Компьютерные инструменты в образовании, 2019
№ 1: 88–100
УДК: 004.4:378:519.6
http://ipo.spb.ru/journal
doi:10.32603/2071-2340-2019-1-88-100

КОМПЛЕКСНЫЙ ПОДХОД К ОБУЧЕНИЮ ТЕСТИРОВАНИЮ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ В ОБРАЗОВАТЕЛЬНОМ ПРОЦЕССЕ ПЕТРГУ

Димитров В. М.[1], преподаватель, dimitrov@cs.petrsu.ru
Кулаков К. А.[1], кандидат физ.-мат. наук, доцент, kulakov@cs.petrsu.ru

[1]Петрозаводский государственный университет,
пр. Ленина, 33, 185910, Республика Карелия, Петрозаводск, Россия

Аннотация

В статье содержатся сведения по организации подготовки специалистов в области информационных технологий в Петрозаводском государственном университете и используемым для этого инструментам по одному из основополагающих этапов разработки программного обеспечения — тестированию. Описаны три этапа подготовки: знакомство с технологий тестирования (1 курс, бакалавриат), тестирование как этап разработки командного проекта (3 курс, бакаливриат) и организация процесса тестирования (2 курс, магистратура). Для каждого этапа рассмотрен набор инструментов, применяемых в учебном процессе. Обучение на этапах выстроено от простого использования инструментов до управления процессом тестирования в команде разработчиков.

Ключевые слова: *информационные технологии, технология разработки программного обеспечения, тестирование, обеспечение качества программного обеспечения, учебный процесс.*

Цитирование: Димитров В. М., Кулаков К. А. Комплексный подход к обучению тестированию программного обеспечения в образовательном процессе ПетрГУ // Компьютерные инструменты в образовании. 2019. № 1. С. 88–100. doi: 10.32603/2071-2340-2019-1-88-100

1. ВВЕДЕНИЕ

Информационно-телекоммуникационные системы являются одним из приоритетных направлений развития науки, технологии и техники в Российской Федерации [1]. Для обеспечения рынка труда квалифицированными кадрами в Институте математики и информационных технологий Петрозаводского государственного университета (ПетрГУ) выполняется подготовка специалистов в области информационных технологий (ИТ) по следующим направлениям:

- 01.03.02 «Прикладная математика и информатика» (академический бакалавриат);
- 09.03.04 «Программная инженерия» (академический бакалавриат);
- 09.03.02 «Информационные системы и технологии» (академический бакалавриат);
- 01.04.02 «Прикладная математика и информатика» (магистратура);

- 09.04.02 «Информационные системы и технологии» (магистратура).

Институтом математики и информационных технологий ведется постоянный диалог с региональными работодателями для оценки качества образования выпускников. Для будущих соискателей работодателями отмечаются такие навыки, как: владение современными технологиями и инструментарием, командная работа и получение качественного результата.

Одной из проблем академического образования является отсутствие мотивации к применению полученных знаний и навыков [2]. Как правило, мотивация приходит с опытом, когда необходимость применения полученных знаний и навыков подтверждается практическими задачами. Таким образом, актуальной является задача мотивации обучающихся к изучению современных технологий и инструментов и к получению качественного результата своей деятельности.

В рамках статьи мы рассматриваем такой важный процесс разработки программного обеспечения как тестирование [3–5]. В академическом образовании тестирование рассматривается в обобщенном теоретическом аспекте с демонстрационными практическими заданиями [6]. В результате, у бакалавров не формируется навык подготовки результатов своей деятельности к различным проверкам и навык самостоятельных проверок полученных результатов.

В проведенном нами анализе существующих работ [7–9] рассмотрено преподавание дисциплины тестирование ПО с точки зрения одной дисциплины, а не комплексного подхода в рамках полного цикла обучения. Например, в [8] разбираются конкретные примеры кода для обучения тестированию в рамках вводного курса в ведение в тестирование. Все теоретические и практические аспекты тестирования, рассмотренные в статье [7], предлагаются обучающимся в рамках дисциплины «Технология разработки программного обеспечения» и «Верификация программного обеспечения».

В статье представлен практико-ориентированный подход к изучению технологий и инструментов тестирования в рамках проектной деятельности по разработке программного обеспечения. Изучение разделено на три этапа: получение базовых навыков (1 курс бакалавриата), отработка командной работы (3 курс бакалавриата) и понимание роли тестирования в процессе разработки программного обеспечения (2 курс магистратуры).

Статья имеет следующую структуру. В п. 2 отражены используемые практики и инструменты для изучения основ тестирования. Участие тестирования в процессе командной разработки программного обеспечения представлено в п. 3. В п. 4 описана организация процесса тестирования программного обеспечения. В п. 5 представлен анализ применения комплексного подхода, выполненный при помощи анкетирования различных групп обучающихся.

2. ЗНАКОМСТВО С ТЕХНОЛОГИЕЙ ТЕСТИРОВАНИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Изучение языков программирования является ключевым элементом становления специалиста в области ИТ. Однако изучение, как правило, представляет собой формальное описание языковых конструкций и способов их применения. Таким образом, возникает необходимость воспитания культуры оформления и проверки результатов с младших курсов.

