматизации, в целом, которые слишком часто воспринимаются только как центры затрат, а информационные технологии не рассматриваются как актив. В результате, такая позиция приводит к снижению эффективности работы подразделений автоматизации, а, следовательно, и падению качества информационного обеспечения бизнеса, что сказывается, в подавляющем большинстве случаев, и на бизнесе, как таковом.

* такой перевод, вместо просто “кадрового обеспечения”, в большей степени соответствует принятому использованию термина staffing. Часто, staffing подразумевает и высокую текучесть кадров.

2.2.3 Процесс (Process)

Процесс (в общем случае, жизненный цикл, *прим. автора*) является набором работ (activities), методов, практик и, своего рода, трансформаций, которые используются людьми для разработки и сопровождения программных систем и ассоциированных с ними продуктов. На уровне процесса, деятельность по сопровождению программного обеспечения имеет очень много общего с разработкой, например, в части конфигурационного управления, являющегося критически важной составляющей обоих видов деятельности. В то же время, сопровождение включает работы, не представленные в процессе разработки (в теме 3.2 представлено описание такого рода уникальных работ). Эта деятельность требует от менеджмента специального внимания.

2.2.4 Организационные аспекты сопровождения (Organizational aspects of maintenance)

В первую очередь, организационные вопросы подразумевают какая организация будет отвечать и/или какие функции необходимо выполнять для обеспечения деятельности по сопровождению. Команда, разрабатывавшая программный продукт, далеко не всегда отвечает за его сопровождение. Это не только стандартное управленческое решение независимых поставщиков программного обеспечения, но, также, часто встречается в организациях, использующих программные продукты в целях автоматизации своих бизнес-функций.

При решении вопроса, где (кем) будут осуществляться функции по сопровождению, может быть принято решение оставить их непосредственно тем, кто разрабатывал систему (как в терминах организации/компании, так и подразумевая непосредственно коллектив разработчиков), или передать другой команде или стороне (maintaner). Часто, выбор сопровождающей организации

mailto:sorlik@borland.ru
http://sorlik.blogspot.com

осуществляется исходя из тех соображений, которые выглядят обоснованными для обеспечения адекватной поддержки системы и возможности ее эволюционирования для удовлетворения меняющихся потребностей пользователей. К сожалению (чего, в принципе, и следовало ожидать), универсальных подходов в решении данного вопроса, кем будет сопровождаться система – нет. Соответствующие решения принимаются в каждом конкретном случае, с учетом его специфики (case-by-case). Но, что действительно важно отметить, делегирование или назначение полномочий и ответственности по сопровождению должно быть произведено по отношению только к одной организации или лицу (менеджеру соответствующей команды поддержки). Все, так или иначе, зависит от организационной структуры организации/компании, эксплуатирующей программное обеспечение.

2.2.5 Аутсоурсинг (Outsourcing)

Заимствованный термин “аутсоурсинг” уже прижился не только в среде ИТ-менеджеров, он стал частью современного бизнеса и управленческих практик. Суть его заключается в передаче работ, в первую очередь, вспомогательных (непрофильных для организации) “на сторону”. Крупные корпорации передают в управление другим организациям целые портфели программных систем, а, иногда, и целиком всю ИТ-инфраструктуру. В то же время, существенно более часто, сопровождение передается другим организациям только для “второстепенных” программных систем (или, как минимум, не критичных для выполнения бизнес-функций), так как владельцы таких систем не желают терять контроль над ассоциированными с этими системами данными и/или функциональностью. Отмечается, что некоторые передают работы по сопровождению “в аутсоурсинг” только в тех случаях, если убеждены в стратегическом контроле над сопровождением.

