1 Стратегии тестирования программ на этапе разработки

Стратегия тестирования представляет собой описание общего подхода к тестированию и целей тестирования.

Стратегия тестирования — это план проведения работ по тестированию системы или её модуля, учитывающий специфику функциональности и зависимости с другими компонентами системы и платформы. Стратегия определяет типы тестов, которые нужно выполнять для данного функционала системы, включает описание необходимых подходов с точки зрения целей тестирования и может задавать описания или требования к необходимым для проведения тестирования инструментам и инфраструктуре.

Стратегия отвечает на вопросы:

- Как, каким образом тестирование даст ответ, что данный функционал работает?
- Что нужно сделать и чем пользоваться из инструментальных средств, для достижения целей тестирования?
- Когда определённая функциональность будет тестироваться и соответственно когда ожидать получения результатов?

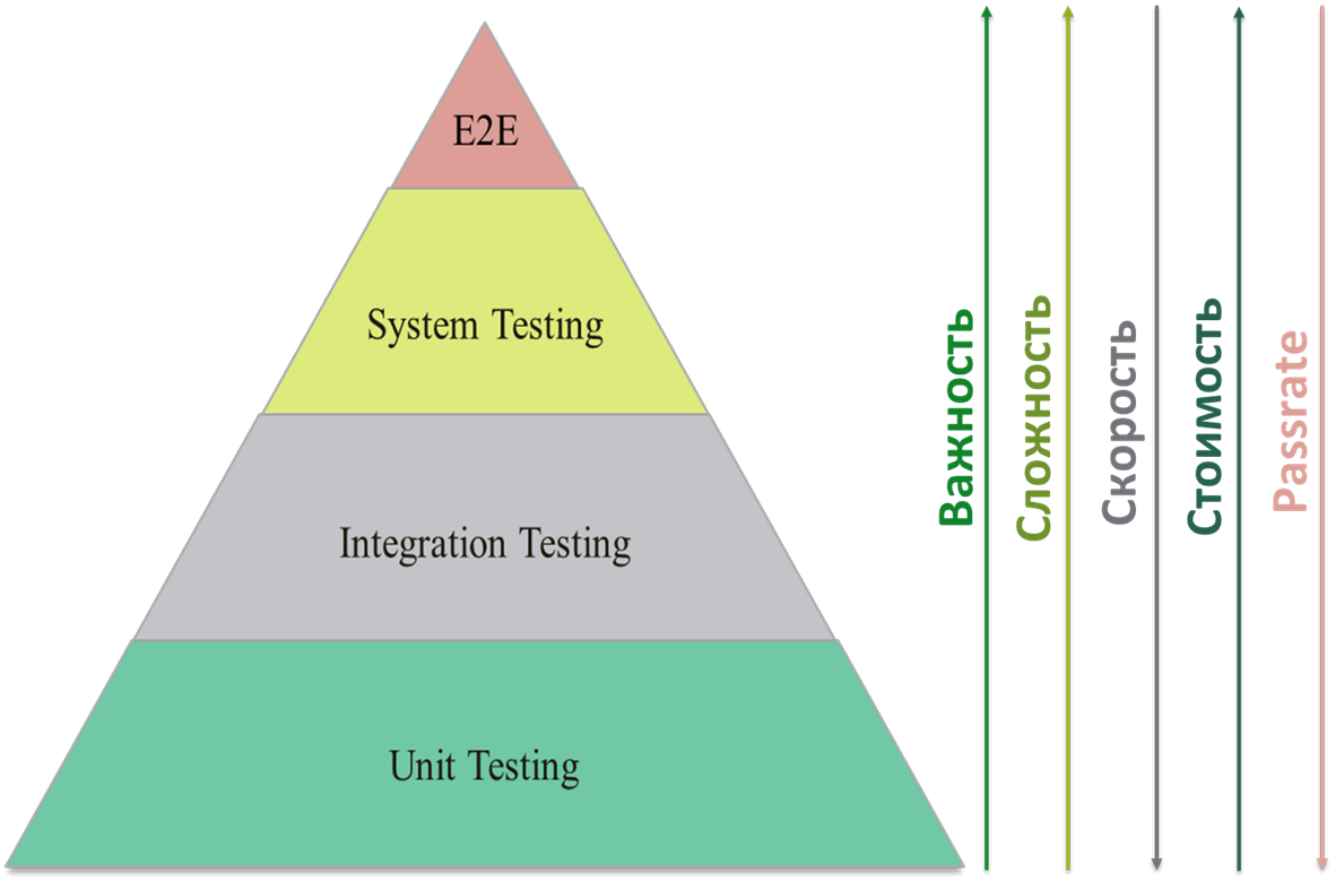
Содержание стратегии будет разным в зависимости от проекта, поэтому нет единого для всех шаблона. Часто используются следующие пункты:

