Введение

Хаос тестирования

Проект находится под угрозой срыва, а сроки его окончания быстро приближаются. Руководство решает протестировать продукт лишь тогда, когда уже утрачены почти все возможности для повышения качества программного обеспечения. Если какого-то неудачливого программиста назначают на тестирование программного обеспечения, и к тому же несчастному не дается никаких указаний и никто ничем в организации не может помочь, то обычно это рассматривается как "перевод в чистилище". Несмотря на плохое состояние требований, и другой документации, продукт разрабатывается и будет выпущен. Задача, поставленная перед тестером, заключается в том, чтобы свести к минимуму количество неприятных сюрпризов, которые могут возникнуть после установки продукта у заказчика. Находясь под сильным давлением, неподготовленный тестер оказывается в сложной ситуации и не представляет себе, что делать. Несведущий менеджер может купить какой-либо инструмент автоматического тестирования несмотря на отсутствие пригодных для автоматизации тестов. Именно такое положение вещей способствует распространению дурной славы о тестировании.

Тестирование программного обеспечения — это отдельная дисциплина, требующая специальных навыков. Тестирование программного обеспечения не интуитивно; кто-то должен знать, *как* это делается. Наивные менеджеры ошибочно полагают, что тестирование программного обеспечения доступно любому программисту, — если он может составлять программы, значит, может и тестировать продукт. Мотивом для написания этой книги стало желание рассказать о поэтапном методе, дающем возможность начать собственно тестирование.

На сегодняшний день существует большое количество хороших книг по тестированию программного обеспечения. В работах, которые посвящены проектированию тестовых примеров, описываются такие проверенные методы, как анализ граничных значений, разбиение на классы эквивалентности, таблицы решений, тестирование синтаксиса, причинно-следственные диаграммы, потоковые методы и т.д. Некоторые начинающие тестеры еще до применения перечисленных выше методов интересуются тем, как можно справиться с плохой спецификацией. Во многих работах утверждается, что для проведения тестирования необходимы хорошие требования (предполагается, что они уже существуют), хотя автору этой книги не встречались такие работы, в которых объяснялся бы переход от требований к тестовым примерам. В хаосе разработки программного обеспечения адекватные требования редко бывают представлены, а если они и есть, под вопросом остается их полнота и корректность. Анализ требований в большинстве случаев ложится на тестера, который должен перед

заданием каких-либо тестов рассмотреть проблемы, связанные с требованиями. В сжатые сроки с ограниченными ресурсами нельзя провести всестороннее тестирование, однако можно сделать разумный выбор и выполнить работу по тестированию с максимально возможной эффективностью.

Основная цель тестера — изучить оптимальные подходы к задачам тестирования и, в конечном счете, наладить рабочий процесс тестирования для будущих проектов.

Философия автора

Чтобы изложить идеи, касающиеся проведения тестирования, в этой книге рассматривается несколько примеров с так называемыми "требованиями". По определению, хорошие требования должны быть недвусмысленными и тестируемыми. Тот факт, что представленные образцы требований неполны, не означает, что тестирование проводить нельзя. Слово "требования" здесь используется нестрого и приравнивается к некоему виду описания продукта. Если предложенные образцы сценариев тестирования неприемлемы для уже сформировавшихся организаций, занимающихся производством ПО, они помогут отдельным тестерам в случае необходимости быстро начать процесс тестирования. Основная задача здесь заключается в том, чтобы показать, что для начала тестирования можно использовать любую информацию о продукте, даже неполную.

Изложенные выше рассуждения не следует рассматривать как аргумент в пользу работы с плохими требованиями. Правильный анализ требований позволяет исправить многие упущения. Доказано, что пересмотры и инспекции с точки зрения затрат — наиболее эффективные методы для обнаружения проблем на ранних стадиях цикла разработки. Автору часто приходилось сдерживаться, чтобы не высказать менеджерам проекта свое негодование: "Требования совсем никуда не годятся и мы не можем продуктивно проводить тестирование до тех пор, пока вы не проясните ситуацию". В действительности эта фраза обычно содержит непечатные идиоматические выражения и произносится под чьи-то вздохи, однако несмотря ни на что, сроки выпуска продукта стремительно приближаются.

