НАДЁЖНОСТЬ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Лекция 11:
Подготовка к тестированию

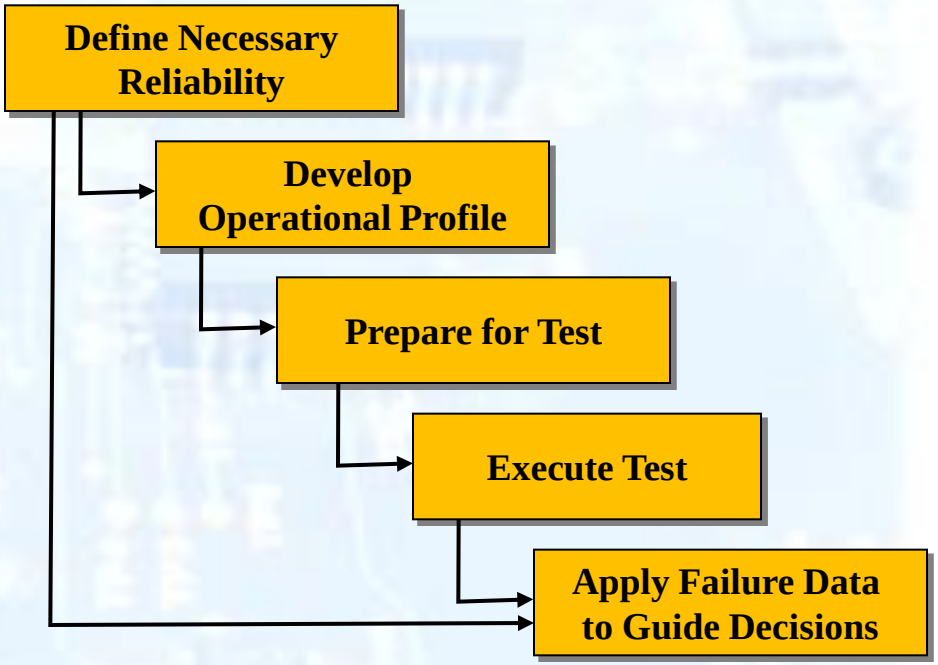
ВМиК МГУ им. М.В. Ломоносова,
Кафедра АСВК, Лаборатория Вычислительных Комплексов
Ассистент Волканов Д.Ю.

План лекции

- Прямые и косвенные переменные
- Полевые испытания и регрессионное тестирования
- Что такое сценарий теста?
- Как управлять сценариями теста?
- Процесс тестирования
- Классы эквивалентности и граничные условия
- Как разрабатывать сценарии тестов?
- Подготовка успешного тестирования

SRE: процесс

- 5 шагов в SRE процессе:
  - Определить необходимый уровень надежности
  - Разработать функциональный разрез
  - Подготовка для тестирования
  - Выполнить тестирование
  - Проанализировать данные об ошибках и использовать их для принятия решений



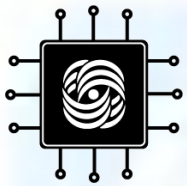
Основные понятия (1)

- **Прогон:** наименьшее задание, которое может быть вызвана внешним воздействием. Прогон ассоциируется с входным состоянием (набором входных переменных). Прогоны с одинаковыми входными состояниями называются прогонами одного типа
- **Операция** – группа типов прогона с точки зрения её архитектуры программы
- **Тестовая процедура** – программа, которая устанавливает переменные окружения и вызывает случайные тестовые сценарии в случайные моменты времени

Основные понятия (2)

Типы тестов

- **Тест на поиск ошибок:** выполнение одной операции, максимально независимое от других операций
- **Нагрузочный тест:** тестирование на реальных данных с подсчётом показателей производительности
- **Регрессионный тест:** проверка функциональности после каждого значительного изменения в коде



Основные понятия (3)

- **Прямые входные переменные явно управляют поведением программы**
  - **Пример:** аргументы, введённые данные
  - Важны во время тестирования функциональности
- **Косвенные входные переменные** могут только влиять на результат операций
  - **Пример:** сетевой траффик, переменные окружения
  - Важны во время нагрузочного тестирования

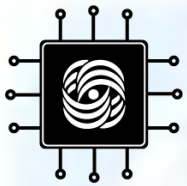
Пример

- Прямые и косвенные входные переменные и их эффект на результат операции

Direct	Indirect
Originator = 201 908 5577	Operational mode = prime hours
Forwardee = 908 555 1212	Database state: signified by time
Billing type = per call	Resource state: signified by time
Dialing type = standard	
Screening = yes	

Что такое тестовый сценарий?

