ТЕХНОЛОГИИ АВТОМАТИЗАЦИИ ТЕСТИРОВАНИЯ И ОТЛАДКИ ПО

Автоматизация тестирования

Автоматизированное тестирование программного обеспечения — часть процесса тестирования на этапе контроля качества в процессе разработки программного обеспечения. Оно использует программные средства для выполнения тестов и проверки результатов выполнения, что помогает сократить время тестирования и упростить его процесс.

Подходы

Существует два основных подхода к автоматизации тестирования: тестирование на уровне кода и тестирование пользовательского интерфейса (в частности, GUI-тестирование). К первому типу относится, в частности, модульное тестирование. Ко второму — имитация действий пользователя - функциональное тестирование (с помощью специальных тестовых фреймворков.)

- **Три уровня автоматизации тестирования**
- Условно, тестируемое приложение можно разбить на 3 уровня:
- **Unit Tests Layer**
- **Functional Tests Layer (Non-UI)**
- **GUI Tests Layer**
- Для обеспечения лучшего качества продукта, рекомендуется автоматизировать все 3 уровня. Рассмотрим более детально стратегию автоматизации тестирования на основе трехуровневой модели:

- **Уровень модульного тестирования (Unit Test layer)**
  - Под автоматизированными тестами на этом уровне понимаются **Компонентные или Модульные тесты** написанные разработчиками. Тестировщикам никто не запрещает писать такие тесты, которые будут проверять код, конечно же, если их квалификация позволяет это. Наличие подобных тестов на ранних стадиях проекта, а также постоянное их пополнение новыми тестами, проверяющими «баг фиксы», убережет проект от многих серьезных проблем.
- **Уровень функционального тестирование (Functional Test Layer non-ui)**
  - Как правило не всю бизнес логику приложения можно протестировать через GUI слой. Это может быть особенностью реализации, которая прячет бизнес логику от пользователей. Именно по этой причине по договоренности с разработчиками, для команды тестирования может быть реализован доступ напрямую к функциональному слою, дающий возможность тестировать непосредственно бизнес логику приложения, минуя пользовательский интерфейс.
- **Уровень тестирования через пользовательский интерфейс (GUI Test Layer)**
  - На данном уровне есть возможность тестировать не только интерфейс пользователя, но также и функциональность, выполняя операции вызывающую бизнес логику приложения. С нашей точки зрения, такого рода сквозные тесты дают больший эффект нежели просто тестирование функционального слоя, так как мы тестируем функциональность, эмулируя действия конечного пользователя, через графический интерфейс.

Автоматизация отладки ПО

- **Отладка** — этап разработки компьютерной программы, на котором обнаруживают, локализуют и устраняют ошибки.

- Поиск некоторых ошибок, содержащихся в структуре данных и, в особенности, в логической структуре программы, можно автоматизировать. Автоматизированный анализ программы на предмет выявления ошибок в ней производится без ее выполнения на ЭВМ и поэтому попадает в категорию «статических».
- Система автоматизации отладки программного обеспечения (САО ПО) представляет собой совокупность программных средств, предназначенных для автоматизации процессов установления факта правильности функционирования разработанных программных изделий, а также для обнаружения, локализации и устранения ошибок в алгоритмах и программах. САО ПО выполняет следующие функции:
- - контроль правильности тестов и заданий на отладку, составленных на входных языках отладки, и выдача информации о месте и характере ошибок;
- - выполнение отладочных заданий при помощи системы трансляции заданий, тестов и их исполнения;
- - моделирование и интерпретация системы команд управляющей ЭВМ в тех случаях, когда отладка программного обеспечения выполняется на вычислительной машине с иной системой команд;
- - выдача оператору результатов отладки и необходимых промежуточных данных на языке отладки после их предварительной обработки;
- - корректировка отлаживаемой программы по заданию оператора с целью исправления обнаруженных ошибок.

- Ниже перечислены примеры ошибок, которые можно обна ружить автоматизирующими анализаторами программ.
- 1. Переменные не описаны или описаны неправильно.
- 2. Переменные используются прежде, чем им было присвоено значение, или присвоенное значение никогда не используется.
- 3. Используются недопустимые языковые формы (например, арифметика с разнотипными переменными).
- 4. Нарушение соглашений о наименованиях (переменных, подпрограмм, меток операторов...).
- 5. Переусложненная структура (циклы или условные операторы слишком глубоко или неправильно вложены).
- 6. Проверка аргументов подпрограммы (соответствие формального и фактического значений).
- 7. Противоречия в дереве вызываемых подпрограмм (не пре дусмотренные циклы).
- 8. Противоречивость набора глобальных данных (общих блоков).
- 9. Невыполнимые условия.
- 10. Потеря управления (например, оператором GO TO пере дано управление на несуществующий оператор).
- 11. Незамкнутая логика.
- 12. Ошибочная логика (например, потенциально бесконечные циклы).

- Ниже перечислены примеры ошибок, которые можно обна ружить автоматизирующими анализаторами программ.
- 1. Переменные не описаны или описаны неправильно.
- 2. Переменные используются прежде, чем им было присвоено значение, или присвоенное значение никогда не используется.
- 3. Используются недопустимые языковые формы (например, арифметика с разнотипными переменными).
- 4. Нарушение соглашений о наименованиях (переменных, подпрограмм, меток операторов...).
- 5. Переусложненная структура (циклы или условные операторы слишком глубоко или неправильно вложены).
- 6. Проверка аргументов подпрограммы (соответствие формального и фактического значений).
- 7. Противоречия в дереве вызываемых подпрограмм (не пре дусмотренные циклы).
- 8. Противоречивость набора глобальных данных (общих блоков).
- 9. Невыполнимые условия.
- 10. Потеря управления (например, оператором GO TO пере дано управление на несуществующий оператор).
- 11. Незамкнутая логика.
- 12. Ошибочная логика (например, потенциально бесконечные циклы).

Вывод

- Автоматизация тестирования и отладки программного обеспечения позволяют быстро и надежно тестировать, и исправлять ошибки в программном обеспечении.