Министерство науки и высшего образования Российской Федерации
Севастопольский государственный университет

Реферат
по дисциплине «Технологии создания программных продуктов»
на тему «Методология программирования XP (Extreme Programming)»

Выполнил: студент группы ИС/б-20-3-о
Юрьева П.В.
Проверил:
ст. преп. кафедры ИС
Строганов В.А.

2022

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Создание программных продуктов – сложный процесс, основанный на определенной технологии и инструментарии его разработки. На сегодняшний день одной из самых динамично развивающихся составляющих в области программной инженерии являются методологии разработки программного обеспечения. Подробному рассмотрению одной из них и посвящён настоящий реферат.

Существование множества различных методологий разработки программного обеспечения связано с тем, что ни одна из них не является универсальной. А выбор методологии разработки для конкретного проекта зависит от предъявляемых требований.

Целью реферата является ознакомление с методологией разработки программных продуктов XP (Extreme Programming), укрепление знаний, полученных на лекционных занятиях «Технологии создания программных продуктов», а также получение практических навыков работы с литературными источниками.

Установим основные понятия.

Методология - принципы и способы организации теоретической и практической деятельности. Совокупность методов, применяемых в какой-либо науке» [1].

Применительно к разработке программного обеспечения, это определение можно переформулировать так: «Методология есть принципы и способы организации деятельности проектной группы для создания программного продукта».

Если разбивать данное определение на элементы, то можно выделить два базовых понятия:

1) Проектная группа – это коллектив людей, непосредственно занятых созданием готового решения. Именно группа (команда) использует методологию, так как в организации деятельности людей и состоит основное

назначение методологий.

2) Программный продукт. Именно продукт является конечной целью в любой методологии. В последнее время, вместо термина «программный продукт», все чаще используют термин «решение» («solution»).

Разумеется, XP является не единственной методологией разработки. Например, широко известна методология Scrum, которая так же, как и XP является одной из составляющих Agile, но они имеет достаточно серьезные различия. XP является методологией разработки ПО, а Scrum – методологией управления проектами. Также, в пример можно привести различие в длительности итераций (спринтов) - в XP команды работают в итерациях, которые длятся 1-2 недели, в то время как в Scrum спринты составляют по времени от 2 до 4 недель.

1. ИСТОРИЯ ВОЗНИКНОВЕНИЯ МЕТОДОЛОГИИ

Данная методологии была разработаны Кентом Беком в 1996 году, во время того как он работал над проектом системы для расчета зарплатных ведомостей Chrysler Comprehensive Compensation System. В процессе работы над проектом было необходимо ускорить разработку, и Кентом Беком было предложено сосредоточиться на главном и отбросить остальное, чтобы улучшить выполнение сроков. В компании данный подход не прижился, но исходя из результатов работы Бек написал книгу, которая объясняла данный подход.

Два основных влияния сформировали разработку программного обеспечения в 1990-х годах:

1. Объектно-ориентированное программирование заменило процедурное программирование как парадигму программирования, предпочитаемую некоторыми разработчиками.
2. Внешний рост Интернета подчеркивал скорость выхода на рынок и рост компании как конкурентные бизнес-факторы.

Быстро меняющиеся требования требовали более коротких жизненных циклов продукта и часто вступали в противоречие с традиционными методами разработки программного обеспечения [2].

XP основана на ценностях, принципах и практиках, и ее цель - позволить небольшим и средним командам создавать высококачественное программное обеспечение и адаптироваться к меняющимся требованиям.

Основной идеей методологии является устранить высокую стоимость изменений, вносимых в ПО в процессе как разработки, так и эксплуатации.

2. СОДЕРЖАНИЕ МЕТОДОЛОГИИ

2.1 Понятие методологии XP

Как описывает Бек в своих записях [5], экстремальное программирование - это методология разработки программного обеспечения, которая является частью того, что в совокупности известно, как. Agile –гибкий подход к разработке ПО в команде разработчиков.