В Институте математики и информационных технологий ПетрГУ в качестве первого и базового языка выступает язык C. Конечным результатом соответствующего курса «Основы информатики и программирования» выступают модули для многомодульного

приложения. Полученный результат зачастую обладает низким качеством: отсутствует оформление кода в соответствии с выбранным стандартом, комментарии к коду, присутствуют различные ошибки и дефекты программирования.

В рамках курса «Введение в тестирование» обучающиеся получают базовые навыки в области тестирования и обеспечения качества программного обеспечения. Курс состоит из двух этапов:

1) тестирование модуля (базовое задание);
2) тестирование многомодульного приложения (основное задание).

В рамках базового задания обучающиеся осваивают технологии и инструменты тестирования и применяют на практике полученные знания для тестирования модуля из курса «Основы информатики и программирования».

Первым инструментом выступает среда разработки QtSDK [10]. Среда обладает различным набором полезных функций, таких как автодополнение и автоформатирование, а также имеет встроенный модуль для запуска тестов и представления результатов тестирования. В рамках изучения и освоения среды QtSDK обучающимся необходимо, используя шаблон проекта [11], скомпилировать и запустить простое приложение, содержащее разработанный модуль. В ходе выполнения задания обучающиеся закрепляют полученные ранее навыки по организации многомодульных приложений, необходимости в заголовочных файлах и выделению компонент приложения.

Вторым инструментом выступает среда тестирования Google Test [12]. Среда позволяет тестировать приложения на языках C/C++. В рамках освоения среды Google Test обучающимся необходимо написать несколько тестов для функций своего модуля. В ходе выполнения задания вырабатывается практический опыт преодоления различных технических сложностей. Например, тестируемая функция не возвращает результат работы, а выводит его на экран. Таким образом, обучающимся наглядно демонстрируются возможные проблемы некачественной разработки программного обеспечения.

Демонстрация качества тестирования выбранного модуля осуществляется с помощью веб сервиса совместной разработки Github [13], распределенного веб сервиса сборки и тестирования программного обеспечения Travis CI [14], веб-сервиса оценки покрытия кода Coveralls [15] и веб-сервиса оценки качества написания кода SonarCloud [16]. Использование веб-сервисов продиктовано независимостью оценки результатов с помощью автоматических средств, простотой настройки и наглядностью. Кроме того, большая часть региональных работодателей используют аналогичные или подобные системы в своих процессах, а у потенциальных соискателей требуют портфолио в виде перечня вкладов в проекты с ссылками. В рамках освоения инструментов обучающимся необходимо выполнить публикацию кода, запуск автоматических тестов и получение отчета о покрытии кода тестами. Как правило, в ходе выполнения задачи выясняется, что покрытие не достигает 100 % и требуется написание одного или нескольких тестов для обеспечения полного покрытия кода.

По результатам выполнения базового задания обучающиеся получают рабочий проект с подключенным модулем, набор тестов и результаты покрытия кода тестами. Полученные навыки использования инструментов разработки и тестирования закрепляются в основном задании.

В ходе основного задания обучающиеся разбиваются на пары и реализуют совместный проект по разработке программного средства. При этом отслеживается использование коллективной деятельности: исходный код приложения и тесты к нему пишутся

разными людьми. Таким образом, будущие специалисты на практике приучаются к качественному написанию кода.

По результатам выполнения основного задания обучающиеся демонстрируют созданные программные средства и тесты к нему, а также оценку покрытия кода тестами и оценку качества написания кода. Таким образом, в ходе прохождения курса «Введение в тестирование» формируется понимание важности получения качественного результата, осваиваются на практике популярные инструменты и средства разработки. Результаты выполнения проекта могут быть добавлены к портфолио для демонстрации будущим работодателям.

3. УЧАСТИЕ ТЕСТИРОВАНИЯ В ПРОЦЕССЕ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Для бакалавров третьего года по специальностям «Прикладная информатика и математика», «Информационные системы и технологии» и «Программная инженерия» в ПетрГУ читается курс «Технологии производства программного обеспечения». Курс предполагает деление на небольшие проектные группы (от трех до шести человек) и выполнение группового проекта по разработке ПО. Курс является годовым с лекциями в первом семестре и отдельными проектами для каждого семестра.

Куратор и инструкторы курса отслеживают общее соблюдение технологии производства ПО, помогают распределить отведенное время, скорректировать направления работы в случае отклонений от плана. Роли и задачи каждого участника, общение с заказчиком, выбор технологий, проектные решения команда реализует самостоятельно.

Также стоит отметить, что если в первом семестре проектная группа самостоятельно выбирает тему проекта, то во втором семестре к составлению списка проектов привлекаются реальные заказчики (как региональные работодатели, так и сотрудники ПетрГУ), которым важен полученный результат в конце проекта. Проектная группа выбирает проект из списка доступных и в течение семестра работает с заказчиком. При оценке проекта учитывается в том числе и мнение заказчика. Такое построение работы позволяет мотивировать обучающихся на получение результата и зарекомендовать себя в профессиональной среде своего региона.

