только в наших собственных руках. И мы сможем жить так долго, как мы этого хотим.

Библиографический список

1. Ромул, Марк. Сингулярность уже близко [Электронный ресурс] //Сайт проекта novaduus. Режим доступа: http://novadeus.com/wp-content/uploads/Singularity.pdf (дата обращения 17.06.14)

А.С. Плесовских, (Екатеринбургский экономико-технологический колледж)
студент группы 339-ИС
Руководитель: преподаватель спец. дисциплин
С.В. Ермолаева

РАЗРАБОТКА ЧЕРЕЗ ТЕСТИРОВАНИЕ В ОБРАЗОВАНИИ

В современной разработке программного обеспечения применяется множество методологий. Одна из них -- это экстремальное программирование. В первую очередь, экстремальное программирование направлено на повышение качества продукта и упрощение изменений оного. Важной техникой этой методологии является разработка через тестирование (англ. test-driven development, TDD). Благодаря этой технике можно существенно повысить качество обратной связи, что особенно полезно на раннем этапе обучения программированию.

Прежде чем перейти непосредственно к делу хотелось бы упомянуть об актуальности проблемы тестирования. Все уважающие себя компании тратят огромные деньги на тестирование своих продуктов. Связанно это с тем, что неустойки, оговоренный в SLA (Соглашение об уровне услуг) могут быть очень значительными, а репутацию компании восстановить крайне сложно.

Принятие в штат разработчика, который привык писать тесты для компании может сулить естественное снижение затрат на ручное и полуавтоматическое тестирование, следовательно такой сотрудник будет очень полезен, что не может не отразиться на уровне его оклада.

Для начала хотелось бы подробнее рассказать про суть данной техники. Для этого введем пару несложных понятий:

Модульное тестирование (англ. unit testing) — это метод тестирования программного обеспечения, при котором отдельно тестируются разные единицы исходного когда (модули, интерфейсы и т.д.). Этот метод тестирования позволяет легче вносить правки в код и быстрее локализовать возможные ошибки.

Мок объект (от англ. mock object) — это объект в объектно-ориентированном программировании, реализующий поведение некой сущности в рамках моделируемого окружения. Моки часто применяются для симуляции поведения медленных и зависящих от внешних данных интерфейсов.

TDD состоит из нескольких основных этапов, которые раз за разом повторяются циклично. В базовом представлении этих этапов три: red, green и refactor.

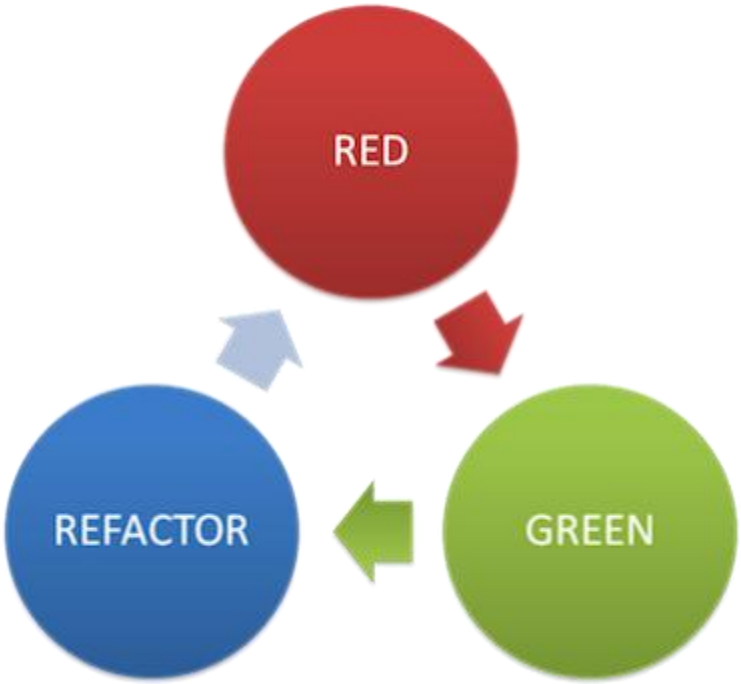

Рисунок 16 – Модель TDD

На первом этапе (red, красный) происходит написание unit-теста на пока еще несуществующую функцию программы. Этот тест должен быть простым и лаконичным, а реализация самой функции в идеале не должна занимать много времени. После этого все написанные ранее тесты запускаются, и последний должен быть “красным”, что значит, что он провален.

Запускать тест на еще нереализованную часть программы нужно для уверенности в том, что этот тест действительно на что-то проверяет. Часто для написания абстрагированных тестов использую вышеупомянутые Моки.

На втором этапе (green, зеленый) происходит написание минимального количества кода, которое необходимо для прохождения последнего теста. При этом все предыдущие тесты также должны быть успешно пройдены.

На третьем этапе (refactor, рефакторинт) происходит переработка кода под принятые стандарты и его оптимизация.

При обучении студентов программированию написанию тестов отводится очень мало времени. А чаще всего не отводится совсем. Это большая проблема, потому что привыкание к хорошим практикам программирования порой занимает очень много времени.

При постановке задачи, так или иначе оглашаются какие-нибудь требования к конечному решению. Если эти требования представляют из себя тест или набор тестов, то это может упростить жизнь как студенту, так и преподавателю. Для преподавателя выигрыш будет в том, что у него есть уже готовое, автоматизированное средство проверки работы. Для студента же выигрыш будет в том, что у него будет отличное средство самоконтроля, и он постепенно будет привыкать к хорошим техникам разработки.

Стандартный для TDD цикл red-green-refactor может успешно применяться и в решении домашних заданий, когда после каждой лекции учитель предоставляет ученикам новые требования и тесты. Такой подход может повысить качество кода, так как студентам придется писать более “чистый” код, чтобы самому потом можно было в нём разобраться.

Библиографический список

1. Kent Beck, Test-Driven Development: By Example [Текст]. – Addison-Wesley Professional; 1 edition, 2002 – 240 с.

2. Lee Copeland. Extreme Programming [Электронный ресурс]. // Сайт computerworld. – Режим доступа: http://www.computerworld.com/s/article/66192/Extreme_Programming?taxonomyId=063 (дата обращения 17.06.14)

Ю.П. Забанных (Российский государственный профессионально-педагогический университет)
студент группы ЗКТ-517

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ В СВАРОЧНОМ ПРОИЗВОДСТВЕ

В современном мире, на заре XXI века, ИК технологии получили широкое применение во всех сферах жизнедеятельности человека. Особое место они занимают в росте производстве, т. к. это основной показатель высокого жизненного уровня населения. Продукция сварочного производства на сегодня не обходится без нововведений на базе информационных и коммуникационных технологий (ИКТ).

Развитие компьютерного инженеринга дает возможность сэкономить время на разработку и продвижение изделия на протяжении всего процесса изготовления. С помощью компьютерных технологий моделирование процессов при сварке дает возможность быстро оптимизировать технологическое решение.

Разработано много программных продуктов, с помощью которых достигается высокая точность расчетов и обрабатывается большой объем информации. Создается необходимая технологическая документация, рассчитывается прочность, долговечность, жесткость сварных конструкций. Кроме этого дается информация о необходимом применении оборудования и материалов для изготовления.

Одной из таких программ является: Компас-Вертикаль – система автоматизированного проектирования технологических процессов (САПР ТП). Результатом такой программы является электронное описание изделия, содержащего полную информацию, необходимую для поддержки всех этапов его жизненного цикла.