Agile представляет собой группу различных методик гибкого управления проектами в команде разработки. Процесс разбивается на так называемые итерации (спринты), в процессе которого команда разрабатывает часть продукта, которую в дальнейшем можно оценить и протестировать.

Один из основных принципов Agile – удовлетворение потребностей заказчиков. Частые регулярные поставки в ПО в одинаковый промежуток времени показывают, что в работе есть прогресс. Также необходимо в процессе всего цикла разработки поддерживать сотрудничество между разработчиками и заказчиком, так как непосредственное общение является наиболее эффективным методом работы [6].

К гибким методологиям относят: XP (Extreme Programming), DSDM (Dynamic Systems Development Method), Scrum, FDD (Feature driven development), BDD (Behavior-driven development) и другие.

Как было сказано выше, основная идея методологии экстремального программирования (далее XP) – уменьшить стоимость изменений в ПО в процессе разработки и эксплуатации.

2.2 Основные ценности методологии

Как показано в [4], Кент Бек сформулировал 4 ключевые ценности XP:

1. Коммуникация (Communication). Если при работе над проектом возникает проблема, зачастую ее причиной является то, что в какой-то момент

времени кто-то не сказал кому-то что-то важное.

2. Простота (Simplicity). XP всегда предполагает, что решение задачи будет произведено простым способом из всех возможных.

3. Обратная связь (Feedback). Чем раньше и чаще будут получены отзывы о разрабатываемом продукте, тем точнее можно выбрать дальнейший курс развития проекта.

4. Храбрость (Courage). Храбрость необходима для того, чтобы в нужные моменты идти на рискованные меры и, например, отказываться от кода, на разработку которого было потрачено большое количество времени, или же, обнаружив ошибку, вернуться назад.

Сама методология имеет 12 принципов, разбитых на 4 группы. Она ориентирована на группы до 10-50 человек, находящихся в одном помещении. Время, затрачиваемое на прохождение одной итерации – от 2 до 4 недель. Рассмотрим каждую группу с принципами ниже.

2.3. Группы и принципы методологии

В источнике [4] регламентируются такие группы, на которые делятся методологии:

Первая группа. Короткий цикл обратной связи.

— **Игра в планирование**:

Данный принцип подразумевает диалог между заказчиком разработчиками, с целью определить текущие задачи, приоритеты, сроки выполнения.

Основная цель – быстро сформулировать приблизительный план работы и постепенно в процессе работы дополнять его, по мере добавления четкости задач.

Со стороны разработчиков формулируются следующие задачи:

1. Оценить время для реализации каждого пожелания

2. Оценить затраты на технологии, которые будут использоваться во время реализации требований заказчика

3. Грамотно распределить задачи между членами команды

4. Оценит риск, который может возникнуть во время реализации проекта

5. Определить порядок действий, в котором будут реализованы все пожелания заказчика (учитывая текущую итерацию). При выполнении рискованных пожелании в первую очередь можно снизить общий риск проекта.

Со стороны заказчика регламентируется:

1. Объем работ (набор пожеланий для выпуска и набор пожеланий для итерации);

2. Дата завершения (дедлайн) работы;

3. Назначение приоритетов – исходя из полезности запрашиваемых функций продукта необходимо определить, что необходимо реализовать в первую очередь.

— **Заказчик на рабочей площадке**:

Смысл принципа в том, что любой разработчик может задать вопрос, получить любую нужную консультацию с заказчиком. Это необходимо для того, чтобы разработчики могли работать с оптимальной скоростью.

— **Разработка через тестирование**:

В XP используется две разновидности тестирования:

1. Тестирование модулей (unit testing)

Тесты модулей разрабатываются по мере того, как разработчики пишут код. Данные тесты в любой момент могут сообщить заказчику, что код функционирует нормально. Можно сказать, что тесты модулей– разновидность документации о том, что процесс разработки идет и все работает корректно.