Для успешного завершения проекта команда должна выполнить несколько этапов: сбор и анализ требований к разрабатываемому программному продукту, проектирование, написание плана тестирования, кодирование, выполнение тестирования и аттестация проекта.

Рассмотрим особенности обучения тестированию, присущие данному курсу.

Командная работа предполагает, что разные люди, приступая к этапу тестирования, обращаются к одному и тому же тексту или коду, поэтому текст теста должен быть описан максимально понятно и подробно или код функции или метода класса должен быть документирован и выдержан в принятом в команде стиле. Это позволяет развивать в обучающихся ответственность за свой текст или код.

Особенностью данного курса для инженерных специальностей является то, что на кодирование отводится лишь малая часть времени (обычно занимает две-три недели из всего семестра). Остальная часть времени связана с разработкой и написанием документации по проекту.

В связи с этим было решено предоставить для обучающихся инструмент для совместной работы над документами. Таким инструментом была выбрана технология

wiki [17, 18] и реализация этой технологии MediaWiki [19, 20], которая была развернута в вычислительной системе кафедры Информатики и математического обеспечения ПетрГУ с персонифицированным доступом к изменению содержания страниц. Обучающиеся и преподаватели получили следующие преимущества:

— простой язык для оформления текста (обычно легко осваивается самостоятельно, иногда требуется только демонстрация принципов использования) и добавления другого медиа контента (например изображения или видео);
— доступ к документации из любого места с компьютером и сетью интернет;
— возможность совместно и параллельно наполнять содержимое документации без необходимости находиться в одном месте.
— возможность отслеживать динамику содержимого страниц и персональную ответственность за изменения;
— возможность восстановить ранее измененное содержимое.

Одним из этапов создания ПО в данном курсе является написание и подготовка тестов. Стоит отметить, что подготовка тестов идет до написания кода, и это является важнейшим технологическим требованием к проектным группам. Многие команды допускают ошибки на данном этапе, и задача курса состоит в том, чтобы продемонстрировать преимущества написания тестов до написания кода. Таким образом, тестирование в курсе формирует абстрактное мышление, навыки предугадывания и предположения возможных ситуаций, возникающих в ходе использования результатов проекта.

Подготовке тестов предшествует раздел в документе спецификации требований, в котором перечисляются критерии аттестации проекта, описывающие, как будет проверяться проект на соответствие требованиям.

Подготовка к тестированию описывается в отдельном документе, который называется планом тестирования. Он должен включать в себя следующие разделы:

— описание методов тестирования;
— варианты тестов;
— трассируемость требований в тестах.

Описание методов тестирования подразумевает общее описание групп тестов и контекст применения тестов. Например, если проект связан с разработкой ПО для специфичного аппаратного обеспечения или ПО должно работать в специфичных условиях (температурных или атмосферных), то такое аппаратное обеспечение или такие условия должны быть описаны. Также контекст может быть специфичен для теста. Например, проверка корректности авторизации пользователя в системе подразумевает наличие учетной записи пользователя с требуемыми входными параметрами.

Также в данном разделе описываются инструменты для проведения тестирования. Группам предоставляется право самостоятельно выбрать набор инструментов для проведения тестирования.

В рамках курса требуется применить как минимум три группы вариантов тестов:

- Блочные тесты направлены на тестирование одного изолированного конкретного блока кода (обычно это метод класса или функция модуля). Описание должно содержать набор конкретных входных параметров, результат, получаемый после выполнения кода, и особые условия на выполнение теста. В описание особых условий входит необходимость описать исключения или прерывания выполнения кода из-за специфичных входных параметров. Пример описания блочных тестов одного из учебного проекта в системе wiki приведен на рис. 1.

Класс SnmpContext

ID	Метод	Тип	Описание	Входные данные	Ожидаемый результат
CTX_A1	getCommunityStrings()	Черного ящика, общий	Получение vlan-зависимых строк сообществ на устройстве. Применяется к устройству с ENTITY-MIB (172.20.255.110)	IP-Адрес устройства, параметры доступа к нему, entity-mib	Вернется список строк сообществ на устройстве, соответствующий информации в ENTITY-MIB
CTX_A2	getCommunityStrings()	Черного ящика, общий	Получение vlan-зависимых строк сообществ на устройстве. Применяется к устройству без ENTITY-MIB (172.20.254.246)	IP-Адрес устройства, параметры доступа к нему, список vlan на устройстве	Вернется список строк сообществ на устройстве, составленный из строки сообщества из параметров доступа и приклееных к ней номеров vlan через символ "@"
CTX_A3	getSnmpContextNames()	Черного ящика, общий	Получение vlan-зависимых имен snmp-контекстов устройстве. Применяется к устройству с ENTITY-MIB (172.20.255.110)	IP-Адрес устройства, параметры доступа к нему, entity-mib	Вернется список имен контекстов на устройстве, соответствующий информации в ENTITY-MIB
CTX_A4	getSnmpContextNames()	Черного ящика, общий	Получение vlan-зависимых имен snmp-контекстов устройстве. Применяется к устройству без ENTITY-MIB (172.20.254.246)	IP-Адрес устройства, параметры доступа к нему, entity-mib	Вернется список имен контекстов на устройстве, состоящий из имен ""

Рис. 1. Пример описания блочных тестов учебного проекта (снимок экрана из системы wiki)

- Интеграционные тесты должны протестировать взаимодействие между модулями или классами. Перед началом описания интеграционных тестов необходимо составить схему интеграции — последовательность, в которой будут тестироваться и добавляться новые модули или группы модулей до тех пор, пока не будет собран весь код в единый контекст. На рис. 2 приведен пример интеграционной схемы одного из учебных проектов.