Автор не раз наблюдал, когда для решения вопросов сопровождения (при сохранении “стратегического контроля”), компании, для которых информационные технологии не являются профильными, но воспринимаются в качестве актива, формируют специализированные дочерние бизнес-структуры, которым и передаются функции сопровождения, а также и непосредственно разработки программных систем и, более того, поддержки и развития всей ИТ-инфраструктуры. Это делается с тем, чтобы функционируя в качестве самостоятельной бизнес-сущности, уже бывшие внутрикорпоративные подразделения автоматизации, могли обеспечить большую прозрачность финансовых потоков, затрат, связанных с информационным технологиями. Но, это тема уже относится к общим вопросам управления и, безусловно, требует самостоятельного обсуждения, в контексте, опять-таки, конкретной организации или бизнес-структуры. Однако, нельзя было не обозначить важность обсуждаемого вопроса в данном контексте, ведь именно деятельность по сопровождению часто подвигает организации-потребители ИТ к принятию столь серьезных организационных и бизнес-решений.

При этом, подчеркивает SWEBOK, контроль сложно измерить. В свою очередь, перед аутсоурсером (организацией, принимающей на себя ответственность по сопровождению) стоит серьезная проблема по определению содержания соответствующих работ, в том числе, для описания содержания соответствующего контракта. Отмечается, что около 50% сервисов, предоставляемых аутсоурсером, проводятся без соответствующего детального и однозначно интерпретируемого соглашения (service level agreement, SLA). Компании, занимающиеся аутсоурсингом, обычно затрачивают несколько месяцев на оценку программного обеспечения прежде, чем заключают соответствующий контракт. Еще один вопрос, требующий специального внимания, заключается в необходимости определения процесса и процедур передачи программного обеспечения на внешнее сопровождение.

2.3 Оценка стоимости сопровождения (Maintenance Cost Estimation)

Как уже отмечалось, инженеры должны понимать разницу в различных категориях сопровождения. Это, в большой степени, необходимо для оценки соответствующих затрат. С точки зрения планирования, как составной части проектной и управленческой деятельности, оценка стоимости является важным аспектом деятельности по сопровождению программного обеспечения.

2.3.1 Оценка стоимости (Cost Estimation)

При обсуждении анализа влияния (см. 2.1.3 Impact Analysis) говорилось о том, что такой анализ помогает в оценке стоимости работ по сопровождению. На эти затраты оказывает влияние

mailto:sorlik@borland.ru
http://sorlik.blogspot.com

множество технических и других факторов. ISO/IEC 14764 (Standard for Software Engineering - Software Maintenance)определяет, что “существует два наиболее популярных метода оценки стоимости сопровождения – параметрическая модель и использование опыта”. Чаще всего, оба этих подхода комбинируются для повышения точности оценки.

2.3.2 Параметрические модели (Parametric models)

SWEBOK приводит ряд источников, подробно рассматривающих вопросы оценки стоимости сопровождения и, в частности, параметрические модели. Для использования таких моделей используются данные предыдущих проектов. Наравне с историческими данными используется метод функциональных точек (см. стандарт IEEE 14143.1-00).

2.3.3 Опыт (Experience)

Среди тех подходов, которые позволяют повысить точность оценок, полученных при использовании параметрических моделей – применение опыта (в форме экспертного мнения, например, при использовании техники оценки “Delphi”, название которой происходит от “делфийского оракула”), аналогий, а также структуры декомпозиции работ. Наилучшие результаты получаются в случае сочетания эмпирических методов с имеющимся опытом. Получаемые данные используются как результат программы измерения аспектов сопровождения.

2.4 Измерения в сопровождении программного обеспечения (Software Maintenance Measurement)

Формы и данные измерений в процессе сопровождения могут объединяться в единую программу корпоративную программу количественных оценок, проводимых в отношении программного обеспечения. Многие организации используют популярный и практичный подход для измерений, базирующийся на оценке количества проблем и статуса их решений (issue-driven measurement). Идеи этого подхода систематизированы в проекте Practical Software and Systems Measurement (PSM). Существуют общие (для всего жизненного цикла) метрики и, соответственно, их категории, в частности, определяемые Институтом Программной Инженерии университета Карнеги-Меллон (Software Engineering Institute, Carnegie-Mellon University – SEI CMU): размер, усилия, расписание и качество. Применение этих метрик является хорошей отправной точкой для оценки работ со стороны организации, отвечающей за сопровождение.