2. Приемочное тестирование

Данное тестирование осуществляется непосредственно заказчиком, основываясь на собственных пожеланиях. Они позволяют заказчику

убедиться, что все указанные пожелания по системе действительно там присутствуют.

— **Парное программирование:**

Любой программный код в XP реализуется двумя разработчиками, которые одновременно реализуют поставленную задачу. Это позволяет повысить качество кода, скорость решения задачи, а также происходит обмен опытом между разработчиками. Еще одним преимуществом парного программирования является то, что в разработанном коде будут разбираться два человека, благодаря чему знание о внутреннем устройстве системы быстро распространится между членами команды.

Вторая группа. Процесс разработки организуется непрерывно, а не пакетно:

— **Частный выпуск небольших обновлений**:

Данный принцип дает понять, удовлетворяет ли разрабатываемое решение пожеланиям заказчика. Чем раньше заказчик приступит к эксплуатации разрабатываемой системы, тем быстрее от него поступит информация о соответствиях требованию, что позволит направить разработку решения в необходимое русло.

— **Непрерывная интеграция**:

Данный принцип также позволяет избежать проблем, связанных с правильностью работы модулей. В XP интеграция происходит несколько раз в день, после того, как код прошел все написанные тесты.

— **Использование рефакторинга**:

Рефактрингом называют методику уличения кода без преобразования его функциональности. Данный принцип позволяет сохранить код чистым и аккуратным, а это значит, что в дальнейшем количество проблем с кодом может быть меньше, и его понимание другими будущими разработчиками будет гораздо выше.

Третья группа. Понимание проекта всеми участниками.

— **Метафора системы**:

Выступает в качестве аналога архитектуры системы. Основная цель заключается в том, чтобы каждый участник проекта понимал архитектуру, основные функции и назначение будущего программного продукта. При этом, текущий проект может описываться путём сравнения с ранее выполненными проектами.

— **Простота проектирования**:

При проектировании желательно придерживаться простоты и ясности, что гарантирует что проект будет разрабатываться быстрее. При использовании простого проектирования сам проект будет проще подвергнуть изменениям в процессе разработки под новые требования.

— **Простота дизайна**.

Под простым дизайном Кент подразумевает:

1. Корректное срабатывание всех тестов;
2. Отсутствие дублирующего кода;
3. Код хорошо выражает намерения программиста для каждого из участков кода;
4. Использование классов и методов по минимуму.

Стоит учитывать, что, если система обладает простым дизайном, это не значит, что она маленькая. Дизайн должен быть максимально простой, но при этом реализовывать весь требуемый функционал.

— **Стандарты кодирования**:

Данный принцип подразумевает, что в команде устанавливаются общие стандарты кодирования. Благодаря наличию стандартов, члены команды не тратят время на ненужные споры, и обеспечивается эффективное исполнение остальных практик.

Если же в команде не используются стандарты, то это усложняет выполнение рефакторинга – при смене партнера в паре возникают затруднения, и работа замедляется.

— **Коллективное владение кодом**:

Любой разработчик может вносить изменения в любой фрагмент кода программы. За каждый участок кода отвечает, как минимум два разработчика, но ответственность за работоспособность кода несут все участники команды. Данный принцип повышает универсальность программистов. Если при редактировании кода нарушается работоспособность системы, то действует правило «Кто сломал, тот и исправляет».

Четвёртая группа. Социальная защита программиста.

— **Обязательная 40 часовая рабочая неделя:**

Важно, чтобы работоспособность и продуктивность каждого члена команды сохранялось на протяжении всего процесса разработки системы. Автор данной методологии считает, что для эффективной работы необходимо работать не более 40 часов в неделю. Даже если есть возможность работать сверхурочно, не обязательно заниматься этим постоянно – может произойти выгорание, и производительность разработчика пострадает.

Считается, что самыми главными практиками являются:

1. Планирование и предварительная оценка трудозатрат;
2. Частые выпуски версий и короткие итерации;
3. Предварительное тестирование (тестирование модулей);
4. Парное программирование;
5. Рефакторинг;
6. Постоянная интеграция.