- Тестовый сценарий – спецификация прогона в виде перечисления имён входных переменных и их значений
- Тестовый сценарий не зависит от режима функционирования программы
- Один и тот же тестовый сценарий может выполняться в разных режимах функционирования. Поэтому тестовый сценарий может соответствовать сразу нескольким прогонам с разным потенциально возможными отказами



Хороший тестовый сценарий

- Существует разумная вероятность обнаружения ошибки
- Не обладает избыточностью
- Лучше других подходит к сценарию
- Не слишком прост и не слишком сложен
- Делает ошибку в программе очевидной

Пример

- Тестовый сценарий определяется входным состоянием
- В теории (не обязательно на практике) можно провести прогоны для любого входного состояния

Direct
Originator = 201 908 5577
Forwardee = 908 555 1212
Billing type = per call
Dialing type = standard
Screening = yes

Тестовые сценарии и прогоны

- Спецификация косвенных входных переменных даёт тестовому сценарию недостающий контекст для того, чтобы тот стал прогоном
- Во время тестов функциональности влияние косвенных входных переменных на результат должно быть сведено к минимуму, чтобы убедиться в реальной надёжности системы
- Использование косвенных входных переменных важно во время нагрузочных тестов

Процедура

Состоит из двух этапов:

1) **Подготовка тестовых сценариев (управление сценариями)**
   - Использование реальных данных и функциональных профилей
2) **Подготовка тестовых процедур**
   - Создание одной процедуры для каждого режима функционирования, используя его профиль и частоты использования операций

Управление тестовыми сценариями

Подготовка тестовых сценариев включает:

1. Подсчёт числа новых тестовых сценариев, которые нужно создать для данной версии программы
2. Распределение сценариев среди подсистем, которые необходимо тестировать [уровень системы]
3. Распределение сценариев между новыми операциями внутри каждой подсистемы [уровень операций]
4. Описание новых сценариев тестирования
5. Добавление новых сценариев

1. Подсчёт числа новых сценариев

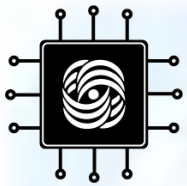
- **Методология:**
  - Число тестовых сценариев зависит от:
    - ***Времени:***

    (доступное время × доступные люди)/
    (среднее время создания сценария)
    - ***Стоимости:***

    (доступный бюджет) / (средний расход на создание сценария)
  - Число выбирается как минимальное из рассчитанных значений

2. Распределение сценариев между подсистемами (1)

Большинство сценариев нужно ориентировать на поиск ошибок в разрабатываемом продукте



2. Распределение сценариев между подсистемами (2)

- Большинство сценариев нужно ориентировать на поиск ошибок в разрабатываемом продукте
- Создавайте больше сценариев для операций с большей вероятностью использования
- Не создавайте сценариев для приобретённых компонентов, если они уже использовались ранее. В противном случае создайте для них число сценариев, соответствующее риску их использования в контексте надёжности системы
- Если ваша программа значительно зависит от функций операционной системы и её надёжность неизвестна, то на её проверку необходимо выделить 20-30% сценариев

3. Распределение сценариев между операциями (1)

1. Конвертируйте графические представления профилей функционирования в табличное представление
2. Найдите новые критические операции, которые редко встречаются, и выделите для каждой из них приблизительное число требуемых сценариев
3. Выделите по одному сценарию для каждой новой редко встречающейся операции
4. Распределите оставшееся число сценариев между остальными новыми операциями

3. Распределение сценариев между операциями (2)

Найдите *пороговую вероятность*: 0,5 делёные на число новых сценариев. Выделите по одному тесту для каждой новой нечастой операции