Более детальное обсуждение вопросов измерений в отношении продуктов и процессов представлено в области знаний “Процесс программной инженерии (Software Engineering Process). В свою очередь, вопросы организации программы измерений относятся к области знаний “Управление программной инженерией” (Software Engineering Management).

2.4.1 Специализированные метрики (Specific Measures)

Существуют различные методы внутренней оценки продуктивности (benchmarking) персонала сопровождения для сравнения работы различных групп сопровождения. Организация, ведущая сопровождение, должна определить метрики, по которым будут оцениваться соответствующие работы. Стандарты IEEE 1219 (Standard for Software Maintenance) и ISO/IEC 9126-01 (Software Engineering – Product Quality – Part 1: Quality Model, 2001 г.) предлагают специализированные метрики, ориентированные именно на вопросы сопровождения и соответствующие программы.

Ниже представлены типичные метрики оценки работ по сопровождению, соответствующих распространенной классификации эксплуатационных характеристик программного обеспечения:

- Анализируемость (Analyzability): оценка (в первую очередь, дополнительных) усилий или ресурсов, необходимых для диагностики недостатков или причин сбоев, а также для идентификации тех фрагментов программной системы, которые должны быть модифицированы.
- Изменяемость (Changeability): оценка усилий, необходимых для проведения заданных модификаций.
- Стабильность (Stability): оценка случаев непредусмотренного поведения системы, включая ситуации, обнаруженные в процессе тестирования.
- Тестируемость (Testability): оценка усилий персонала сопровождения и пользователей по тестированию модифицированного программного обеспечения.

mailto:sorlik@borland.ru
http://sorlik.blogspot.com

3. Процесс сопровождения (Maintenance Process)

Вопросы организации процесса сопровождения напрямую связаны с соответствующими стандартами и de facto практиками реализации такого процесса. Тема “Работы по сопровождению” (Maintenance Activities) различает вопросы сопровождения и разработки и показывает взаимосвязь с другими аспектами деятельности программной инженерии.

Типичные и распространенные потребности в процессах программной инженерии подробно описаны и документированы в различных источниках. Одна из наиболее детально проработанных и распространенных (на уровне стандарта de facto) процессных моделей, изначально созданных с ориентацией на программное обеспечение – CMMI (Capability Maturity Model Integration – интегрированная модель зрелости), разработанная в Институте программной инженерии университета Карнеги-Меллон (SEI CMU). CMMI, в частности, уделяет специальное внимание процессам сопровождения. Существуют и другие, менее распространенные, но тем не менее развивающиеся модели.

3.1 Процессы сопровождения (Maintenance Processes)

Процессы сопровождения описывают необходимые работы и детальные входы/выходы этих работ. Эти процессы рассматриваются в стандартах IEEE 1219 (Standard for Software Maintenance) и ISO/IEC 14764 (Standard for Software Engineering - Software Maintenance).

Процесс сопровождения начинается по стандарту IEEE 1219 с момента передачи программной системы в эксплуатацию (post-delivery stage) и касается таких вопросов, как планирование деятельности по сопровождению (см. рисунок 2).