Также рекомендуется проводить собрания команды каждое утро для поддержания общего уровня владения информацией [5].

2.4 Внедрение XP

Рекомендации, которые дает Кен Ауэр в [4]:

Одной из особенностей XP является то, что ее можно применять

практически к любой человеческой деятельности. Данная методология предписывает поступать самым простым методом из всех возможных.

Автор методологии рекомендует при внедрении методологии в первый раз выполнить данные шаги:

1. Найти подходящий проект;
2. Найти компаньона;
3. Найти цель работы;
4. Собрать все необходимые инструменты.

При соблюдении данных советов можно будет беспрепятственно приступить к изучению и внедрению практик XP в команду.

На рисунке 1 графически изображено, как можно внедрить XP в команду, и как будет происходить ежедневная работа.

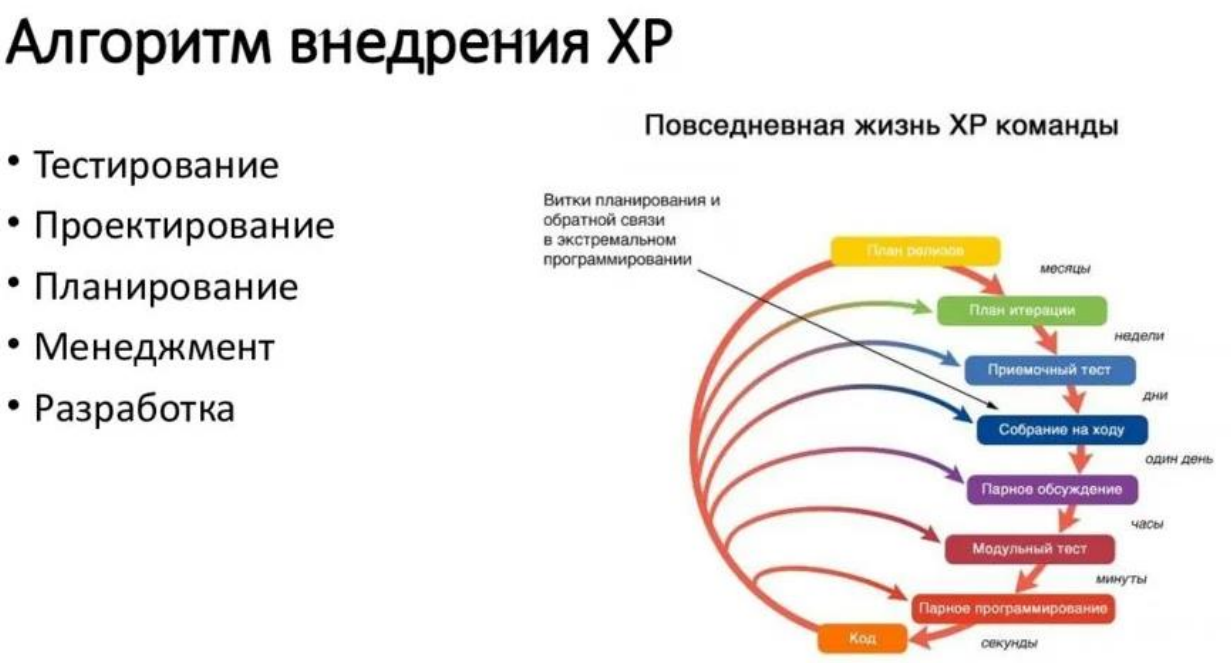

Рисунок 1 - Алгоритм внедрения XP

3. ИЗВЕСТНЫЕ ПРИМЕНЕНИЯ МЕТОДОЛОГИИ

Любая компания-разработчик программного обеспечения может использовать метод экстремального программирования при выполнении своих проектов. Единственное, что они должны сделать перед началом работы с XP, - это понять ее основные принципы и теоретические основы.

Одним из самых популярных применений XP – в Chrysler Comprehensive Compensation System, где и зародилась данная методология.