Выделите по 2-4 сценария для каждой нечастой критической операции

Распределите оставшиеся тестовые сценарии между оставшимися новыми операциями

3.1 Критические операции

- Критическая операция – операция, получение результата которой принципиально важно, а отказ выполнения ведёт к значительному ущербу (определённому в соответствии с классами тяжести отказа)
- Пример критической операции – остановка реактора атомной электростанции в случае его перегрева. Эта операция используется крайне редко, но очень важна
- Выделяйте только редкие критические операции, так как остальные будут проверены за счёт их использования сразу во многих сценариях выполнения программы

3.2 Нечастые операции

- ***Нечастые операции*** – те которые бы не проверялись в случае равномерного распределения тестовых сценариев из-за очень маленькой вероятности их использования
- Установите порог вероятности использования операции. Хороший выбор его значения равен отношению 0.5 к количеству новых тестовых сценариев
- Выделите по одному тестовому сценарию для всех операций с вероятностью ниже порогового значения

Пример (1)

- Число тестовых сценариев: 500
- Первая версия программы. Все сценарии новые
- Пусть в программе есть одна критическая операция. Выделяем для ей проверки 2 сценария
- Рассчитываем пороговое значение вероятности использования: 0.5 / 500 = 0.001
- Пусть число операций с вероятностью ниже пороговой равно двум. Выделяем по тесту на каждую из них
- Распределяем оставшиеся сценарии 500 - (2+2) = 496 между остальными операциями в соответствии с их вероятностями использования

Пример (2)

- Вероятности операций для нормального режима функционирования

Operation	Occurrence probability
Process voice call, no pager, answer	0.18
Process voice call, no pager, no answer	0.17
Process voice call, pager, answer	0.17
Process fax call	0.15
Process voice call, pager, answer on page	0.12
Process voice call, pager, no answer on page	0.10
Phone number entry	0.10
Audit section of phone number database	0.009
Add subscriber	0.0005
Delete subscriber	0.0005
Recover from hardware failure	0.000001
Total	1.0

Нечастые операции с вероятностью ниже пороговой

Критическая операция

Пример (2)

- Вероятности операций для нормального режима функционирования

Operation	Occurrence probability
Process voice call, no pager, answer	0.18
Process voice call, no pager, no answer	0.17
Process voice call, pager, answer	0.17
Process fax call	0.15
Process voice call, pager, answer on page	0.12
Process voice call, pager, no answer on page	0.10
Phone number entry	0.10
Audit section of phone number database	0.009
Add subscriber	0.0005
Delete subscriber	0.0005
Recover from hardware failure	0.000001
Total	1.0

Распределяем сценарии в соответствии с вероятностями

Нечастые операции с вероятностью ниже пороговой

Критическая операция

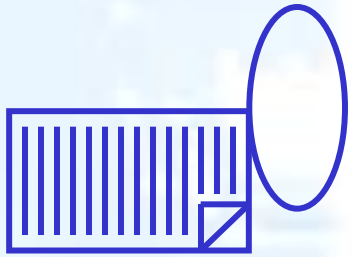
4. Задание тестовых сценариев (1)

- **Простое определение:**
  - Тестовый сценарий – спецификация прогона в виде перечисления имён входных переменных и их значений
- **Лучшее определение:**
  - Тестовый сценарий – случай использования, описанный в терминах тестовых входных данных, условий выполнения и ожидаемых результатов

Случаи использования должны быть описаны до начала разработки тестовых сценариев

4. Задание тестовых сценариев (2)

- Обычное описание случая использования включает:
  - Имя случая использования
  - Инициатор
  - Цель
  - Предусловия
  - Результат (Постусловия)
  - Детальное описание
  - Исключения и альтернативные пути выполнения

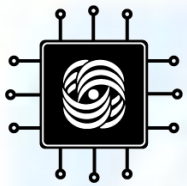


Тестовые сценарии

- Для каждого случая использования должно быть создано не меньше двух сценариев:
  - Один для штатного выполнения, другой – для отказа
- Может создаваться и большее количество сценариев, покрывающих исключения и альтернативные пути выполнения

5. Добавление нового сценария в набор тестов программы (1)

- Набор тестов обычно состоит из старых сценариев (оставшихся после выхода предыдущих версия программ) и новых сценариев (для добавленных функций)
- Для каждого режима функционирования программы необходимо разработать свой профиль функционирования теста, отличающийся от профиля программы:
  - Редким операциям уделяется больше внимания
  - Добавлены новые операции