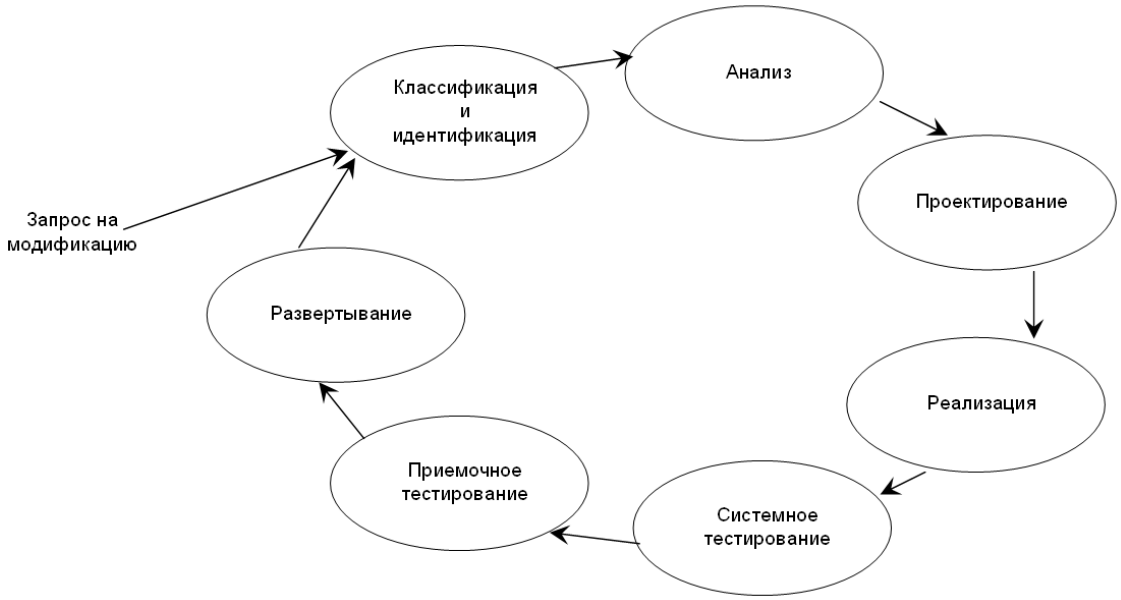

Рисунок 2. Работы в процессе сопровождения по стандарту IEEE 1219.

Стандарт ISO/IEC 14764 уточняет положения, связанные с процессом сопровождения, стандарта жизненного цикла 12207. Работы по сопровождению, описанные в этом стандарте аналогичны работам в IEEE 1219, за исключением того, что сгруппированы несколько иначе (см. рисунок 3).

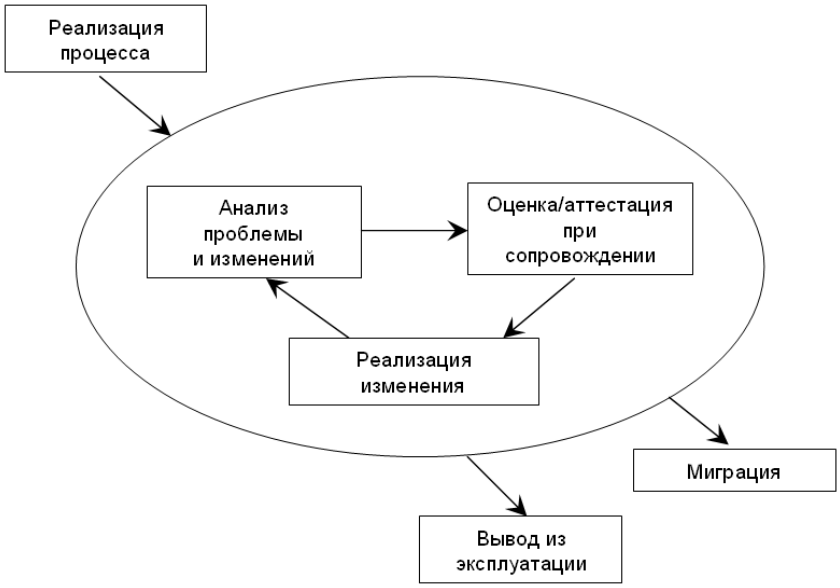

Рисунок 3. Процесс сопровождения по стандарту ISO/IEC 14764.

Работы по сопровождению в стандарте 14764 разбиты на задачи:

- Process Implementation – реализация процесса
- Problem and Modification Analysis – анализ проблем и <необходимых> модификаций
- Modification Implementation –проведение модификаций (реализация изменений)
- Maintenance Review/Acceptance – оценка и принятие <проведенных работ> при сопровождении
- Migration – миграция (на модифицированную или новую версию программного обеспечения)
- Software Retirement – вывод из эксплуатации (прекращение эксплуатации программного обеспечения)

В представленных в SWEBOK источниках можно найти описание истории эволюции соответствующих процессных моделей упоминаемых стандартов ISO/IEC и IEEE. Кроме того, существует и общая (обобщенная) модель процессов сопровождения. Agile-методологии, активно развивающиеся в последние годы, предлагают “облегченные” (light или lightweight) процессы, в том числе, и для организации деятельности по сопровождению, например, Extreme maintenance (см. соответствующие источники, указанные в SWEBOK).

