высказывать собственное мнение о прочитанном – составляет неотъемлемую часть культуры человека в современном обществе.

УДК 321

Ярош Н.С.

ПРИНЦИПЫ ТЕСТИРОВАНИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

БНТУ, Минск
Научный руководитель: Дробыш А.А.

Программное обеспечение стало настолько неотъемлемой частью нашей жизни, что во многих случаях мы не осознаем, насколько сильно от него зависим. Не осознаем ровно до тех пор, пока что-то пойдет не так.

Первые программные системы разрабатывались в рамках программ научных исследований или программ для нужд министерств обороны. Тестирование таких продуктов проводилось строго формализованно с записью всех тестовых процедур, тестовых данных, полученных результатов. Тестирование выделялось в отдельный процесс, который начинался после завершения кодирования, но при этом, как правило, выполнялось тем же персоналом.

Существующие на сегодня методы тестирования программного обеспечения не позволяют однозначно и полностью выявить все дефекты и установить корректность функционирования анализируемой программы, поэтому все существующие методы тестирования действуют в рамках формального процесса проверки исследуемого или разрабатываемого программного обеспечения. Поэтому необходимо понять, какова сфера действия и ограничения тестирования и как правильно его выполнять.

Прежде чем дать определение, что такое «тестирование», попробуем понять, чем тестирование не является.

1. Тестирование не разработка. Тестировщики умеют программировать, в том числе и тесты (автоматизация тестирование = программирование), могут разрабатывать какие-то вспомогательные программы (для себя). Тем не менее, тестирование – это не деятельность по разработке программного обеспечения.

2. Тестирование не анализ и не деятельность по сбору и анализу требований.Хотя, в процессе тестирования иногда приходится уточнять требования, а иногда приходится их анализировать, но эта деятельность не основная, скорее, это приходится делать просто по необходимости.

3. Тестирование не управление. Несмотря на то, что во многих организациях есть такая роль, как «тест-менеджер». Конечно же, тестировщиками надо управлять. Но само по себе тестирование управлением не является.

4. Тестирование не техписательство, однако тестировщикам приходится документировать свои тесты и свою работу.

Тестирование нельзя считать ни одной из этих деятельностей просто потому, что в процессе разработки (или анализа требований, или написания документации для своих тестов) всю эту работу тестировщики делают для себя, а не для кого-то другого.

Тестирование – очень тонкий процесс, и у тестировщиков нет навязчивого желания сломать ваше приложение. Тестирование как метод верификации – это парадокс. Тестирование программы для того, чтобы проверить ее качество, теоретически равносильно втыканию булавок в куклу, причем очень маленьких булавок в очень большую куклу. Разрешить этот парадокс можно, определив реалистичные ожидания.

Часто тестированию в литературе придается огромное значение: «Тестирование программного обеспечения – это процесс оценки качества компьютерных программ. Тестирование – это практическое техническое изучение, проводимое для

того, чтобы предоставить заинтересованным сторонам информацию о качестве продукта или сервиса с учетом контекста, в котором, как предполагается, он будет работать». На самом деле, тестирование программы мало что говорит о ее качестве, поскольку 10 или даже 10 млн тестов – это лишь капля в океане всех возможных случаев.

Безусловно, связь между тестами и качеством программ существует, но она довольно слаба. Успешный тест позволяет оценить качество только в том случае, если прежде он не был пройден. Тогда это свидетельствует об отсутствии неудачи и, как правило, – отсутствии самой ошибки.

Если систематически отслеживать неудачи и ошибки, то их регистрация может дать представление о том, сколько еще ошибок осталось.

Единственная бесспорная связь – это отрицательная связь, или «фальсификация». Неудачный тест говорит о недостаточном качестве программы. Кроме того, если раньше тест проходил, а теперь нет, то это свидетельство регрессии, указывающее на возможные проблемы качества в программе и в процессе разработки. Самое известное высказывание о тестировании следующее: «Тестирование программ можно использовать для того, чтобы показать наличие ошибок и никогда для того, чтобы показать их отсутствие!» Очень немногие понимают, что для тестировщиков это означает наилучшую возможность для саморекламы. Безусловно, любая методика, позволяющая находить ошибки, крайне важна для «заинтересованных лиц», от менеджеров до разработчиков и пользователей.

Изложенные в статье принципы сформулированы на основе практического опыта тестирования программного обеспечения и исследований, предварявших разработку автоматизированных инструментальных средств, таких как AutoTest.