5. Добавление нового сценария в набор тестов программы (2)

- Для описания профиля функционирования теста для заданного режима начните с:
  - Режима и профиля функционирования (для первой версии программы)
  - Режима и профиля функционирования теста (для последующих версий программы)
- Измените вероятности операций с учётом новых редких критических операций
- Измените вероятности вновь добавленных операций с учётом взаимного влияния операций друг на друга

Тестовая процедура

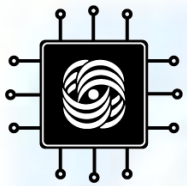
- Тестовая процедура – программа, которая устанавливает переменные окружения и вызывает случайные тестовые сценарии в случайные моменты времени
- Подготавливается по одной тестовой процедуре для каждого режима функционирования

Как описать тестовый сценарий?

- Для заданной операции определите наборы значений прямых входных переменных, для которых программа будет вести себя по одному сценарию
- Тогда тестовый сценарий можно описать как:
  - Комбинацию значений входных переменных
  - Время их передачи на вход программе
- Если число выделенных комбинаций превосходит число доступных сценариев, то сценарии должны создаваться только для их части

Как создать тестовый сценарий

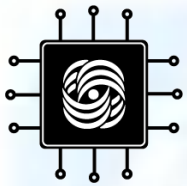
1. На основе пользовательских сценариев и случаев использования
2. На базе классов эквивалентности
   - Подготовка одного теста для класса операций
3. На базе граничных условий
   - Программа, которая не работает на простых условиях, не работает и на граничных условиях
4. На основе диаграммы изменения состояний



1. Классы эквивалентности

Группа сценариев «эквивалентна» если:

- Все они проверяют одну операцию
- Если один сценарий может найти ошибку, остальные скорее всего тоже смогут
- Если один сценарий не может найти ошибку, остальные – тоже не смогут
- Они используют одинаковый набор входных переменных и влияют на значения одинакового набора выходных переменных



Замечание (1)

- Не забывайте о классах эквивалентности для некорректных входных данных

Пример:

Пусть программа принимает на вход любое целое число между 1 и 99

Тогда существует 4 класса эквивалентности:

- Целые числа между 1 и 99 (корректный ввод)
- Любое целое число меньше 1 (некорректный ввод)
- Любое целое число больше 99 (некорректный ввод)
- Не целое число (некорректный ввод)

Замечание (2)

- Обращайте внимание на диапазон значений переменных

Пример:

Пусть программа принимает на вход любое целое число между 1 и 99

Тогда существует 4 класса эквивалентности:

- Сделайте тест для очень больших значений
- Проверьте небольшие отрицательные значения, так как знак может обрабатываться некорректно

Замечание (3)

- Проверяйте переполнение стека рекурсивных программ

Пример:

Пусть программа осуществляет рекурсивный обход дерева в глубину

- Проверьте корректность её работы на большом дереве, вытянутом в список

Замечание (4)

- Если входные данные принадлежат к одной группе, то все числа из этой группы должны включаться в один класс эквивалентности

Пример:

Пусть программа просит пользователя ввести имя

- Программа должна корректно принимать любые буквы алфавита в любом регистре
- Попытайтесь ввести однобуквенное или очень длинное имя
- Проверьте корректность обработки символов не из таблицы ASCII

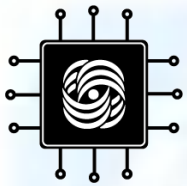
Замечание (5)

- Обращайте внимание на зависимости между входными переменными

Пример:

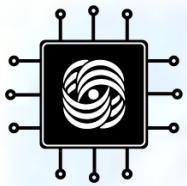
Путь программа принимает на вход значения трёх углов треугольника

- Попытайтесь ввести углы, сумма которых не равна 180 градусам



2. Граничные условия

- Проверьте, корректны ли установленные на диапазоны граничные условия
- Проверьте, могут ли нестрогие знаки быть заменены на строгие и наоборот
- Проверьте корректность ограничения числа итераций циклов
- …