3.2 Работы по сопровождению (Maintenance Activities)

Как уже отмечалось, многие работы по сопровождению похожи на аспекты деятельности по разработке. В обоих случаях необходимо проводить анализ, проектирование, кодирование, тестирование и документирование. Специалисты по сопровождению должны отслеживать требования так же, как и инженеры-разработчики и обновлять документацию по мере разработки и/или выпуска обновленных или новых релизов продукта. Стандарт ISO/IEC 14764 рекомендует, чтобы персонал или организации, отвечающие за сопровождение, в случае использования элементов процессов разработки в своей деятельности, адаптировали эти процессы <целиком> в соответствии со своими потребностями. В то же время, деятельность по сопровождению обладает и определенными уникальными чертами, что приводит к необходимости использования специализированных процессов.

3.2.1 Уникальные работы (Unique activities)

Существует ряд процессов, работ и практик, уникальных для деятельности по сопровождению. SWEBOK приводит следующие примеры такого рода уникальных характеристик:

- Передача (Transition): контролируемая и координируемая деятельность по передаче программного обеспечения от разработчиков группе, службе или организации, отвечающей

mailto:sorlik@borland.ru
http://sorlik.blogspot.com

за дальнейшую поддержку;

- Принятие/отклонение запросов на модификацию (Modification Request Acceptance/Rejection): запросы на изменения могут как приниматься и передаваться в работу, так и отклоняться по различным обоснованным причинам – объему и/или сложности требуемых изменений, а также необходимых для этого усилий. По мнению автора, соответствующие решения могут также приниматься на основе приоритетности, оценке обоснованности, отсутствии ресурсов (в том числе, отсутствия возможности привлечения разработчиков к решению задач по модификации, при реальном наличии такой потребности), утвержденной запланированности к реализации в следующем релизе и т.п..

- Средства извещения персонала сопровождения и отслеживания статуса запросов на модификацию и отчетов об ошибках (Modification Request and Problem Report Help Desk): функция поддержки конечных пользователей, инициирующая работы по оценке (assessment, подразумевая в том числе оценку необходимости), анализу приоритетности и стоимости модификаций, связанных с поступившим запросом или сообщенной проблемой.

- Анализ влияния (Impact Analysis): анализ возможных последствий изменений, вносимых в существующую систему - рассматривался выше в теме 2.1.3.

- Поддержка программного обеспечения (Software Support): работы по консультированию пользователей, проводимые в ответ на их информационные запросы (request for information), например, касающиеся соответствующих бизнес-правил (см. область знаний “Требования к программному обеспечению”, *прим. автора*), проверки, содержания данных и специфических (ad hoc) вопросов пользователей и их сообщений о проблемах (ошибках, сбоях, непредусмотренному поведению, непониманию аспектов работы с системой и т.п.).

- Контракты и обязательства: к ним относятся классическое соглашение об уровне предоставляемого сервиса - Service Level Agreement (SLA), а также другие договорные аспекты, на основании которых, группа/служба/организация по сопровождению выполняет соответствующие работы.

С точки зрения автора, на практике сложно провести грань между разделяемыми в SWEBOK функциями Help Desk и Software Support –эти функции, обычно, совмещены с процессной точки зрения.

3.2.2 Дополнительные работы, поддерживающие процесс сопровождения (Support activities)

Столь длинный перевод названия данной темы связан с содержанием соответствующих работ, описываемых SWEBOK, как работы персонала сопровождения, не включающие явного взаимодействия с пользователями, но необходимые для осуществления соответствующей деятельности. К таким работам относятся: планирование сопровождения. Конфигурационное управление (software configuration management), проверка и аттестация (V&V – verification and validation), оценка качества программного обеспечения (software quality assessment), различные аспекты обзора, анализа и оценки (reviews), аудит (audit) и обучение (training) пользователей.