Принцип 1. Определение. Для того чтобы протестировать программу, нужно попытаться заставить ее работать неверно.

В силу этого принципа процесс тестирования обретает цель: его единственная задача – найти ошибки, инициируя неудачное выполнение. Любое умозаключение по поводу качества относится к области гарантии качества, но никак не к области тестирования. Это определение также напоминает нам, что тестирование связано только лишь с поиском ошибок.

Принцип 2. Тесты или спецификации. Тесты не заменяют спецификации (законченное описание поведения программы, которую требуется разработать).

Опасность заблуждения, что тесты могут выступать в роли спецификаций, была продемонстрирована целым рядом программных катастроф, которые произошли только потому, что никто не подумал об исключительном случае. Несмотря на то, что в спецификациях тоже могут быть упущены из виду какие-то случаи, по крайней мере, в них предпринимается попытка некого обобщения. Всегда нужно помнить об ошибке, которую уже когда-то обнаружили.

Принцип 3. Регрессивное тестирование. Любое неудачное выполнение должно порождать тестовый случай, который навсегда становится частью тестового пакета данного проекта.

Этот принцип касается всех ошибок, возникших во время разработки и тестирования. Отсюда вытекает необходимость в инструментарии, позволяющем превращать неудачное исполнение в воспроизводимый тестовый случай.

Принцип 4. Использование предсказаний. Определение успеха или неудачи тестов должно происходить автоматически.

Эта формулировка оставляет открытым вопрос о форме таких предсказаний. Зачастую предсказания специфицируются отдельно. В исследованиях они встроенные, поскольку

анализируемое программное обеспечение уже включает в себя контракты, согласно которым тесты используются как предсказания.

Принцип 4. Вариант. Предсказания должны быть частью текста программы как варианты. Успех или неудача теста должны определяться автоматически, причем в рамках этого процесса необходимо вести мониторинг выполнения варианта во время работы программы.

Принцип 5. Тестовые случаи. Эффективный процесс тестирования должен включать в себя тестовые случаи, проверяемые вручную и автоматически.

Достоинством тестов, выполняемых вручную, является их глубина: они отражают понимание разработчиком имеющегося круга проблем и структуры данных. Преимущество автоматических тестов в их широте: они выполняют проверку большого диапазона значений, в том числе экстремальных, которые люди могут пропустить.

Принцип 6. Эмпирические оценки стратегий тестирования. Оценивайте любую стратегию тестирования, однако, какой бы интересной она ни казалась, прибегайте к объективной оценке, используя точные критерии в воспроизводимом процессе тестирования.

Если поместить несколько пчел и мух в бутылку и перевернуть ее донышком к источнику света, пчелы, привлеченные светом, будут биться о стекло и умрут от голода и истощения. Мухи же, ничего не понимающие, испробуют все направления и вылетят из бутылки через пару минут. Это прекрасная метафора для тех случаев, когда, очевидная глупость превосходит очевидный ум, как это часто происходит при тестировании.

Принцип 7. Критерий оценки. Самое важное свойство стратегии тестирования – это число обнаруженных ошибок как функция времени.

Функция обнаружения, то есть число ошибок в зависимости от времени, полезна по двум причинам – используя программную базу с известными ошибками, можно оценить стратегию, посмотрев, сколько ошибок база позволит обнаружить за данное время.

Тестирование – это порождение неудач. Неудовлетворительное выполнение программы – это «неудача», указывающая на «ошибку» в программе, которая сама по себе является следствием «заблуждения» программиста.

УДК 321

Ярош Н.С.

СРЕДСТВА СОЗДАНИЯ ИНТЕРНЕТ-ПРИЛОЖЕНИЙ

БНТУ, Минск
Научный руководитель: Астапчик Н.И.

Веб-программирование (веб-разработка) – наиболее бурно развивающийся в настоящее время раздел программирования, ориентированный на создание динамических Интернет-приложений.

Языки, используемые для веб-программирования, можно разделить на две группы: клиентские и серверные.

Клиентские языки – приложения, написанные с использованием данной технологии, обрабатываются на стороне пользователя (в основном браузером).

Преимуществом использования клиентского языка состоит в том, что обработка скриптов производится до отправки на сервер. Отсюда вытекает ограничение, что при помощи клиентского языка невозможно записать данные на сервер.

HTML. HyperText Markup Language (язык разметки гипертекста) – стандартный язык разметки документов. Большинство веб-страниц создаются при помощи HTML. Хотя HTML интерпретируется браузером и отображается в виде документа,