МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«Гжельский государственный университет»** (ГГУ)

Колледж ГГУ

Специальность 09.02.07 Информационные системы и программирование

**Реферат**

**По предмету «Технология разработки программного обеспечения»**

**На тему «Тестовое покрытие»**

ВЫПОЛНИЛ:

Студент группы ИСП-0-17

Скрябин С.И.

ПРОВЕРИЛА:

Прокуронова А.Ю.

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

п. Электроизолятор

2019 г.

# **Тестовое Покрытие**

**Тестовое Покрытие** (**Test Coverage**) - это одна из метрик оценки качества тестирования, представляющая из себя плотность покрытия тестами требований либо исполняемого кода.

Если рассматривать тестирование как "проверку соответствия между реальным и ожидаемым поведением программы, осуществляемая на конечном наборе тестов", то именно этот конечный набор тестов и будет определять тестовое покрытие:

*Чем выше требуемый уровень тестового покрытия, тем больше тестов будет выбрано, для проверки тестируемых требований или исполняемого кода.*

Сложность современного программного обеспечения и инфраструктуры сделало невыполнимой задачу проведения тестирования со 100% тестовым покрытием. Поэтому для разработки набора тестов, обеспечивающего более-менее высокий уровень покрытия можно использовать специальные инструменты либо техники тест дизайна.

Существуют следующие подходы к оценке и измерению тестового покрытия:

1. [**Покрытие требований (Requirements Coverage)**](http://www.protesting.ru/testing/testcoverage.html#requirements) - оценка покрытия тестами функциональных и нефункциональных требований к продукту путем построения матриц трассировки (traceability matrix).
2. [**Покрытие кода (Code Coverage)**](http://www.protesting.ru/testing/testcoverage.html#code) - оценка покрытия исполняемого кода тестами, путем отслеживания непроверенных в процессе тестирования частей программного обеспечения.
3. [**Тестовое покрытие на базе анализа потока управления**](http://www.protesting.ru/testing/testcoverage.html#flow) - оценка покрытия основанная на определении путей выполнения кода программного модуля и создания выполняемых тест кейсов для покрытия этих путей.

**Различия**:  
Метод покрытия требований сосредоточен на проверке соответствия набора проводимых тестов требованиям к продукту, в то время как анализ покрытия кода - на полноте проверки тестами, разработанной части продукта (исходного кода), а анализ потока управления - на прохождении путей в графе или модели выполнения тестируемых функций (Control Flow Graph).

**Ограничения**:  
Метод оценки покрытия кода не выявит нереализованные требования, так как работает не с конечным продуктом, а с существующим исходным кодом.  
Метод покрытия требований может оставить непроверенными некоторые участки кода, потому что не учитывает конечную реализацию.

## Покрытие требований (Requirements Coverage)

Расчет тестового покрытия относительно требований проводится по формуле:

Tcov = (Lcov/Ltotal) \* 100%

где:  
**Tcov –** тестовое покрытие, **Lcov** – количевство требований, проверяемых тест кейсами, **Ltotal** – общее количество требований.

Для измерения покрытия требований, необходимо проанализировать требования к продукту и разбить их на пункты. Опционально каждый пункт связывается с тест кейсами, проверяющими его. Совокупность этих связей - и является матрицей трассировки. Проследив связи, можно понять какие именно требования проверяет тестовый случай.

Тесты не связанные с требованиями не имеют смысла. Требования, не связанные с тестами - это "белые пятна", т.е. выполнив все созданные тест кейсы, нельзя дать ответ реализовано данное требование в продукте или нет.

Для оптимизации тестового покрытия при тестировании на основании требований, наилучшим способом будет использование стандартных техник тест дизайна.

## Покрытие кода (Code Coverage)

Расчет тестового покрытия относительно исполняемого кода программного обеспечения проводится по формуле:

Tcov = (Ltc/Lcode) \* 100%

где:  
**Tcov** – тестовое покрытие, **Lct** – количество строк кода, покрытыми тестами, **Lcode** – общее количесво строк кода.

В настоящее время существует инструментарий (например: Clover), позволяющий проанализировать в какие строки были вхождения во время проведения тестирования, благодаря чему можно значительно увеличить покрытие, добавив новые тесты для конкретных случаев, а также избавиться от дублирующих тестов. Проведение такого анализа кода и последующая оптимизация покрытия достаточно легко реализуется в рамках тестирования белого ящика (white-box testing) при модульном, интеграционном и системном тестировании; при тестировании же черного ящика (black-box testing) задача становится довольно дорогостоящей, так как требует много времени и ресурсов на установку, конфигурацию и анализ результатов работы, как со стороны тестировщиков, так и разработчиков.

## Тестовое покрытие на базе анализа потока управления

**Тестирование потоков управления** (Control Flow Testing) - это одна из техник тестирования белого ящика, основанная на определении путей выполнения кода программного модуля и создания выполняемых тест кейсов для покрытия этих путей.

Фундаментом для тестирования потоков управления является построение графов потоков управления (Control Flow Graph), основными блоками которых являются:

* блок процесса - одна точка входа и одна точка выхода
* точка альтернативы - одна точка входа, две и более точки выхода
* точка соединения - две и более точек входа, одна точка выхода

Для тестирования потоков управления определены разные **уровни тестового покрытия**:

|  |  |  |
| --- | --- | --- |
| **Уровень** | **Название** | **Краткое описание** |
| Уровень 0 | -- | “Тестируй все что протестируешь, пользователи протестируют остальное” На английском языке это звучит намного элегантнее: *“Test whatever you test, users will test the rest”* |
| Уровень 1 | Покрытие операторов | Каждый оператор должен быть выполнен как минимум один раз. |
| Уровень 2 | Покрытие альтернатив [**[2]**](http://www.protesting.ru/testing/testcoverage.html#[2]) / Покрытие ветвей | Каждый узел с ветвлением (альтернатива) выполнен как минимум один раз. |
| Уровень 3 | Покрытие условий | Каждое условие, имеющее TRUE и FALSE на выходе, выполнено как минимум один раз. |
| Уровень 4 | Покрытие условий альтернатив | Тестовые случаи создаются для каждого условия и альтернативы |
| Уровень 5 | Покрытие множественных условий | Достигается покрытие альтернатив, условий и условий альтернатив (Уровни 2, 3 и 4) |
| Уровень 6 | “Покрытие бесконечного числа путей” | Если, в случае зацикливания, количество путей становится бесконечным, допускается существенное их сокращение, ограничивая количество циклов выполнения, для уменьшения числа тестовых случаев. |
| Уровень 7 | Покрытие путей | Все пути должны быть проверены |