Также, к таким специальным (внутренним) работам относится обучение персонала сопровождения.

В силу особой значимости (с точки зрения автора - обязательности) ряда упомянутых работ, им посвящены следующие под-темы данной секции области знаний по сопровождению программного обеспечения.

3.2.3 Работы по планированию сопровождения (Maintenance planning activity)

Планирование является более чем необходимым для адекватного проведения работ по сопровождению и должно касаться связанных с этим вопросов с нескольких точек зрения:

- Бизнес-планирование (организационный уровень)
- Планирование непосредственных работ по сопровождению (уровень передачи программного обеспечения – см. выше 3.2.1)
- Планирование релизов/версий (уровень программного обеспечения)

mailto:sorlik@borland.ru
http://sorlik.blogspot.com

- Планирование обработки конкретных запросов на изменение (уровень запроса)

На уровне индивидуального запроса, работы по планированию проводятся вместе с проведением анализа влияния (см. 2.1.3). В свою очередь, планирование релизов/версий требует от персонала сопровождения выполнения задач:

- Получения и сбора информации о датах размещения индивидуальных запросов и отчетов
- Достижения соглашения с пользователями о содержании (функциональности, поведении и т.п.) последующих релизов/версий программного обеспечения
- Идентификации потенциальных конфликтов и возможных альтернатив <реализации необходимых запросов>
- Оценки рисков для <функционирования> текущего релиза и разработки плана “отката” на немодифицированный (текущий, до внесения изменений, касающихся рассматриваемого запроса) вариант системы, в случае возникновения проблем, связанных с модификацией
- Информирования всех заинтересованных лиц

Несмотря на то, что разработка программных системы, обычно, занимает от несколько месяцев (что более типично) до нескольких лет, сопровождение (как деятельность по поддержке использования) и активная эксплуатация систем занимает несколько лет, а то и более* (5-10-...). Проведение оценки ресурсов – неотъемлемая часть планирования. Ресурсы, необходимые для сопровождения должны быть оценены и заложены в бюджет еще при *разработке* системы. Планирование работ по сопровождению должно начинаться одновременно с принятием решения о создании системы и согласоваться с целями обеспечения качества (отмечается в IEEE 1061-98 Standard for a Software Quality Metrics Methodology).

Вначале необходимо определить *концепцию сопровождения*. Такой документ, например, по стандарту ISO/IEC 14764 (Standard for Software Engineering - Software Maintenance) должен касаться следующих вопросов:

- Содержания деятельности по сопровождению
- Адаптации процесса сопровождения
- Идентификации организации, которая будет заниматься сопровождением
- Оценки стоимости сопровождения

После разработки концепции деятельности по сопровождению должен быть сформирован соответствующий *план сопровождения*. Этот план должен подготавливаться одновременно с разработкой программной системы. План должен определять как пользователи будут размещать свои запросы на модификацию (изменения) или сообщать об ошибках, сбоях и проблемах. Вопросам планирования уделяют специальное внимание уже упоминавшиеся стандарты IEEE 1219 (Standard for Software Maintenance) и ISO/IEC 14764 (Standard for Software Engineering - Software Maintenance). Стандарт ISO/IEC 14764 предоставляет специальные рекомендации (guidelines) по организации плана сопровождения.

Наконец, на уровне бизнес-вопросов, структура, отвечающая за сопровождение, должна проводить общую деятельность по бизнес-планированию, касающееся бюджетирования, финансового менеджмента и управления человеческими ресурсами, так же, как и любое другое (в том числе, профильное, если речь идет о потребителях ИТ) подразделение организации/компании. Необходимые знание в области управления также затрагиваются в области знаний SWEBOK “Связанные дисциплины”.

