УДК 005.8

А.С. Наркевич, ст. преп. (БГТУ, г. Минск)

ОРГАНИЗАЦИЯ КОМАНДНОЙ РАБОТЫ СТУДЕНТОВ НАД ПРОЕКТОМ НА ОСНОВЕ ГИБКИХ МЕТОДОВ УПРАВЛЕНИЯ ПРОЕКТАМИ AGILE

В докладе предоставлен опыт организации командной работы студентов первого курса над проектом на основе гибких методов управления проектами Agile с целью приобретения практических навыков анализа, проектирования, тестирования, документирования и разработки небольших программных комплексов.

Дисциплина «Основы программной инженерии» является базовой для студентов специальности 1-40 01 01 «Программное обеспечение информационных технологий». Целью курса является ознакомление с принципами организации и создания надежного, качественного программного обеспечения (ПО), удовлетворяющего предъявляемым к нему требованиям.

Основными задачами освоения дисциплины являются:

– рассмотрение технологических основ процесса разработки программного обеспечения;

– приобретение навыков анализа, проектирования, тестирования, документирования и разработки небольших программных комплексов;

– освоение инструментальных средств разработки;

– получение практического опыта работы в команде.

Заключительная лабораторная работа курса, длительностью 6 часов, является итоговой, и выполняя ее, студенты участвуют в командной работе над собственным проектом. Цель работы:

– изучить жизненный цикл разработки программного обеспечения;

– изучить методологии проектирования программных продуктов;

– рассмотреть технологии быстрой разработки программного обеспечения;

– выполнить проектирование и разработку отдельных элементов системы;

– изучить возможности использования унифицированного языка моделирования при проектировании программных систем;

– построить диаграмму вариантов использования;

– выполнить тестирование;

– внедрение и сопровождение программных продуктов;

– получить навыки совместной работы в команде.

Группа студентов делится на небольшие команды, состоящие из 5–6 студентов, для разработки собственного проекта.

Назначение, цель, имя проекта, описание, входные/выходные данные, модули, входящие в состав проекта (возможен выбор из предлагаемого в задании списка задач различной сложности), используемые инструменты управления проектами определяются командой.

Начальные требования к проекту предоставляются в виде списка задач (backlog), из которого студенты составляют sprint backlog.

В качестве методологии процесса разработки ПО студентам было предложено проанализировать различные методологии проектирования ПО, рассмотренные в лекционном курсе, и выбрать подходящие модели для работы над проектом. Рекомендовано использовать Agile – гибкий итеративно-инкрементальный подход к управлению проектами и продуктами, ориентированный на динамическое формирование требований и обеспечение их реализации небольшой группой разработчиков. Согласно данному подходу, весь проект разбивается на несколько небольших подпроектов, из которые затем формируется готовый продукт.

Таким образом, инициация и верхнеуровневое планирование проводятся для всего проекта, а этапы разработка, тестирование и прочие проводятся для каждого модуля отдельно. Это обеспечило успешное выполнение проекта каждой командой не зависимо от уровня подготовки участников команды.

Студенты проанализировали различные методологии и выбрали подходящие модели работы над проектом, управления командой, определили инструментарий для взаимодействия участников команды на протяжении всего жизненного цикла проекта.

Из множество методов, базирующихся на идеях Agile, использовались Scrum, Kanban и даже методология XP.

Вне зависимости от применяемой методологии первым этапом разработки является формулировка требований к продукту.

Набор требований к продукту представляет собой техническое задание, при этом требования делятся на функциональные (что система позволяет сделать, желаемая функциональность, с точки зрения её поведения и информации, которыми она будет управлять) и нефункциональные (требования к оборудованию, операционной системе и т. п.).

Для формализации функциональных требований команды строят UML-диаграммы вариантов использования, которые являются ос-

новой для пользовательской документации, а также используются для написания тестов.

В ходе разработки проекта команды проводили встречи, в ходе которых обсуждались ближайшее задачи, требующие выполнения, и дальнейшее развитие проекта, новые идеи и предложения.

Студентами применялись следующие способы организации совместной работы над проектом:

– удаленный репозиторий в Github;

– канбан-доска (добавление задачи, назначение исполнителей, изменение статуса задачи, проведение тестирования);

– средство Issues в GitHub для организации обсуждения, назначение проблем и задач членам команды.

Роли и задачи распределись между членами команды самостоятельно, в порядке обсуждения, кроме того участниками проекта были согласованы правила совместной работы с удаленным репозиторием.

Разработка проекта проводилась в IDE Visual Studio с использованием системы контроля версий.

Процесс тестирования включил в себя составление плана тестирования, набора тестовых случаев, выполнение тестирования каждого модуля и проекта в целом. Тестовый случай содержит описание тестовых сценариев. Каждый такой случай привязывается к соответствующему Use Case.

Лидер команды представляет готовую версию проекта. Члены команды анализируют полученный результат и принимают решение о продолжении или завершении работы над проектом. При необходимости выполняется планирование следующего спринта.

В соответствии с заданием документация на проект может содержать следующие типы документов: техническое задание, частное техническое задание на модуль/задачу, сценарий использования (Use Case), сценарий тестирования (Test Case), отчет об ошибке (Bug Report), руководство пользователя.

Основные вопросы управления командой проекта (ролевая модель команды, общение в команде) решались студентами самостоятельно.

Пример студенческого проекта BSTUdent.

Общение, разработка, обмен файлами и разработка были организованы с помощью системы контроля версий Git и GitHub. Был создан репозиторий с названием BSTUdent Hub, содержащий 6 веток: main, styles, INFO, timetable, game, binary-code. В репозиторий были добавлены контрибьютеры – члены команды, каждый член команды использовал отдельную ветку для разработки.

Проект включает в себя следующие модули: расписание занятий группы, калькулятор двоичных чисел, игра «Змейка», реализованная на C++, Time Table.

В ходе тестирования, в зависимости от результата теста (failed/passed), формировалась задача и карточка с заданием на доработку помещалась в Kanban для устранения проблем, указанных тестировщиком, или переходила на следующую стадию.

В качестве инструмента совместной работы использовался GitHub и следующие его возможности:

– добавление членов команды, настройка прав доступа для членов команды;

– Pull Requests – отправка запроса и слияние;

– Issues – отслеживание ошибок, обсуждение новых идей для реализации, список задач;

– ревью кода – комментарии к строкам и URL-запросы;

– wiki – создание документации на проект.

Была создана доска «Канбан», для размещения карточек с основным планом разработки той, или иной части проекта.

Студентами применялись следующие дополнительные возможности issues Github:

– Labels: выделение цветом категорий для каждого issue (полезны для фильтрации);

– Milestones: вехи, связанные с категориями (для определения того, какие проблемы требуют обработки, и также позволяет автоматически обновлять индикатор выполнения при закрытии связанной с вехой проблемы);

– Search: поиск для issues и milestones;

– Assignment: каждый вопрос может быть назначен ответственному лицу для исправления проблемы;

– автоматическое закрытие проблемы, помеченной как выполненная (Fixes#[issue-number]);

– Mentions: позволяет оставить примечание, указывая #[issue-number], т.к. номера issue являются гиперссылками.

В результате выполнения собственного уникального проекта студенты научились определять цель, ставить задачи, составлять и реализовывать план проекта; получили опыт работы над проектом – первый опыт совместной деятельности в процессе выполнения творческого задания, развили способность к коммуникации.