Русов Артем Валерьевич
Тестировщик программного обеспечения
QA Академия руководитель
(г. Минск, Беларусь)

СТРАТЕГИИ И МЕТОДЫ ТЕСТИРОВАНИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Аннотация: *В статье рассматриваются современные стратегии тестирования программного обеспечения, включая поведенческое и структурное тестирование. Описаны основные особенности каждого метода, их преимущества и ограничения. Показано, что исчерпывающее тестирование программного обеспечения невозможно из-за большого количества возможных комбинаций входных данных. Для повышения качества программного продукта предлагается использовать комплексный подход, включающий тестирование на различных этапах разработки. Особое внимание уделено важности тестирования в реальных условиях эксплуатации, которое позволяет выявить трудновоспроизводимые ошибки. Статья подчёркивает необходимость системного применения различных методов для достижения высокого уровня качества ПО.*

Ключевые слова: *стратегия тестирования программного обеспечения, поведенческое тестирование, структурное тестирование, комплексный подход, качество программного обеспечения, дефекты, реальное тестирование, стратегии тестирования.*

Abstract: *The article discusses modern software testing strategies, including behavioral and structural testing. The main features of each method, their advantages and limitations are described. It is shown that exhaustive software testing is impossible due to a large number of possible combinations of input data. To improve the quality of a software product it is suggested to use an integrated*

approach including testing at different stages of development. Special attention is paid to the importance of testing in real operating conditions, which allows detecting errors that are difficult to reproduce. The article emphasizes the necessity of systematic application of various methods to achieve a high level of software quality.

Keywords: *software testing strategy, behavioral testing, structural testing, integrated approach, software quality, defects, real-world testing, testing strategies.*

Современные программные продукты зачастую создаются в условиях сжатых сроков и ограниченного финансирования. Программирование, которое когда-то считалось искусством, теперь стало повседневной работой для миллионов специалистов. Однако, в стремлении быстро завершить проекты, разработчики часто упускают из виду важность обеспечения качественного уровня своих продуктов, что в итоге может привести к ненужным рискам для пользователей. Ключевая причина, подтверждающая необходимость внедрения тестирования в разработку программного обеспечения, заключается в снижении непредвиденных расходов как для разработчиков, так и для конечных пользователей. Эти расходы возникают из-за сбоев в разработке и использовании ПО, вызванных необходимостью исправления дефектов. Ошибки, выявленные и устраненные на начальных этапах разработки, обходятся значительно дешевле как для разработчиков, так и для клиентов, чем те, которые обнаруживаются уже во время эксплуатации продукта. Кроме того, тестирование играет важную роль, предоставляя данные о допущенных в ходе разработки ошибках. Оперативное информирование разработчиков и менеджеров проектов о таких ошибках значительно снижает вероятность их повторного появления, что, в свою очередь, улучшает общее качество программного обеспечения.

В данном исследовании объектом анализа выступают современные стратегии тестирования, применяемые в процессе разработки программного

обеспечения. Основная цель исследования — это изучение и систематизация существующих методов тестирования программных продуктов.

Для достижения поставленной цели были сформулированы и выполнены следующие задачи:

- Изучить существующие методы тестирования программного обеспечения.
- Проанализировать преимущества и недостатки этих подходов.
- Разработать единую модель применения различных тестовых методов в технологическом цикле разработки программного обеспечения.

Тестирование программного обеспечения, по своей сути, представляет собой процесс обнаружения ошибок в программе. Естественно, идеальным вариантом было бы организовать процесс так, чтобы выявить все возможные ошибки. Однако для того, чтобы подтвердить, что программа полностью лишена ошибок, необходимо:

- подготовить все возможные комбинации входных данных, включая некорректные;
- выполнить программу на всех возможных вариациях входных данных;
- проанализировать все выходные данные и убедиться, что каждый набор результатов соответствует правильному ответу.