Примеры

Пусть программа ожидает ввода прописной буквы в диапазоне A-Z

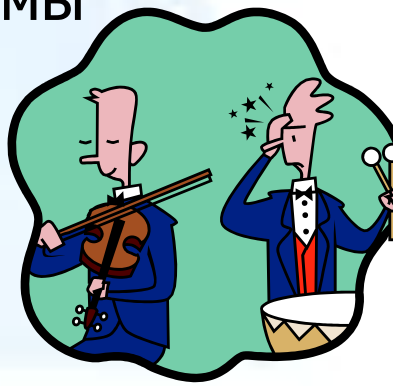
- Проверьте корректность её работы на символах непосредственно перед ‘A’ и сразу после ‘Z’

Пусть программа ожидает ввода строго заданного числа параметров

- Попытайтесь ввести на один параметр меньше и больше

3. Визуальное изменение состояния программы

- Каждое взаимодействие с программой (установка входного параметра, выбор пункта из меню, …) изменяет состояние программы
  - Проверьте все рутинные операции, которые будет совершать пользователь
  - Проверьте любые конфигурации, которые могут как-то повлиять на поведение программы
  - Выберете случайно несколько других операций



Документирование отказа

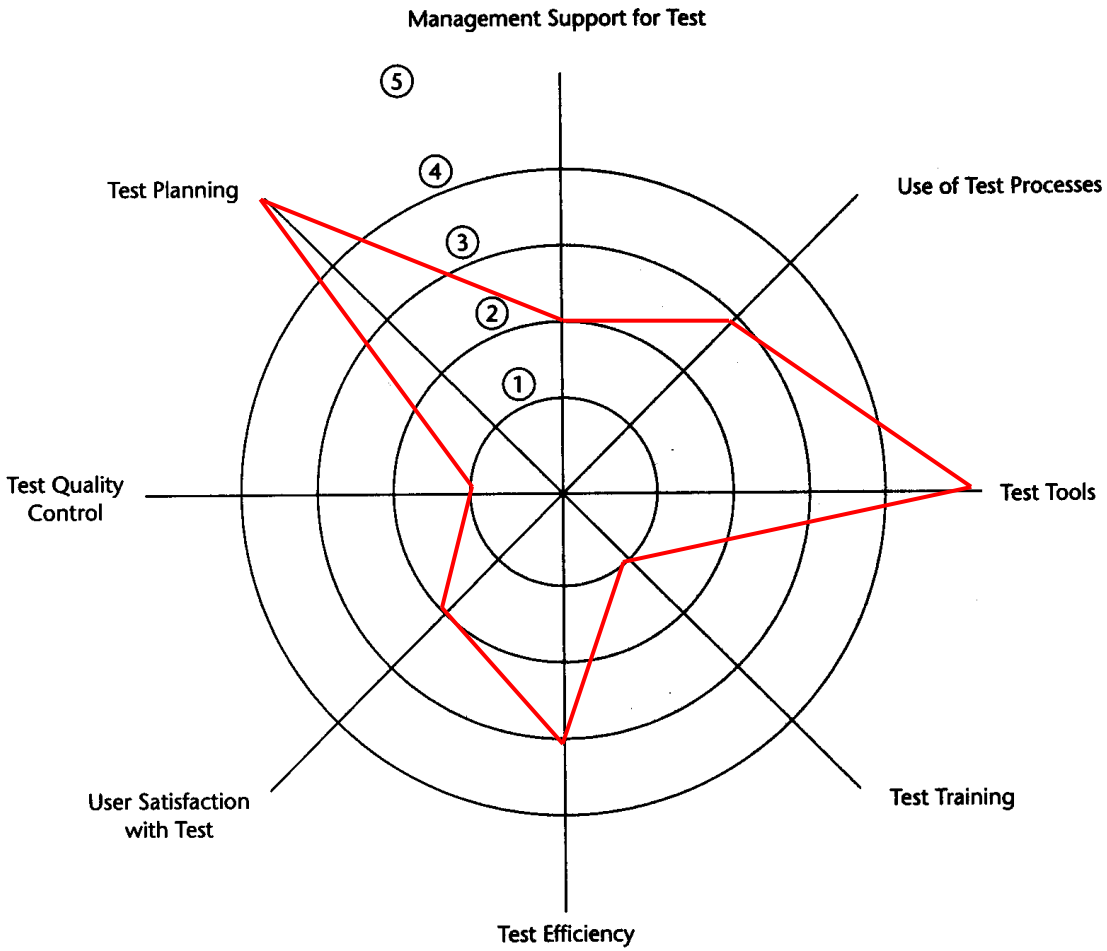
- Запишите полный набор входящих параметров, на которых программа отказала
- Создайте тестовый сценарий, способный воспроизвести отказ
- Найдите класс эквивалентных тестовых сценариев
- Созданный сценарий впоследствии может быть использован в качестве регрессионного