Некоторые примеры проектов:

1. Компания Acxiom, Джи Ханнула

Приложение: база данных управления кампанией;

Срок реализации: 3 года;

В компании Acxiom на основе склада данных создали приложение управления бизнесом. Небольшая команда разработчиков - всего 10 человек - при создании приложения твердо придерживалась принципов объектно-ориентированного программирования и коллективной разработки. Из трех лет, затраченных на разработку, в течение двух последних команда использовала методы экстремального программирования и именно благодаря этому достигла успеха [3].

2. Ford Motor, Дон Уэлс.

Приложение: система анализа затрат;

Срок реализации: 6 лет;

Отдел финансовых систем компании Ford Motor разрабатывает аналитическую систему Vehicle Costing and Profit System (VCAPS), которая создает отчеты по доходам от производства, расходам, чистому доходу и прибыли. Входными данными для системы являются инженерные спецификации продукции, фиксированные затраты и расходы и переменные затраты, например, рабочие часы. VCAPS аккумулирует все эти данные и подготавливает подробные отчеты с анализом затрат, которые обеспечивают эффективное прогнозирование и принятие корпоративных решений. Работа

над проектом VCAPS была начата в 1993 году.

С помощью XP было достигнуто то что оперативное реагирование на изменения позволило достигнуть такого качества системы, которое позволило избежать опасных перезапусков. Спустя полтора года использования XP количество сбоев снизилось, и система имела высокую стабильность [3].

ЗАКЛЮЧЕНИЕ

Extreme Programming – это комбинация практик, и при использовании верных комбинаций (можно использовать одну из практик, или же некоторое количество практик) существенно поднимется скорость и качество разработки.

Перечислим достоинства данной методологии:

1) Методология имеет большую гибкость за счет коротких итераций и регулярных внесений изменений, основанных на обратной связи заказчика;

2) Данная методология предоставляет возможность быстро и аккуратно внести изменения в ПО при необходимости;

3) Высокое качество получающегося в результате кода;

4) Благодаря регулярным общениям заказчика и разработчиков о требованиях и пожеланиях, в конце нет необходимости в убеждении заказчика в том, что результат соответствует его ожиданиям.

Но стоит отметить, что у данной методологии есть и недостатки:

1) При использовании данной методологии невозможно выполнить достаточно сложные и большие проекты;

2) Невозможно планировать сроки и трудоемкость проекта на достаточно долгую перспективу, а также нельзя четко предсказать длительные результаты проекта в отношении качества результата и затрат времени и ресурсов;

3) Данная методология не приспособлена для тех случаев, когда возможные решения не находятся на основе ранее полученного опыта, а требуют затраты времени на исследование.

Экстремальное программирование стоит внедрять, когда задача, которую поставил заказчик, небольшая, и есть возможность регулярного общения с заказчиком, а также команда может самоорганизоваться и проявить храбрость – вовремя отказаться от ненужного кода или поменять развитие в другую сторону.

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. С.И. Ожегов, Н.Ю. Шведова Толковый словарь русского языка. - 2-е изд. - Москва: Азь, 1992

2. Технологии экстремального программирования // НИУ "Высшая школа экономики" URL: https://publications.hse.ru/mirror/pubs/share/folder/c9va6oio94/direct/140075183 (дата обращения: 24.05.2022).;

3. Экстремальное программирование // Издательство "Открытые системы" URL: https://www.osp.ru/os/2000/01-02/178193 (дата обращения: 23.05.2022).

4. Кен Ауэр, Рой Миллер Экстремальное программирование: постановка процесса с первых шагов и до победного конца. - 2-е изд. - Питер, 2003

5. Кент Бек, Мартин Фаулер Экстремальное программирование: планирование. - 2-е изд. - Питер: 2003. - 144 с.;

6. Официальный сайт с публикациями о Extereme Programming // Ronjeffries URL: https://ronjeffries.com/ (дата обращения: 25.05.2022).