Хотя здесь не будет отстаиваться идея "срезания углов", все же существуют способы сокращения рутинной деятельности, которые могут помочь документировать процесс тестирования. Лучше иметь непродуманный список тестов, чем совсем ничего. Самое меньшее, что нужно сделать — это документировать след, пусть даже элементарный, чтобы фиксировать работу, выполненную при тестировании, и иметь возможность продемонстрировать то, что уже было сделано.

Последующая работа над тестированием даст возможность улучшить эти начальные наработки путем создания тестовой документации и разработки процедуры тестирования, которая будет в большей степени способствовать успешному завершению процесса.

Само по себе знание того, с чего начать тестирование, является серьезным достижением. Основная задача этой книги заключается в том, чтобы помочь тестеру понять, как информация о продукте преобразуется в тестовые примеры. Существует множество книг, в которых объясняются концепции и методы тестирования программного обеспечения. Чтобы не повторять того, что уже было написано другими

авторами, здесь будут сделаны ссылки на их работы. В первую очередь, эта книга должна помочь начать тестирование. Данная книга дополняет существующую на сегодня литературу по тестированию программного обеспечения и представляет собой введение к методикам тестирования ПО.

Предполагаемая аудитория

Эта книга может быть интересной:

- для специалистов, которые впервые занялись тестированием программного обеспечения и которым нечем руководствоваться или же негде подучиться;
- для менеджеров или руководителей, которые сами могут быть опытными специалистами по тестированию или которые ищут возможности получить руководство для начинающих тестеров;
- опытных программистов, занимающихся тестированием;
- опытных тестеров, ищущих новые идеи.

Если читатель не очень знаком с понятиями тестирования программного обеспечения, он должен уметь обращаться с компьютером, а также уметь пользоваться текстовым процессором и электронными таблицами.

Перечень должностных лиц

В книге используются следующие общие термины для описания должностных лиц.

Тестер	Специалист, который описывает и проводит тесты.
Разработчик	Специалист, который создает приложение (включая проект и исходный код), а также проводит окончательную интеграцию продукта.
Руководитель проекта	Авторитетный специалист, занимающийся календарным планом и кадровым обеспечением.
Авторитетное лицо проекта	Специалист по предметной области, обладающий достаточным опытом и авторитетом для составления и согласования требований.

Эти должности охарактеризованы без какой-либо привязки к структуре организации или к цепочке подотчетности служащих. Описание кадрового обеспечения и ответственности за работу меняется от организации к организации.

В зависимости от того, как комплектуется штат проекта, тестер может находиться в одной группе с разработчиками или же принадлежать к отдельной группе тестирования. В иных проектах может потребоваться, чтобы один и тот же человек выполнял одновременно задачи разработки и тестирования, меняя, таким образом,

свое положение в проекте. В идеале работу, связанную с тестированием, выполняет обученный специалист по тестированию программного обеспечения, однако в некоторых проектах на роль тестера назначают любого свободного специалиста.

Авторитетным лицом проекта может быть менеджер по маркетингу, руководитель компании или же посредник, занимающийся поддержкой заказчиков. Эта роль необходима для определения содержания проекта и предотвращения дальнейшего хаоса. Авторитетное лицо проекта отвечает за принятие решений относительно того, какие функции будут включены в продукт; недостаток такого жесткого контроля выливается в ряд проблем, возникающих при согласовании требований.

В организациях могут быть задействованы и другие должностные лица. Ключевым моментом здесь является назначение сотрудников (каждый из которых должен иметь специальные навыки) для выполнения необходимых задач.

Среда разработки программного обеспечения: формирование и зрелость

Примеры, упомянутые в этой книге, содержат неполные сведения о продукте — именно то, чего можно ожидать от несформировавшейся организации разработки ПО. Автор постарается воздержаться от критики рабочей среды и от пропаганды улучшения процесса разработки ПО. Реалии таковы, что многие компании функционируют в далеко не идеальных условиях. За исключением случая, когда полностью отсутствуют подходящие процессы разработки, продукт можно разрабатывать и сдавать заказчику. Однако все же нужно провести хотя бы минимальное тестирование. При плохих требованиях тестер тратит больше времени на обнаружение недостатков в определении требований вместо того, чтобы выполнять задачи, связанные с тестированием.