- **Обзор и объем**: объем работ по тестированию (что тестировать и зачем тестировать) и обзор тестируемого продукта;

- **Подход к тестированию** (Test Approach):
  - **Уровни тестирования** (Test levels). Модульное, интеграционное, сквозное (системное).
  - **Виды тестирования** (Test Types). Ручное/автоматизированное, функциональное, тестирование UI, тестирование скорости и надежности, тестирование пользовательского опыта.
  - **Роли и обязанности** (Roles and responsibilities);
  - **Требования к окружениям** (Environment requirements);
    - **Инструменты тестирования**: инструменты, необходимые для проведения тестов (TMS, багтрекинговая система, стек автоматизации);
- **Результаты тестирования** (Test deliverables): документация, которую необходимо создать до, во время и по окончании тестирования;
- **Метрики тестирования** (Testing metrics): метрики, которые следует использовать в проекте для анализа статуса проекта;
- **Матрица отслеживания требований** (RTM);
- **Риски и способы их снижения** (Risk and mitigation): все риски тестирования и план по их снижению;
- **Инструмент отчетности** (Reporting tool): как будут отслеживаться дефекты и проблемы;
- **Результаты тестов** (Test Summary): виды сводных отчетов о тестах, которые будут создаваться, с указанием периодичности. Сводные отчеты о тестах будут генерироваться ежедневно, еженедельно или ежемесячно, в зависимости от критичности проекта.

Уровни тестирования описываются пирамидой тестов. «Пирамида тестов» - метафора, которая означает группировку динамических тестов программного обеспечения по разным уровням. Она также дает представление, какое количество тестов

должно быть в каждой из этих групп. Основной принцип разделения уровней - тест должен быть на том же уровне, что и тестируемый объект. В тесте более высокого уровня вы не тестируете всю условную логику и пограничные случаи, которые уже покрыты тестами более низкого уровня.



Уровни тестирования:

- **Unit/component/program/module testing** - тестируется минимально-атомарный модуль программы, чаще всего это одна функция или метод. Таких тестов должно быть больше всего;
- **Integration testing** - несколько модулей программы тестируются вместе;
- **System testing** - вся программа тестируется полностью;
- **Acceptance testing** - программа принимается заказчиком на соответствие заявленным требованиям либо тестировщики

проходят end-to-end сценарии с точки зрения пользователя;

2 Тестирование черного ящика

Самым высоким уровнем в иерархии подходов к тестированию будет понятие метода. Некоторые виды тестирования могут выполняться методом как черного ящика, так и белого. Если упростить, то отличаются они знанием внутреннего устройства объекта тестирования.