Тем не менее, для реализации такого тестирования потребовались бы огромные затраты времени, финансов и человеческих ресурсов. Очевидно, что даже для небольших программ с несколькими входными параметрами количество тестовых сценариев может достигать миллионов комбинаций. Это подчеркивает, что проведение исчерпывающего тестирования программного обеспечения является невозможным. Следовательно, полное выявление всех ошибок в коде программы также недостижимо. Именно поэтому хаотичный, случайный процесс тестирования не окажет существенного влияния на качество продукта. В условиях ограниченного времени и ресурсов приходится выбирать небольшое подмножество тестовых сценариев, что становится

вынужденной мерой. Такой выбор должен быть основан не на случайности, а на определённой стратегии, которой следует придерживаться в процессе тестирования. Это доказывает, что изучение современных методов системного, а не интуитивного тестирования — это актуальная задача.

На сегодняшний день выделяются две основные, противоположные парадигмы: тестирование на основе поведения и структурное тестирование. Также существует отдельная стратегия — тестирование в реальных условиях.

Поведенческое тестирование базируется на технических спецификациях программного обеспечения и часто называется тестированием «черного ящика» (black-box testing). В рамках этого подхода нет необходимости понимать внутреннее устройство программы. Она воспринимается как некий черный ящик, о котором известно лишь то, что есть входные и выходные данные. Как именно программа обрабатывает эти входные данные, остаётся неизвестным. В процессе такого тестирования на основе требований к ПО формируется набор входных данных, на основе которого прогнозируется ожидаемое поведение программы. Затем выполняется тестирование, результаты которого сравниваются с предсказанными.

К методам поведенческого тестирования относятся тестирование управляющих потоков, потоков данных, доменов, синтаксиса, систем с конечным числом состояний и циклов.

- Можно выделить следующие характерные черты методов поведенческого тестирования: Способность проверять как систему в целом, так и её отдельные модули и интерфейсы между ними.
- Возможность выполнения тестов без доступа к исходному коду, что ускоряет процесс тестирования и делает его более объективным.
- Тесты базируются на спецификациях, а не на исходном коде, что устраняет необходимость корректировать тестовые наборы при изменениях в коде.

- Простота автоматизации и выполнения регрессионного тестирования на основе тестов, разработанных по стратегии «черного ящика».

Некоторые редко встречающиеся сценарии могут остаться непроверенными, так как тесты чаще всего покрывают основную функциональность.

Структурное тестирование основывается на внутреннем устройстве программного обеспечения. Этот подход также известен как тестирование «белого ящика» или «прозрачного ящика». В этом случае тестирование проводится на основе понимания того, как программа устроена изнутри. Программа рассматривается как прозрачный ящик, где известны не только входные и выходные данные, но и механизм их преобразования.

Структурное тестирование ещё называют тестированием с покрытием логики. Оно схоже с тестированием потоков управления из стратегии «черного ящика», однако здесь предоставляется детальная информация о логике программы, поскольку тестировщик имеет доступ к коду.

Набор тестовых данных формируется исходя из структуры программы и расположения таких элементов, как циклы и условные операторы. Основная задача этих данных — обеспечить необходимый уровень покрытия программного кода, что подразумевает выполнение операторов программы при обработке входных данных, а также процент выполненных операторов относительно общего их числа. Предсказание результатов сводится к определению правильного направления потока управления и верного состояния внутренних переменных на каждом этапе выполнения программы.

Ключевыми методами структурного тестирования являются покрытие операторов программы, покрытие ветвлений и покрытие условий.

К особенностям структурного тестирования можно отнести следующие:

- Низкие затраты на устранение дефектов. Локализация ошибки в конкретном модуле программы предотвращает её распространение на другие части и минимизирует усилия, необходимые для её исправления.

- Обеспечивается полное покрытие исходного кода, благодаря чему даже редко встречающиеся сценарии не останутся непроверенными.
- Возможность отслеживать поток управления и целостность данных во время выполнения программы с использованием контрольных точек и отладочных инструментов.
- Зависимость тестов от исходного кода, что вынуждает тестировщика постоянно обновлять тесты при изменениях в программе.
- Необходимость глубокого понимания кода со стороны тестировщика, что увеличивает затраты на ресурсы для тестирования.
- Трудности при тестировании системы целиком — чаще всего тесты разрабатываются для отдельных модулей.