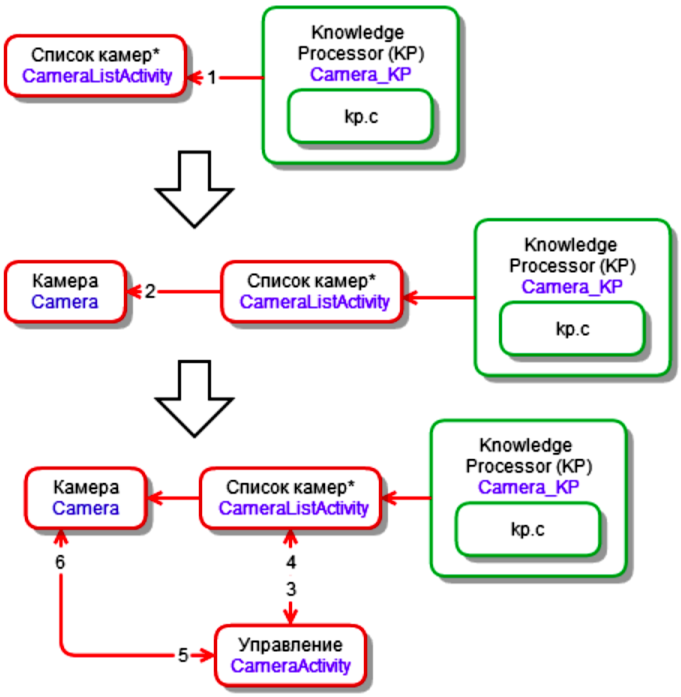

Рис. 2. Пример схемы интеграции учебного проекта (снимок экрана из системы wiki)

- Аттестационные тесты направлены на комплексное тестирование программного продукта в соответствии с критериями аттестации, которые должны быть описаны в документе спецификации требований.

Для описания трассируемости требований тестов составляется таблица, в которой по столбцам перечислены тесты, а по строкам требования. На пересечении указывается, как данный тест проверяет (тестирует) данное требование. Эта таблица служит для того, чтобы проверить, полностью ли требования покрыты тестами. Для каждой группы тестов создается собственная таблица трассируемости.

Следующим важным этапом с точки зрения обеспечения качества ПО является проведение тестирования. В ходе тестирования требуется оформление документа, который называется журналом тестирования (или документ о выполнении тестирования). Для каждого проведенного теста оформляется запись, которая содержит следующую информацию: дата проведения теста, описание теста (ссылка на тест из документа плана тестирования), номер попытки, тестировщик, входные данные, выходные данные и результат. Пример журнала тестирования приведен на рис. 3.

ID	Дата	Тестировал	Количество попыток	Входные данные	Результат
VDB_A1	12.05.2014	Андреев	3	Устройство 192.168.112.29, SNMPv3, логин: andreev, пароль: засекречен, dot1qTbale: {[dot1qVlanCurrentEgressPorts: FF FF dot1qVlanIndex: 65], [dot1qVlanCurrentEgressPorts: 00 F0 dot1qVlanIndex: 102], [dot1qVlanCurrentEgressPorts: 00 F0 dot1qVlanIndex: 1]}	NullPointerException, --- тест провален
VDB_A1	12.05.2014	Андреев	3	Устройство 192.168.112.29, SNMPv3, логин: andreev, пароль: засекречен, dot1qTbale: {[dot1qVlanCurrentEgressPorts: FF FF dot1qVlanIndex: 65], [dot1qVlanCurrentEgressPorts: 00 F0 dot1qVlanIndex: 102], [dot1qVlanCurrentEgressPorts: FF FF dot1qVlanIndex: 1]}	таблица: 65-1, 65-2, 65-3 (и т.д. - порты от 1 до 32), 102 - порты от 9 до 16, 1 - порты от 1 до 32 --- все правильно
VDB_A2	12.05.2014	Андреев	3	Устройство 172.20.255.110, SNMPv2c, сообщество: forIMO	NullPointerException, --- как и ожидалось
VDB_A2	12.05.2014	Андреев	3	Устройство 172.20.254.236, SNMPv2c, сообщество: forIMO	NullPointerException, --- как и ожидалось

Рис. 3. Пример журнала тестирования (снимок экрана из системы wiki)

В случае выявления дефекта ПО тестировщик должен составить подробное описание возникновения данного дефекта: среда запуска ПО (в том числе перечисление используемых аппаратных и программных средств), порядок выполнения шагов, приводящих к появлению дефекта. Чем подробнее будет представлена данная информация, тем быстрее и качественнее разработчики смогут устранить дефект. Это позволяет воспитывать в обучающихся чувство ответственности за выполняемую работу и мотивировать их на применение знаний и навыков для достижения результата.