Что необходимо для успешного тестирования?

- Планирование тестирования
- Технологии тестирования
- Средства тестирования
- Управление тестированием
- Обучение построению тестов
- Эффективность тестов
- Контроль качества тестов
- Взаимодействие с пользователями

Диаграмма оценки тестирования

- Успешность тестирования оценивается на основании 8 параметров.

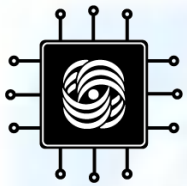


Планирование тестирования

- Соответствует ли план тестирования целям проекта и требованиям, которые к нему предъявляются?
- Вовлечены ли пользователи системы в процесс создания плана тестирования? Включает ли он рекомендации пользователей?
- Изменился ли план тестирования после изменения требований к программе?
- Содержит ли план описание требований к программному обеспечению? Согласны ли пользователи с этими требованиями?

Технологии тестирования

- Существует ли подробная документация по используемым технологиям тестирования? Ведутся ли тесты в соответствии с этой технологией?
- Покрывают ли используемые технологии тестирования весь необходимый спектр мероприятий?
- Ведётся ли разработка механизмов улучшения технологий тестирования? Участвует ли команда в их разработке?



Средства тестирования

- Используются ли автоматизированные средства для генерации и повторного использования тестовых данных?
- Правильно ли выбраны средства тестирования для данного программного обеспечения?
- Умеют ли разработчики тестов пользоваться доступными средствами тестирования?
- Учитываются ли в тестовом плане характеристики средств тестирования?
- Существует ли возможность получения консультации специалистов по работе с конкретными средствами тестирования?

Управление

- Выделено ли достаточное количество ресурсов на выполнение заявленного тестового плана?
- Происходили ли ротации в составе команды тестирования на протяжении разработки проекта?
- Правильно ли распределены ресурсы между отделами разработки ПО и его тестированием?
- Соблюдаются ли командой тестирования существующие технологии?

Обучение

- Предусмотрено ли обучение команды тестирования работе с новыми средствами?
- Завершено ли обучение команды до начала процесса тестирования?
- Разбирается ли команда в теоретических основах технологий тестирования? Понимают ли они зачем проводятся те или иные мероприятия?
- Обладают ли члены команды достаточными знаниями для анализа результатов тестирования?

Эффективность

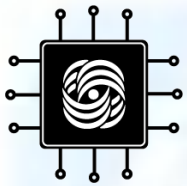
- Учитываются ли классы тяжести отказов во время разработки тестов для их обнаружения?
- Используются ли средства подсчёта статистики результатов тестирования?
- Учитывают ли выделенный бюджет и составленный план возможные отклонения?
- Принимается ли во внимание эффективность использования средств автоматизированного тестирования?
- Учитывается ли вероятность ошибки, подсчитанная на основе предыдущей статистики?

Контроль качества тестирования

- Проводятся ли мероприятия по недопущению повторения совершённых командой тестирования ошибок?
- Проверяется ли разработанный план тестирования на соответствие существующим стандартам?
- Предусмотрены ли способы проверки выполнения плана тестирования?
- Ведётся ли регулярная отчётность?
- Собираются ли резюмирующие внутренние отчёты об общей эффективности тестирования в процессе разработки ПО?

Тестирование и пользователи

- Предоставляется ли пользователям удобный способ выражения своих замечаний по поводу использования продукта?
- Обладают ли пользователи достаточными знаниями для оценки фронта работ по тестированию?
- Ознакомлены ли пользователи с планом тестирования? Одобрен ли ими существующий план?



Спасибо за внимание!