* эта оценка базируется и на опыте автора, когда в начале 90-х он принимал непосредственное участие в проектировании и разработке системы автоматизации добровольного медицинского страхования (в одной из крупнейших российских страховых компаний), выведенной, в дальнейшем, из эксплуатации на рубеже 2000 года, то есть система функционировала около 7 лет, а некоторые ее фрагменты как самостоятельные приложения и того дольше. Многие банковские приложения, особенно, на западе, в первых своих версиях были разработаны еще в середине 80-х, а некоторые ИТ-системы в мире были созданы (в своем изначальном варианте) и в 70-е годы, что, в частности, привело к усиленному вниманию к проблеме 2000 года (Y2K problem).

3.2.4 Конфигурационное управление (Software configuration management)

Стандарт IEEE 1219, посвященный организации сопровождения программного обеспечения, определяет конфигурационное управление как критически важный элемент процесса сопровождения. Процедуры конфигурационного управления должны обеспечивать проверку,

mailto:sorlik@borland.ru
http://sorlik.blogspot.com

аттестацию и аудит на всех шагах, требуемых для идентификации, авторизации, реализации и выпуска программного продукта.

Более того, недостаточно просто отслеживать запросы на изменения и сообщения о проблемах (modification requests, problem reports). Должны быть контролируемы и сам программный продукт, и любые изменения (не только в коде, но документации, спецификациях и т.п., то есть любых активах продукта *и* проекта, *прим. автора*). Такой контроль устанавливается реализацией и строгим следованием утвержденным процессам конфигурационного управления (software configuration management, SCM). Область знаний “Конфигурационное управление” подробно описывает и обсуждает процессы, в соответствии с которыми, размещаются, оцениваются и утверждаются запросы на изменения. В ряде отдельных аспектов и характеристик, конфигурационное управление при сопровождении и разработке несколько отличается, что должно контролироваться уже на операционном уровне. Реализация SCM-процесса обеспечивается разработкой и следованием плану конфигурационного управления и соответствующим процедурам (operating procedures). Организация, подразделение или группа сопровождения (в лице представителей) участвует в работе часто формируемого органа Configuration Control Board, отвечающего за рассмотрение и принятие в работу запросов на изменения. Основной целью такого участия является, по мнению SWEBOK, определение содержания следующих релизов/версий.

3.2.5 Качество программного обеспечения (Software quality)

Недостаточно, всего лишь, надеяться, что в процессе и результате сопровождения, качество программного обеспечения будет повышаться. Для поддержки процесса сопровождения должны планироваться и реализовываться соответствующие процедуры и процессы, направленные на повышение качества. Работы и техники по обеспечению качества (Software Quality Assurance, SQA), проверке и аттестации (V&V, verification and validation), обзору, анализу и оценке (review), а также аудиту, должны отбираться в контексте взаимодействия и согласования со всеми другими процессами, направленными на достижение желаемого уровня качества. SWEBOK, основываясь на стандарте ISO/IEC 14764 (Standard for Software Engineering - Software Maintenance), рекомендует адаптировать соответствующие процессы, техники и активы, относящиеся к разработке программного обеспечения. К ним, например, относятся документация по тестированию и результаты тестов. Дополнительные подробности можно найти в соответствующей области знаний “Качество программного обеспечения” (Software Quality).

4. Техники сопровождения (Techniques for Maintenance)

Данная секция вводит некоторые общепринятые техники, используемые в процессе сопровождения программных систем.

4.1 Понимание программных систем (Program Comprehension)

Для реализации изменений программисты тратят значительную часть времени на чтение и формирование понимания программного продукта. Средства работы с кодом являются ключевым инструментом для решения этой задачи. Четкая, однозначная и лаконичная документация обеспечивает адекватное понимание программных систем.