После завершения всех этапов разработки подсчитываются метрики проекта, в том числе тестирования: количество запланированных тестов, количество выполненных тестов, включая попытки для выполнения одного теста, количество обнаруженных дефектов. Метрики позволяют количественно оценить проделанную работу.

На данном этапе у обучающихся формируется способность приобретать и использовать организационно-управленческие навыки в профессиональной и социальной деятельности, составлять и контролировать план выполняемой работы, планировать необ-

ходимые для выполнения работы ресурсы, оценивать результаты собственной работы, критически переосмысливать накопленный опыт, изменять при необходимости вид и характер своей профессиональной деятельности.

4. ОРГАНИЗАЦИЯ ПРОЦЕССА ТЕСТИРОВАНИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Курс «Верификация программного обеспечения» суммирует полученные теоретические и практические навыки обучающихся. В нем рассматриваются методические вопросы организации тестирования: подходы к анализу предметной области, методы построения тестов, методы анализа исходного кода программных средств.

Ключевой особенностью курса является упор на организационные моменты. Обучающиеся знакомятся с такими ключевыми моделями разработки программного обеспечения, как «разработка через тестирование» (“Test Driven Development”, TDD) и «разработка через поведение» (“Behavior Driven Development”, BDD). Использование этих моделей позволяет представить тестирование как инструмент контроля выполнения требований заказчика.

Другим ключевым моментом организации является планирование тестирования. Если в курсе «Введение в тестирование» достаточно составить перечень тестов, а в курсе «Технология производства ПО» требуется обеспечить покрытие функциональных требований тестами, то в курсе «Верификация ПО» рассматриваются различные качественные методики оценки полученных наборов тестов.

Ключевым моментом курса выступает документирование. Обучающиеся получают практические навыки в области оформления плана тестирования, журнала выполнения тестирования, протоколов обнаруженных ошибок. При этом используются общепринятые стандарты документирования: ЕСПД [21, 22] и IEEE [23, 24].

Обучающимся рекомендуется придерживаться следующей структуры документации:

- Объект тестирования. Описание объекта тестирования, рамки тестирования, перечень функциональностей объекта тестирования. Для каждой функциональности указать ее участие в аттестационном тестировании.
- Стратегия тестирования. Описание структуры объекта тестирования и связей внутри объекта тестирования (архитектура).
- Детальный план тестов. Перечень блочных, интеграционных, аттестационных и специальных тестов.
- Журнал тестирования. Дата, тестировщик, объект тестирования, перечень выполненных тестов с указанием количества запусков, перечень найденных ошибок.
- Журнал найденных ошибок. Номер отчета об ошибке, дата составления отчета, номер теста, ожидаемый результат, фактический результат.
- Результаты. Оценка качества исследуемого объекта, оценка результатов тестирования.

Таким образом, в конце курса обучающийся должен предоставить отчет. Из опыта прошлых лет, объем отчета составляет в среднем 50–60 страниц.

5. АНАЛИЗ КОМПЛЕКСНОГО ПОДХОДА К ОБУЧЕНИЮ ТЕСТИРОВАНИЮ

Для оценки комплексного подхода к обучению тестированию было выполнено анкетирование следующих групп обучающихся:

- Группа № 1: 1 курс, направление «Программная инженерия» (обучающиеся, прошедшие курс «Введение в тестирование»).
- Группа № 2: 1 курс, направления «Прикладная математики и информатика» и «Информационные системы и технологии» (контрольная группа, курс «Введение в тестирование» не преподавался).
- Группа № 3: 3 курс, направления «Прикладная математики и информатика» и «Информационные системы и технологии» (обучающиеся, прошедшие курс «Технология производства ПО», но курс «Введение в тестирование» не преподавался).

Содержательно анкета представляла из себя два задания: задание на написание кода и задание на выполнение тестирования функции. Формулировка заданий была сделана расплывчатой для выявления компетенций обучающихся:

- Задание № 1. Напишите код для решения следующей задачи: Необходимо вычислить временной интервал между двумя датами. Даты задаются в формате “YYYY-MM-DD”. Если для решения недостаточно описания условия задачи, то укажите ваши предположения для предлагаемого решения. Если не хватает места, продолжите на обороте анкеты.
- Задание № 2. Напишите необходимые проверки (тест-кейсы) для следующей функции: function int getAges(sDate) — возвращает количество полных лет от заданной даты. Параметр sDate — заданная дата в строковом формате “YYYY-MM-DD”.

Для оценки задания № 1 применялся следующий набор критериев:

1. Вопрос, что возвращать (разницу в днях или дату в формате “YYYY-MM-DD” или что-то еще) и самостоятельный ответ.
2. Проверка даты на формат.
3. Проверка, какая из дат больше.
4. Верный алгоритм вычисления (переходы через года, месяцы, учет високосных лет и др.).