Определения:

- Тестирование методом черного ящика (black box testing): Тестирование, функциональное или нефункциональное, без знания внутренней структуры компонента или системы (ISTQB).
- Тестирование на основе спецификации (specification-based testing): Тестирование, основным базисом которого являются внешние вводы и выводы элемента тестирования, обычно на основе спецификации, а не ее реализация в исходном коде или исполнимом программном обеспечении. (ГОСТ 56920)

Тестирование методом «черного ящика» - это стратегия, в которой тестирование основано исключительно на требованиях и спецификациях, при этом мы не знаем, как устроена внутри тестируемая система и работаем исключительно с внешними интерфейсами тестируемой системы или компонента.
Тестирование черного ящика может быть применено на всех уровнях - модульном, интеграционном, системном и приемочном.

Методом "черного ящика" можно проводить функциональное и нефункциональное тестирование.
Functional Testing: этот тип касается функциональных требований или спецификаций приложения (functional requirements or

specifications). Здесь различные действия или функции системы тестируются путем предоставления входных данных и сравнения фактического выхода с ожидаемым выходом.
Non-Functional Testing: Помимо функциональности требований, есть несколько нефункциональных аспектов, которые необходимо протестировать, чтобы улучшить качество и производительность приложения (производительность, устойчивость, масштабируемость).

Преимущества Black box testing:

- Тестировщику не обязательно иметь технический опыт. Важно проводить тестирование, оказываясь на месте пользователя и думая с его точки зрения;
- Тестирование можно начинать после завершения разработки проекта / приложения. И тестировщики, и разработчики работают независимо, не мешая друг другу;
- Это более эффективно для больших и сложных приложений;
- Дефекты и несоответствия можно выявить на ранней стадии тестирования;

Недостатки Black box testing:

- Без каких-либо технических или программных знаний есть вероятность пропустить возможные условия тестируемого сценария;
- В оговоренное время есть вероятность протестировать не все входные и выходные значения;
- Полный Test Coverage невозможен для больших и сложных проектов;

Парадигмы тестирования методом черного ящика (Paradigms of Black Box Software Testing)

Парадигма создает основное направление мышления - она дает понимание и направление для дальнейших исследований или работы. Парадигма тестирования будет определять типы тестов, которые (для человека, работающего в рамках этой парадигмы) актуальны и интересны. Список парадигм:

- **Domain driven**
  - Ключевые идеи:
    - «Попробуйте диапазоны и варианты»;
    - «Разделите мир на классы»;
  - Основной вопрос или цель:
    - Стратегия стратифицированной выборки. Разделите большое пространство возможных тестов на подмножества. Выберите лучших представителей из каждого набора;
- **Stress driven**
  - Ключевые идеи:
    - «Сокруши продукт»;
      - «Проведи его через отказы;
  - Основной вопрос или цель:
    - Узнать о возможностях и слабых сторонах продукта, проведя его через отказ и за его пределами. Что сбои в экстремальных случаях говорят нам об изменениях, необходимых в работе программы в нормальных случаях?
- **Specification driven**
  - Ключевые идеи:
    - «Проверяйте каждое требование»;
  - Основной вопрос или цель:
    - Проверяйте соответствие (conformance) продукта каждому заявлению в каждой спецификации, документе с требованиями и т. д.;

- **Risk driven**
  - Ключевые идеи:
    - «Сначала найди наибольшие ошибки»;
  - Основной вопрос или цель:
    - Расставьте приоритеты при тестировании с точки зрения относительного риска различных областей или проблем, которые мы могли бы проверить;
- **Function Testing**
  - Ключевые идеи:
    - «Модульное тестирование черного ящика»;
    - Основной вопрос или цель:
    - Тщательно проверяйте каждую функцию по очереди;
- **Regression Testing**
  - Ключевые идеи:
    - «Повторить тестирование после изменений»;
    - Основной вопрос или цель:
    - Управляйте рисками, связанными с тем, что (а) исправление ошибки не устраняет ошибку или (б) исправление (или другое изменение) имело побочный эффект (side effect);
- **Scenario / use case / transaction flow**
  - Ключевые идеи:
    - «Делай что-нибудь полезное и интересное»;
    - «Делайте одно за другим»;
    - Основной вопрос или цель:
      - Сложные случаи, отражающие реальное использование;