4.2 Реинжиниринг (Reengineering)*

Реинжиниринг определяется как детальная оценка (examination) и перестройка программного обеспечения для формирования понимания, воссоздания (на уровне модели и, в ряде случаев, требований, *прим. автора*) и дальнейшей реализации его <функций> в новой форме (например, с использованием новых технологий и платформ, при сохранении существующей и расширением и облегчением возможностей добавлений новой функциональности, *прим. автора*). Отмечается, что в индустрии существуют различные позиции в отношении реинжиниринга – одни считают, что реинжиниринг является наиболее радикальной и затратной формой изменений программных систем, другие, что такой подход может применяться и для не столь кардинальных изменений (например, как смена платформы или архитектуры, *прим. автора*). Реинжиниринг, обычно, провидится не столько для улучшения возможностей сопровождения (maintainability), сколько для замены устаревшего программного обеспечения. В принципе, реинжиниринг можно рассматривать как самостоятельный проект (такой позиции придерживается автор), включающий в себя, как отмечает SWEBOK,

mailto:sorlik@borland.ru
http://sorlik.blogspot.com

формирование концепции, применение соответствующих инструментов и техник, анализ и приложения опыта проведения реинжиниринга, а также оценку рисков и преимуществ, связанных с такими работами.

Хочу отметить, что реализация продукта в новом качестве (форме) при сохранении основной функциональности оригинального продукта, является неотъемлемой и определяющей частью реинжиниринга, в отличие от обратного инжиниринга, рассматриваемого ниже и являющегося важной составной частью реинжиниринга.

4.3 Обратный инжиниринг (Reverse engineering)*

“Обратный” инжиниринг (часто путаемый с реинжинирингом, в том числе, в понимании SWEBOK, *прим. автора*) или это процесс анализа программного обеспечения с целью идентификации программных компонент и связей между ними, а также формирования представления о программном обеспечении, с дальнейшей перестройкой в новой форме (уже, в процессе реинжиниринга). Обратный инжиниринг является *пассивным*, предполагая отсутствие деятельности по изменению или созданию нового программного обеспечения. Обычно, в результате усилий по обратному инжинирингу создаются модели вызовов (call graphs) и потоков управления (control flow graphs) на основе исходного кода системы. Один из типов обратного инжиниринга – создание новой документации на существующую систему (redocumentation). Другой из распространенных типов – восстановление дизайна системы (design recovery).

К вопросам обратного инжиниринга, как и к вопросам реинжиниринга, также относятся работы по рефакторингу (см. работы Мартина Фаулера, впервые систематизировавшего и описавшего рефакторинг, *прим. автора*). *Рефакторинг* – трансформация программного обеспечения, в процессе которой программная система реорганизуется (не переписываясь) с целью улучшения структуры, без изменения поведения. Сохранение “формы” (платформы, архитектурных и технологических решений) существующей программной системы позволяет рассматривать рефакторинг как один из вариантов обратного инжиниринга.

* для обеих тем – 4.2 и 4.3 возможно применение слова *реконструкция*, в зависимости от контекста, означающее как полную перестройку или воссоздание чего-либо, идентичного по определенным характеристикам оригинальному образцу, так и восстановление какой-либо сущности по сохранившимся и/или доступным внешним признакам, где восстановление может подразумевать опять-таки создание нового образца или представления об оригинальной сущности.

По мнению автора, возможно объединение тем данной секции, включая реинжиниринг и обратный инжиниринг, в общую тему 4.1 “Reverse and Re-engineering” данной области знаний “Сопровождение”, с дальнейшей детализаций в виде “под-тем” 4.1.x. Такой подход соответствует структуре SWEBOK. При этом, соответствующая структура может быть организована, например, следующим образом:

- *4.1.1 Формирование представления об эксплуатируемой/сопровождаемой системе*: включает восстановление, в первую очередь, бизнес- и функциональных требований к системе:
- *4.1.2 Восстановление детального дизайна системы*: включает идентификацию всех компонентов системы и создание модели вызовов и других связей между компонентами;
- *4.1.3 Рефакторинг*: как возможный процесс структурных изменений, вносимых в систему, в частности для улучшения возможностей по ее дальнейшему сопровождению (включая модификацию, связанную с расширением функциональности);
- *4.1.4 Переработка системы*: создание нового релиза/версии системы в той же форме (например, с использованием той же технологической платформы), что и текущая (эксплуатируемая) версия;
- *4.1.5 Создание новой системы*: рассматривает текущую версию и систему в целом, как устаревшую – legacy.

mailto:sorlik@borland.ru
http://sorlik.blogspot.com