Тестирование в реальных условиях, как правило, проводится непосредственно в месте эксплуатации программного продукта самим пользователем или совместно с заказчиком. В этом случае выполняются реальные сценарии использования программы, подготовленные заказчиком . Эти сценарии обычно не являются повторением тестов, разработанных по стратегии «черного ящика», а представляют собой комбинацию определённого подмножества таких тестов.

К ключевым особенностям тестирования в реальных условиях можно отнести следующие:

- Возможность проведения проверки в реальной эксплуатационной среде на различных конфигурациях систем.
- Высокие затраты на исправление ошибок. Некорректная работа программы может нанести серьёзный ущерб пользователю, а после устранения дефектов программа должна будет пройти повторный цикл тестирования.
- Потребность в поддержке со стороны специалистов по программному обеспечению в виде консультаций.

Анализ современных методов тестирования программного обеспечения показал, что не существует универсального способа полностью устранить все ошибки в ПО. Каждый метод нацелен на обнаружение определённого типа дефектов. Таким образом, для достижения приемлемого уровня качества программного продукта рекомендуется использовать комбинацию различных методов тестирования.

На этапе разработки отдельных модулей наиболее эффективным является применение стратегии структурного тестирования. Тестирование по методу «прозрачного ящика» следует рассматривать как неотъемлемую часть программирования. Именно разработчику следует проводить тестирование каждого модуля сразу после его написания, так как он лучше всех понимает свой код и имеет необходимые инструменты для его отладки.

На этапе интеграции модулей в единую систему целесообразно использовать подход поведенческого тестирования, который позволяет оценить работоспособность всей системы в целом. Такое тестирование должны проводить не программисты, а специалисты по тестированию, поскольку они не участвовали в написании исходного кода и не знают его внутреннего устройства. Их задача — выявить ошибки в соответствии с ожидаемыми требованиями, а не с самим процессом работы программы. Это помогает избежать предвзятости программиста, который может быть слишком уверен в правильности своего кода.

Только после успешного прохождения всех предыдущих этапов тестирования программный продукт может быть подвергнут проверке в реальных условиях эксплуатации. Это помогает выявить те ошибки, которые сложно воспроизвести в условиях ограниченного набора ресурсов для тестирования. В работе были рассмотрены ключевые стратегии, используемые при тестировании современного программного обеспечения. Было установлено, что полное тестирование невозможно из-за огромного числа возможных комбинаций и перестановок входных данных. Именно по этой

причине в тестовый набор выбирается лишь небольшая часть этих данных. Для достижения максимального качества и выявления большего числа дефектов тестирование следует проводить с опорой на существующие методы.

На сегодняшний день ещё не созданы универсальные методы тестирования программного обеспечения. В процессе исследования было выявлено, что каждый метод направлен на обнаружение определённого типа ошибок. Поэтому для обеспечения высокого качества ПО целесообразно применять комплексные подходы к тестированию, интегрируя их на разных этапах разработки программного обеспечения.

Библиографический список

1. Тамре Л. Основы тестирования ПО. – М.: Вильямс, 2003. – 359 с.
2. Липаев В.В. Основы тестирования программ. – М.: Радио и связь, 1986. – 295 с.
3. Бейзер Б. Функциональное тестирование программного обеспечения: Подход "черного ящика". – СПб.: Питер, 2004. – 318 с.
4. Майерс Г. Искусство тестирования ПО. Нью-Йорк: John Wiley & Sons, 2004. 234 с.
5. Блэк Р. Организация процесса тестирования программного обеспечения. Редмонд: Microsoft Press, 1999. 381 с.
6. Канер С., Фолк Д., Кек Нгуен Е. Руководство по тестированию программного обеспечения. – Киев: ДиаСофт, 2000. – 544 с.