В компаниях, занимающихся эффективной разработкой ПО, наблюдаются определенные нормы. В несформировавшихся организациях часто не понимают, как влияют на продукт:

- удобные требования и описания продукта;
- проведение пересмотров и инспекций;
- наличие точек выхода или контрольных точек между этапами;
- обучение и подготовка персонала;
- адекватное планирование времени и ресурсов для тестирования;
- перекрытие процессов тестирования и разработки;
- введение четких процессов разработки и тестирования ПО;
- конфигурационный менеджмент.

Тестирование — это ответственность, которую разделяет и остальная часть команды разработчиков. Взгляд на тестирование по схеме *проектирование – написание кода – тестирование* никогда не даст хороших результатов. Деструктивный подход,

сопровождающийся соперничеством типа “разработчик против тестера”, часто является результатом игнорирования разработчиками процессов тестирования — еще одно подтверждение того, что тестирование — это уникальная дисциплина. Обычно тестер о программировании знает больше, чем разработчик о тестировании. Обоюдное сближение тестеров и разработчиков способствует доброжелательной атмосфере и хорошим отношениям в организации. Работая в постоянном контакте с тестерами, разработчики больше узна*ют* о тестировании программного обеспечения, а значит, и о самом продукте. Эффективное тестирование программного обеспечения требует объединения усилий всех членов проекта.

Краткий обзор

Глава 1 начинается с кошмара: специалист ничего не знает о тестировании, продукт выпускается в следующую пятницу, а документация проекта неполна, неточна, или, что еще хуже, ее вообще не существует. Предполагая, что в следующем проекте у такого специалиста будет больше времени на проведение тестирования, в главе 2 показано использование схем, которые также являются полезной методикой анализа требований. В главе 3 схема преобразуется в тестовые примеры. Обычные и электронные таблицы — неотъемлемый инструмент, который используется специалистами по тестированию программного обеспечения, поэтому в главе 4 показано несколько форматов таблиц и сокращенных форматов документирования тестовых примеров. В главе 5 показано дополнительное применение таблиц. В приложениях, созданных с помощью объектно-ориентированных методов, могут использоваться многие из тех методов проектирования тестовых примеров, которые были рассмотрены в предыдущих главах. В главе 6 описаны некоторые конкретные сложности, возникающие при тестировании объектно-ориентированных систем. В главе 7 анализируются проблемы, с которыми можно столкнуться при тестировании Web-приложений, хотя многие из представленных здесь стратегий применяются аналогично клиент-серверным технологиям.

Не существует единого метода тестирования программного обеспечения. В каждом примере для создания тестов используется свой подход. Читателя может удивить, почему в примере использовался именно этот метод, а не какой-нибудь другой. Причина проста: методы выбирались, исходя из опыта автора. Можно попробовать иные подходы, которые могут оказаться весьма удачными при проведении тестирования. В примерах охвачены различные типы приложений, но основная идея тестирования программного обеспечения одинаково применима ко всем типам, а некоторые концепции неоднократно повторяются во всех главах. Автор рекомендует прочесть все сценарии и не пропускать какие-либо темы только лишь потому, что в примере не отображен интересующий вас тип приложения.

Следуя представленным далее идеям и методам, можно определить и документировать множество тестовых примеров. Глава 8 поможет читателю определиться с наиболее подходящими тестами и снизить таким образом число проводимых тестовых примеров. Создание набора тестовых примеров, которые нужно будет выполнить, — это только часть общей картины. В главе 9 перечисляются другие задачи,

связанные с тестированием и качеством, которые необходимы для создания эффективного ПО.

Для организации, которая впервые рискнула пойти на методичное тестирование программного обеспечения, данная книга — незаменимое руководство. Значительного упорядочения предыдущих хаотических усилий еще недостаточно для соответствия продукта промышленным стандартам. В главе 10 вкратце описаны некоторые из наиболее общих стандартов разработки программного обеспечения и влияние каждого из них на образцы тестовых примеров. Это, так сказать, отправная точка для усовершенствования процесса тестирования.