Для оценки задания № 2 применялся следующий набор критериев:

1. Есть определение контекста в виде текущей даты.
2. Тест sDate на формат.
3. Тест sDate больше, чем текущая дата.
4. Тест, возвращающий 0, с переходом через месяц (например текущая “2019-03-01”, а тестовая “2019-01-01”).
5. Тест, возвращающий 0, с переходом через год (например текущая “2019-03-01”, а тестовая “2018-12-01”).
6. Тест на ровное количество лет ± один день (например текущая “2019-03-01”, а тестовая “2018-03-02”).
7. Тест на ровное количество лет (например текущая “2019-03-01”, а тестовая “2018-03-01”).
8. Тест на ту же дату, что и текущая.
9. Тест на дату с годом, меньшим 1000, 100 и 10 (например, “0103-01-01”).

За каждый критерий обучающий мог получить от 0 до 1 балла. Результаты проведения анкетирования представлены в таблице 1 (общее распределение по критериям). В таблице 1 цифра на пересечении критерия задания и курса/специальности означает среднее значение по всем обучающимся по данному критерию, в строке “Итого” — среднее значение по всем критериям.

По результатам анкетирования можно сделать следующие выводы.

1. Начиная с 1 курса, обучающиеся имеют навыки написания алгоритмов и их реализации на языке программирования. Однако любое отклонение от ожидаемой модели поведения программы слабо контролируется или не контролируется вообще (критерии 1–2 задания 1). Третий курс ожидаемо лучше справляется с первым заданием благодаря постоянной практике.
2. В области оценки качества прикладных результатов программирования первый курс не имеет соответствующих навыков (задание 2, направления ПМИ и ИСИТ). Проведение дисциплины «Введение в тестирование» значительно улучшает навыки верификации программного обеспечения (задание 2, направление ПИ).
3. Разделы дисциплины «Технология производства ПО» по тестированию и обеспечению качества программных продуктов также улучшает навыки обучающихся. Однако отсутствие сильной практической составляющей дисциплины, сфокусированной на обеспечении качества, не позволяет третьему курсу получить высокие оценки по второму заданию.

Таблица 1. Распределение результатов по критериям, курсам (1, 3) и специальностям «Программная инженерия» (ПИ), «Прикладная математики и информатика» (ПМИ) и «Информационные системы и технологии» (ИСИТ).

Задание № 1	**1, ПМИ**	**1, ИСИТ**	**1, ПИ**	**3, ПМИ**	**3, ИСИТ**
1	0.000	0.059	0.071	0.107	0.278
2	0.000	0.059	0.071	0.000	0.000
3	0.250	0.471	0.357	0.643	0.370
4	0.208	0.691	0.679	0.871	0.963
Итого	**0.092**	**0.268**	**0.250**	**0.324**	**0.322**
Задание № 2	**1, ПМИ**	**1, ИСИТ**	**1, ПИ**	**3, ПМИ**	**3, ИСИТ**
1	0.083	0.118	0.071	0.071	0.074
2	0.000	0.059	0.143	0.643	0.296
3	0.000	0.000	0.429	0.143	0.222
4	0.000	0.000	0.357	0.143	0.111
5	0.000	0.000	0.357	0.143	0.074
6	0.000	0.000	0.071	0.071	0.037
7	0.000	0.000	0.357	0.071	0.037
8	0.000	0.000	0.357	0.000	0.037
9	0.000	0.059	0.071	0.214	0.074
Итого	**0.008**	**0.024**	**0.229**	**0.157**	**0.100**

6. ЗАКЛЮЧЕНИЕ

Тестирование является одним из ключевых аспектов разработки качественного программного обеспечения и подготовки специалистов в области ИТ. Применяемый к обучению тестированию ПО в ПетрГУ подход позволяет добиться следующих качеств у обучающихся: понимание необходимости создания качественного программного обеспечения, знание и навыки владения инструментами тестирования, знание средств и методов обеспечения качества программного обеспечения, умение работать в команде, ответственность за выполняемую работу.

Три этапа обучения тестированию «от малого к большему» (овладение инструментами тестирования, тестирование как этап работы в команде, организация процесса те-

стирования) позволяют получить и закрепить навыки создания качественного кода и написания тестов.

Данные выводы подтверждаются проведенным анкетированием: обучающиеся по данному подходу значительно лучше справились с заданием на обеспечение качества программного обеспечения.

