УДК 004.021

ТЕСТИРОВАНИЕ ДУБЛЕРОВ С ПОМОЩЬЮ ФЕЙКОВЫХ ОБЪЕКТОВ

Чумаков А. А.
ст. преп. каф. СПО Онищенко Т. В.
Одесский Национальный Политехнический Университет, УКРАИНА

АННОТАЦИЯ. В данной работе проведенанализ методики *Unit*-тестирования под названием «Тестирование дублерами». Применение данной методики является подходящим решением при тестировани в проектах с большим количеством исходного кода.

Введение. При тестированииипрограммного обеспечения с большой зависимостью есть два варианта покрытия его*unit*-тестами (наборы тестов, направленные на проверку единицы кода). Первый вариант – тестирование необходимого для разработчика класса и классов, от которых он зависит. Второй вариант – изоляция необходимого класса от зависимостей. В данной статье будет рассмотренвторой метод тестирования.

Цель работы. Анализ методики «Тестирование дублерами» [1], как один из способов тестирования в информационной системе с высокой зависимостью, не затрагивающий при этом другие классы помимо него.

Основная часть работы. Иногда покрытие тестами класса является трудной задачей из-за жесткой архитектуры проекта. Такая ситуация может возникнуть, если зависимое окружение не является доступным для разработчика, поскольку не возвращает результаты нужные для теста, или вследствие побочного эффекта в случае их выполнения.

Когда пишется тест, в котором нет возможности (или не хотим) применять реальное окружение объекта, можнозаменить окружение двойниками. Дублер не должен вести себя так же, как и реальное окружение, он должен только предоставлять необходимые *API* для нашего класса. В таком случае класс думает, что наши дублеры это реальное окружение.

«Тестирование дублерами» применяется в случаях:

- если окружение нашего тестируемого класса, является нерабочим;
- если есть непроверенные требования, и исходя из них следует проверить поведение интересуемого класса;
- если нужно, чтобы тестируемая компонента запускалась и собиралась намного быстрее, чем реальный проект.

Каждый из этих случаев можно решить с помощью тестового двойника. Безусловно, нужно осторожно использовать двойники в тестах, потому что тестируется нереальное программное обеспечение. Не следует ничего изменять воригинальном классе, поведение которого пытаемся проверить, иначе это приведет к некорректным испытаниям, и тесты нельзя будет применить к реальному проекту.

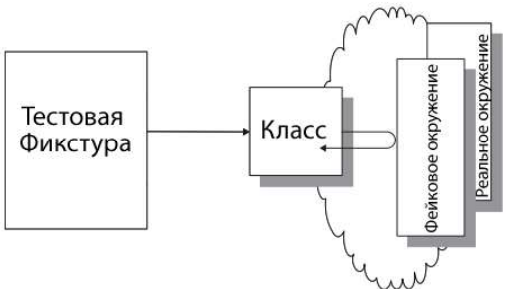

Рис. 1–Схема работы «Тестирование Дублерами»

Существует несколько вариантов реализации «Тестирования дублеров», однако в настоящей статье рассматриваетсяприменение данной методики, используя «Фейковые объекты». Наиболее распространённая причина по использованию фейковых объектов связана с тем, что окружение тестового объекта в данный момент недоступно, работает неисправно, либо будет замедлять тестирование класса.

Фейковый объект – это гораздо упрощенная реализация функционала, предоставляемым реальным окружением, оснащенная лишь нужными для тестирования элементами. Таким образом, тестовое окружение предоставляет идентичные услуги для тестируемого класса, а последний не подозревает, что он общается с дублером.

Данная методика тестирования применима:

- для имитации работы базы данных (замену базы данных набором Хэш-таблиц в памяти, являющимися упрощенными дублерами, которые были «сохранены» ранее в тесте);
- для имитации работы веб-сервиса (при тестировании программного обеспечения, которое зависит от других компонентов, которые доступны в качестве веб-сервисов, можно создать небольшую жестко кодированную или управляемую данными реализацию, которая может использоваться вместо реального веб-сервиса, чтобы сделать тесты более надежными и избежать необходимости создавать тестовый экземпляр реального веб-сервиса в среде разработки);
- при имитации сервиса (при тестировании пользовательских интерфейсов можно избежать чувствительности к сервисному *API*, заменив компонент, который реализует уровень сервиса приложения с помощью фейковых объектов, возвращающий запоминаемые или управляемые результаты. Такой подход позволяет сосредоточиться на тестировании пользовательского интерфейса, не беспокоясь о том, что данные, которые возвращаются, меняются с течением времени.)

Выводы. Тестирование дублерами – методика тестирования, которая во многом является вынужденной мерой при покрытии определенного компонента *unit*-тестами. Вынужденной она является либо из-за масштаба самойинформационной системы, либо из-за плохой архитектуры в самом проекте.

Для применения данной методологии уместно использовать тестовые фреймворки. Один из ярких примеров данных тестовых фреймворков является *GoogleTestFramework*[2]. В данномфреймворке разработчики могут найти нужные элементы для отслеживания вызовов функций из интересуемого класса и последовательность данных вызовов. Также можно задавать активность для *API* дублера информационной системы.

Необходимо не забывать проверять соответствие тестового окружения с его оригиналом. Со временем при модификации программного обеспеченияиспользуемое тестовое окружение может оказатьсяневалидным, а значит тесты для реального проекта станут устаревшими и неправильными.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. «xUnitsTestPatterns»[Электронный ресурс] – Режим доступа: URL: https://amazon.com/
2. «GoogleTest»[Эллектроный ресурс] – Режим доступа: URL:https://github.com/google/googletest
3. «Programmingstuff»[Электронный ресурс] – Режим доступа: URL: http://sergeyteplyakov.blogspot.com