**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 «ТРПО»**

**на тему «Тестовое покрытие»**

ВЫПОЛНИЛА:

Студентка группы ИСП-О-17

Пигарева Е. А.

ПРОВЕРИЛА:

Прокуронова А. Ю.

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

п. Электроизолятор

1. г.

**Тестовое покрытие** - 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.***

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.  
**Самый лучший способ оценить, хорошо ли мы протестировали продукт** – проанализировать пропущенные дефекты. Те, с которыми столкнулись наши пользователи, внедренцы, бизнес. По ним можно многое оценить: что мы проверили недостаточно тщательно, каким областям продукта стоит уделить больше внимания, какой вообще процент пропусков и какова динамика его изменений. С этой метрикой (пожалуй, самой распространённой в тестировании) всё хорошо. Когда мы выпустили продукт, и узнали о пропущенных ошибках, может быть уже слишком поздно: на “хабре” появилась про нас гневная статья, конкуренты стремительно распространяют критику, клиенты потеряли к нам доверие, руководство недовольно.

Чтобы такого не происходило, мы обычно заранее, до релиза, стараемся оценивать качество тестирования: насколько хорошо и тщательно мы проверяем продукт? Каким областям не хватает внимания, где основные риски, какой прогресс? И чтобы ответить на все эти вопросы, мы оцениваем тестовое покрытие.

**Любые метрики оценки** – трата времени. В это время можно тестировать, заводить баги, готовить автотесты. **Какую такую магическую пользу мы получаем благодаря метрикам тестового покрытия, чтобы пожертвовать временем на тестирование?**

1. **Поиск своих слабых зон.**Естественно, это нам нужно? не чтобы просто погоревать, а чтобы знать, где требуются улучшения. Какие функциональные области не покрыты тестами? Что мы не проверили? Где наибольшие риски пропуска ошибок?
2. Редко по результатам оценки покрытия мы получаем 100%. Что улучшать? Куда идти? Какой сейчас процент? Как мы его повысим какой-либо задачей? Как быстро мы дойдём до 100?**Все эти вопросы приносят прозрачности и понятности нашему процессу**, а ответы на них даёт оценка покрытия.
3. **Фокус внимания.**Допустим, в нашем продукте около 50 различных функциональных зон. Выходит новая версия, и мы начинаем тестировать 1-ю из них, и находим там опечатки, и съехавшие на пару пикселей кнопки, и прочую мелочь… И вот время на тестирование завершено, и эта функциональность проверена детально… А остальные 50? Оценка покрытия позволяет нам приоритезировать задачи исходя из текущих реалий и сроков.

Прежде, чем внедрять любую метрику, важно определиться, как вы её будете использовать. Начните с ответа именно на этот вопрос – скорее всего, вы сразу поймёте, как её лучше всего считать. А я только поделюсь в этой статье некоторыми примерами и своим опытом, как это можно сделать. Не для того, чтобы слепо копировать решения – а для того, чтобы ваша фантазия опиралась на этот опыт, продумывая идеально подходящее именно вам решение.

## Покрытие требований (Requirements Coverage)

Расчет тестового покрытия относительно требований проводится по формуле:

**Tcov = (Lcov/Ltotal) \* 100%**

где:  
**Tcov** - тестовое покрытие  
**Lcov** - количество 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. Пример разработки тестовых случаев по имеющимся требованиям рассмотрен в разделе: "[**Практическое применение техник тест дизайна при разработке тест кейсов**](http://www.protesting.ru/testing/testdesign_practice.html)"

Покрытие кода (Code Coverage)

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

**Tcov = (Ltc/Lcode) \* 100%**

где:  
**Tcov** - тестовое покрытие  
**Ltc** - кол-ва строк кода, покрытых тестами  
**Lcode** - общее кол-во строк кода.

## 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. [**[1]**](http://www.protesting.ru/testing/testcoverage.html#[1])

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 |

Основываясь на данных этой таблицы, вы сможете спланировать необходимый уровень тестового покрытия, а также оценить уже имеющийся.