Список литературы

1. Об утверждении приоритетных направлений развития науки, технологий и техники в Российской Федерации и перечня критических технологий Российской Федерации: указ Президента Российской Федерации от 07.07.2011 г. № 899 [Электронный ресурс] // Режим доступа: http://pravo.gov.ru/proxy/ips/?docbody=&nd=102149065&rdk=&firstDoc=1&lastDoc=1 (дата обращения: 21.03.2019).
2. *Оселедец М. В., Новикова Т. А.* К вопросу о мотивациях студентов к обучению // Инновационная экономика и общество. 2014. № 2 (4). С. 67–73.
3. *Соммервилл И.* Инженерия программного обеспечения, 6-е издание: Пер. с англ. М.: Вильямс, 2002. 624 с.
4. *Канер С., Фолк Д., Нгуен Е. К.* Тестирование программного обеспечения. Фундаментальные концепции менеджмента бизнес-приложений = Testing Computer SoftWare: [пер. с англ.]. Киев: ДиаСофт, 2001. 543 с.
5. *Тамре Л.* Введение в тестирование программного обеспечения: [пер. с англ.] М.: Вильямс, 2017. 368 с.
6. НОУ ИНТУИТ | Основы тестирования программного обеспечения | Информация [электронный ресурс]. URL: https://www.intuit.ru/studies/courses/48/48/info (дата обращения: 21.03.2019).
7. *Вишневская Т. И.* Тестирование программного обеспечения — как учебная дисциплина // Образовательные ресурсы и технологии. 2014. № 4. С. 83–89.
8. *Полевщиков И. С.* Особенности изучения способа тестирования базового пути студентами бакалавриата в рамках дисциплины “Тестирование программного обеспечения” (часть 1) // Молодой ученый. 2015. № 18. С. 10–12.
9. *Чикунов В. Д., Ломаш Д. А.* Новые подходы в подготовке IT-специалистов в современных условиях // Транспорт: наука, образование, производство. Сборник научных трудов Международной научно-практической конференции. 2017. С. 221–224.
10. Download Qt: Choose commercial or open source [электронный ресурс]. URL: https://www.qt.io/download (дата обращения: 21.03.2019).
11. GitHub — seekerk/gtest: gtest + qmake + travis + coveralls [электронный ресурс]. URL: https://github.com/seekerk/gtest (дата обращения: 21.03.2019).
12. GitHub — google/googletest: Googletest — Google Testing and Mocking Framework [электронный ресурс]. URL: https://github.com/google/googletest (дата обращения: 21.03.2019).
13. The world’s leading software development platform — GitHub [электронный ресурс]. URL: https://github.com (дата обращения: 21.03.2019).
14. Travis CI — Test and Deploy Your Code with Confidence [электронный ресурс]. URL: https://travis-ci.org (дата обращения: 21.03.2019).
15. Coveralls — Test Coverage History & Statistics [электронный ресурс]. URL: https://coveralls.io (дата обращения: 21.03.2019).
16. SonarCloud | Clean Code, Rockstar Status [электронный ресурс]. URL: https://sonarcloud.io/ (дата обращения: 21.03.2019).
17. Вики // Википедия. [2017—2017]. Дата обновления: 08.05.2017. URL: https://ru.wikipedia.org/?oldid=85297238(дата обращения: 21.03.2019).
18. *Голицына И. Н.* Технология Вики в организации учебной деятельности // Школьные технологии. 2014. № 4. С. 108–114.
19. *Болгариева Е. В.* Технологии WEB 2.0 в образовательной деятельности. 2014. С. 99–104.
20. MediaWiki [электронный ресурс]. URL: https://www.mediawiki.org/wiki/MediaWiki (дата обращения: 21.03.2019).

21. ГОСТ 19.001-77. Единая система программной документации. Общие положения. М.: Стандартинформ, 2010. 6 с.
22. ГОСТ 19.301-79. Единая система программной документации. Программа и методика испытаний. М.: Стандартинформ, 2010. 6 с.
23. IEEE Standard for Software Unit Testing, in ANSI/IEEE Std 1008-1987, 1986.
24. IEEE Standard for Software Test Documentation, in IEEE Std 829-1998, 1998. С. 1–64.

Поступила в редакцию 22.01.2019, окончательный вариант — 21.03.2019.

Computer tools in education, 2019
№ 1: 88–100
http://ipo.spb.ru/journal
doi:10.32603/2071-2340-2019-1-88-100

Complex Approach for Learning Software Testing in the Educational Process PetrSU

Dimitrov V. M.[1], lecturer, dimitrov@cs.petrsu.ru
Kulakov K. A.[1], PhD, associate professor, kulakov@cs.petrsu.ru

[1]Petrozavodsk state University, 33, Lenina st., 185910, Republic of Karelia, Petrozavodsk, Russia

Abstract

The article contains information on the organization of training of specialists in the field of information technology at Petrozavodsk State University and the tools used to do this in one of the fundamental stages of software development, i.e. testing. Three stages are described: familiarity with testing technology (1 course, undergraduate), testing as a development stage of a team project (3 courses, undergraduate) and organization of the testing process (2 courses graduate). For each stage a set of tools is considered, which are used in the educational process. Training in stages is built from the simple use of tools to controling the testing process in the development team.

