Integration_types

Интеграционное тестирование
– это тип тестирования, при котором программные модули объединяются логически и тестируются как группа. Как правило, программный продукт состоит из нескольких программных модулей, написанных разными программистами.

Цель тестирования - это выявление багов при взаимодействии между этими программными модулями, а также направлен на проверку обмена данными между этими самими модулями.

Стратегии, методологии и подходы в интеграционном тестировании

Программная инженерия задает различные стратегии интеграционного тестирования:

▼ **Подход Большого взрыва.**

Здесь все компоненты собираются вместе, а затем тестируются.

Преимущества:

- Удобно для небольших систем.

Недостатки:

- Сложно локализовать баги.
- Учитывая огромное количество интерфейсов, некоторые из них при тестировании можно запросто пропустить.
- Недостаток времени для группы тестирования, т.к тестирование интеграции может начаться только после того, как все модули спроектированы.
- Поскольку все модули тестируются одновременно, критические модули высокого риска не изолируются и тестируются в приоритетном порядке. Периферийные модули, которые имеют дело с пользовательскими интерфейсами, также не изолированы и не проверены на приоритет.

▼ **Инкрементальный подход**

В данном подходе тестирование выполняется путем объединения двух или более логически связанных модулей. Затем добавляются другие связанные

модули и проверяются на правильность функционирования. Процесс продолжается до тех пор, пока все модули не будут соединены и успешно протестированы.

Поэтапный подход, в свою очередь, осуществляется двумя разными методами:

- Снизу вверх
- Сверху вниз

▼ **Заглушка и драйвер**

Инкрементальный подход осуществляется с помощью фиктивных программ, называемых заглушками и драйверами. Заглушки и драйверы не реализуют всю логику программного модуля, а только моделируют обмен данными с вызывающим модулем.

Заглушка: вызывается тестируемым модулем.

Драйвер: вызывает модуль для тестирования.

▼ **Интеграция «снизу вверх»/ “восходящий подход”**

В восходящей стратегии каждый модуль на более низких уровнях тестируется с модулями более высоких уровней, пока не будут протестированы все модули. Требуется помощь драйверов для тестирования

Преимущества:

- Проще локализовать ошибки.
- Не тратится время на ожидание разработки всех модулей, в отличие от подхода Большого взрыва.

Недостатки:

- Критические модули (на верхнем уровне архитектуры программного обеспечения), которые контролируют поток приложения, тестируются последними и могут быть подвержены дефектам.
- Не возможно реализовать ранний прототип

▼ **Интеграция «сверху вниз»/ “нисходящий подход”**

При подходе «сверху вниз» тестирование, что логично, выполняется сверху вниз, следуя потоку управления программной системы. Используются заглушки для тестирования.

Преимущества:

- Проще локализовать баги.
- Возможность получить ранний прототип.
- Критические Модули тестируются на приоритет; основные недостатки дизайна могут быть найдены и исправлены в первую очередь.

Недостатки:

- Нужно много пней.
- Модули на более низком уровне тестируются неадекватно

▼ **Сэндвич (гибридная интеграция)**

Эта стратегия представляет собой комбинацию подходов «сверху вниз» и «снизу вверх». Здесь верхнеуровневые модули тестируются с нижнеуровневыми, а нижнеуровневые модули интегрируются с верхнеуровневыми, соответственно, и тестируются. Эта стратегия использует и заглушки, и драйверы.

Алгоритм интеграционного тестирования:

1. Подготовка план интеграционных тестов
2. Разработка тестовых сценариев.
3. Выполнение тестовых сценариев и фиксирование багов.
4. Отслеживание и повторное тестирование дефектов.
5. Повторять шаги 3 и 4 до успешного завершения интеграции.

▼ **Атрибуты Интеграционного тестирования**

- Методы / Подходы к тестированию (об этом говорили выше.
- Области применения и Тестирование интеграции.
- Роли и обязанности.
- Предварительные условия для Интеграционного тестирования.
- Тестовая среда.
- Планы по снижению рисков и автоматизации.

Критерии старта и окончания интеграционного тестирования

Критерии входа и выхода на этап Интеграционного тестирования, независимо от модели разработки программного обеспечения

▼ **Критерии старта:**

- Модули и модульные компоненты
- Все ошибки с высоким приоритетом исправлены и закрыты
- Все модули должны быть заполнены и успешно интегрированы.
- Наличие плана Интеграционного тестирования, тестовый набор, сценарии, которые должны быть задокументированы.
- Наличие необходимой тестовой среды

▼ **Критерии окончания:**

- Успешное тестирование интегрированного приложения.
- Выполненные тестовые случаи задокументированы
- Все ошибки с высоким приоритетом исправлены и закрыты
- Технические документы должны быть представлены после выпуска Примечания.

Рекомендации

- Сначала определите интеграционную тестовую стратегию, которая не будет противоречить вашим принципам разработки, а затем подготовьте тестовые сценарии и, соответственно, протестируйте данные.
- Изучите архитектуру приложения и определите критические модули. Не забудьте проверить их на приоритет.
- Получите проекты интерфейсов от команды разработки и создайте контрольные примеры для проверки всех интерфейсов в деталях. Интерфейс к базе данных / внешнему оборудованию / программному обеспечению должен быть детально протестирован.
- После тестовых случаев именно тестовые данные играют решающую роль.

- Всегда имейте подготовленные данные перед выполнением. Не выбирайте тестовые данные во время выполнения тестовых случаев.