Keywords: *information technology, software engineering, testing, quality assurance, educational process.*

Citation: V. M. Dimitrov and K. A. Kulakov, “Complex Approach for Learning Software Testing in the Educational Process PetrSU,” *Computer tools in education*, no. 1, pp. 88–100, 2019 (in Russian); doi: 10.32603/2071-2340-2019-1-88-100

References

1. “Ob utverzhdenii prioritetnykh napravlenii razvitiya nauki, tekhnologii i tekhniki v Rossiiskoi Federatsii i perechnya kriticheskikh tekhnologii Rossiiskoi Federatsii: ukaz Prezidenta Rossiiskoi Federatsii ot 07.07.2011 g. № 899” [On approval of the priority directions of development of science, technology and technology in the Russian Federation and the list of critical technologies of the Russian Federation: Decree of the President of the Russian Federation of 07.07.2011, no. 899], [Online]. Available: http://pravo.gov.ru/proxy/ips/?docbody=&nd=102149065&rdk=&firstDoc=1&lastDoc=1 (in Russian).

2. M. V. Oseledets and T. A. Novikova, “K voprosu o motivatsiyakh studentov k obucheniyu” [On the issue of student motivation to learn], *Innovatsionnaya ekonomika i obshchestvo*, no. 2, pp. 67–73, 2014 (in Russian).
3. I. Sommerville, *Inzheneriya programmnogo obespecheniya* [Software engineering], 6th ed., Moscow, Russia: Williams, 2002 (in Russian).
4. C. Kaner, J. Falk, and H. Q. Nguyen, *Testirovanie programmnogo obespecheniya. Fundamental’nye kontseptsii menedzhmenta biznes-prilozhenii* [Testing Computer Software], Kiev, Ukraine: DiaSoft, 2001 (in Russian).
5. L. Tamre, *Vvedenie v testirovanie programmnogo obespecheniya* [Introduction to software testing], Moscow, Russia: Williams, 2017 (in Russian).
6. “NOU INTUIT. Osnovy testirovaniya programmnogo obespecheniya” [NOU INTUIT. Software Testing Basics], [Online]. Available: https://www.intuit.ru/studies/courses/48/48/info (in Russian).
7. T. I. Vishnevskaya, “Testirovanie programmnogo obespecheniya — kak uchebnaya distsiplina” [Software testing — as an academic discipline], *Obrazovatel’nye resursy i tekhnologii*, no. 4, pp. 83–89, 2014 (in Russian).
8. I. S. Polevshchikov, “Osobennosti izucheniya sposoba testirovaniya bazovogo puti studentami bakalavriata v ramkakh distsipliny “Testirovanie programmnogo obespecheniya” (chast’ 1)” [Features of studying the method of testing the basic path of undergraduate students in the discipline “Software Testing” (Part 1)], *Molodoi uchenyi*, no. 18, pp. 10–12, 2015 (in Russian).
9. V. D. Chikunov and D. A. Lomash, “Novye podkhody v podgotovke IT-spetsialistov v sovremennykh usloviyakh” [New approaches in the training of IT-specialists in modern conditions], *Transport: nauka, obrazovanie, proizvodstvo. Sbornik nauchnykh trudov Mezhdunarodnoi nauchno-prakticheskoi konferentsii*, Rostov-on-Don, Russia: Rostovskii gosudarstvenyni universitet putei soobshchenii, 2017, pp. 221–224 (in Russian).
10. “Qt. download,” *Qt Group*, [Online]. Available: https://www.qt.io/download
11. “GitHub — seekerk/gtest: gtest + qmake + travis + coveralls,” GitHub, [Online]. Available: https://github.com/seekerk/gtest
12. “GitHub — google/googletest: Googletest — Google Testing and Mocking,” GitHub, [Online]. Available: https://github.com/google/
13. “The world’s leading software development platform — GitHub,” GitHub, [Online]. Available: https://github.com
14. “Travis CI — Test and Deploy Your Code with Confidence,” *Travis CI*, [Online]. Available: https://travis-ci.org
15. “Coveralls — Test Coverage History & Statistics,” *Coveralls*, [Online]. Available: https://coveralls.io
16. “SonarCloud | CleanCode, RockstarStatus,” *SonarCloud*, [Online]. Available: https://sonarcloud.io/
17. “Wiki,” *Wikipedia*, [Online]. Available: https://ru.wikipedia.org/?oldid=85297238 (in Russian).
18. I. N. Golitsyna, “Tekhnologiya Viki v organizatsii uchebnoi deyatel’nosti” [Technology wiki in the organization of educational activities], *Shkol’nye tekhnologii*, no. 4, pp. 108–114, 2014 (in Russian).
19. E. V. Bolgarieva, *Tekhnologii WEB 2.0 v obrazovatel’noi deyatel’nosti* [WEB 2.0 technologies in educational activities], 2014.
20. “MediaWiki,” [Online]. Available: https://www.mediawiki.org/wiki/MediaWiki
21. *Edinaya sistema programmnoi dokumentatsii. Obshchie polozheniya.* [Unified software documentation system. General provisions], GOST 19.001–77, 2010 (in Russian).
22. *Edinaya sistema programmnoi dokumentatsii. Programma i metodika ispytanii.* [Unified software documentation system. Program and test methods.], GOST 19.301–79, 2010 (in Russian).
23. *IEEE Standard for Software Unit Testing*, ANSI/IEEE Std 1008-1987, 1986.
24. *IEEE Standard for Software Test Documentation*, IEEE Std 829-1998, 1998.

Received 22.01.2019, the final version — 21.03.2019.