Потёмкин А.А., студ. гр. 508 ФК и СКД
БГУ культуры и искусств
Научный руководитель – Гончарова С.А., доцент

МЕТОДОЛОГИЯ СОЗДАНИЯ САЙТА УЧРЕЖДЕНИЯ КУЛЬТУРЫ

За последние несколько лет сильно вырос интерес к так называемым гибким методологиям разработки программного обеспечения (известным также под названием "облегченных методологий").

Как правило, разработка программного обеспечения представляет собой довольно хаотическую деятельность, которую нередко можно охарактеризовать фразой "code and fix" ("пишем и правим"). Единого плана не существует, а общий проект представляет собой просто смесь краткосрочных решений. Такой подход может сгодиться для создания небольшой системы, однако если система начинает расти, добавлять в нее новые свойства становится все более затруднительно. Кроме того, в ней будет появляться все больше ошибок, которые будет все труднее исправлять. Типичные признаки такой системы - долгий тестовый период уже после того, как разработка всей функциональности системы закончена. При этом нарушаются все планы выпуска программы, так как при подобном тестировании и исправлении ошибок адекватное планирование просто невозможно.

Именно так многие и работали довольно продолжительное время. Впрочем, у них всегда была альтернатива - использовать методологию. Методология превращает создание программного продукта в упорядоченный процесс, с помощью которого можно сделать работу исполнителя более прогнозируемой и эффективной. Для этого создается детальное описание процесса создания системы, особое место в котором занимает планирование (аналогично другим инженерным дисциплинам).

РЕПОЗИТОРИЙ БГУКИ

Такие методологии существуют уже давно. Нельзя сказать, что они очень уж эффективны. С еще меньшей степенью уверенности можно говорить об их популярности. Чаще всего их обвиняют в бюрократизме - чтобы следовать такой методологии, нужно выполнять так много различных предписаний, что замедляется весь темп работ. Именно поэтому их называют тяжеловесными методологиями, или, согласно термину Джима Хайсмита (Jim Highsmith), - монументальными.

За последние годы в противовес этим методологиям появилась группа новых, которые раньше было принято называть облегченными (lightweight). Теперь для них используют другой термин - гибкие (agile) методологии. Привлекательность новых методологий для многих заключается в отсутствии бюрократизма, присущего монументальным методологиям. Новые методологии представляют собой попытку достичь необходимого компромисса между слишком перегруженным процессом разработки и полным его отсутствием. Иначе говоря, объема процесса в них как раз достаточно, чтобы получить разумную отдачу.

XP (Extreme Programming). Из всех гибких методологий эта - самая известная. Отчасти это произошло потому, что лидеры XP, в особенности Кент Бек (Kent Beck) наделены замечательной способностью привлекать к себе внимание. Немаловажную роль сыграл и талант Кента вербовать сторонников своего движения и вести их за собой. Можно даже сказать, что популярность XP стала в некотором роде проблемой, так как эта методология практически вытеснила все остальные, а вместе с ними и те ценные идеи, которые они несут.

XP стоит на четырех китах: Коммуникация, Обратная связь, Простота и Смелость. Из них следуют двенадцать практик, которым должны следовать проекты, использующие XP.

РЕПОЗИТОРИЙ БГУКИ

Двенадцать основных приёмов экстремального программирования (по первому изданию книги Extreme programming explained) могут быть объединены в четыре группы:

- Короткий цикл обратной связи (Fine scale feedback)

1. Разработка через тестирование (Test driven development)
2. Игра в планирование (Planning game)
3. Заказчик всегда рядом (Whole team, Onsite customer)
4. Парное программирование (Pair programming)

- Непрерывный, а не пакетный процесс

1. Непрерывная интеграция (Continuous Integration)
2. Рефакторинг (Design Improvement, Refactor)
3. Частые небольшие релизы (Small Releases)

- Понимание, разделяемое всеми

1. Простота (Simple design)
2. Метафора системы (System metaphor)
3. Коллективное владение кодом (Collective code ownership) или выбранными шаблонами проектирования (Collective patterns ownership)
4. Стандарт кодирования (Coding standard or Coding conventions)

- Социальная защищенность программиста (Programmer welfare):

1. 40-часовая рабочая неделя (Sustainable pace, Forty hour week)

Парное программирование. Парное программирование предполагает, что весь код создается парами программистов, работающих за одним компьютером. Один из них работает непосредственно с текстом программы, другой просматривает его работу и следит за общей картиной происходящего. При необходимости клавиатура свободно передается от одного к другому. В течение работы над проектом пары не фиксированы: рекомендуется их перемешивать, чтобы каждый программист в команде имел хорошее представление о всей

РЕПОЗИТОРИЙ БГУКИ

системе. Таким образом, парное программирование усиливает взаимодействие внутри команды.

Коллективное владение. Коллективное владение означает, что каждый член команды несёт ответственность за весь исходный код. Таким образом, каждый вправе вносить изменения в любой участок программы. Парное программирование поддерживает эту практику: работая в разных парах, все программисты знакомятся со всеми частями кода системы. Важное преимущество коллективного владения кодом — в том, что оно ускоряет процесс разработки, поскольку при появлении ошибки её может устранить любой программист.

Давая каждому программисту право изменять код, мы получаем риск появления ошибок, вносимых программистами, которые считают что знают что делают, но не рассматривают некоторые зависимости. Хорошо определённые UNIT-тесты решают эту проблему: если не рассмотренные зависимости порождают ошибки, то следующий запуск UNIT-тестов будет неудачным.

Заказчик всегда рядом. «Заказчик» в XP — это не тот, кто оплачивает счета, а тот, кто на самом деле использует систему. XP утверждает, что заказчик должен быть всё время на связи и доступен для вопросов.

Что мне всегда импонировало в XP, так это та роль, которая в нем отводится тестированию. Все прочие процессы тоже упоминают тестирование, но делают это как-то поверхностно. Что касается XP, то в нем тестирование является той основой, на которой строится разработка. При этом каждый программист пишет тесты одновременно с кодом разрабатываемой системы. Эти тесты используются при постоянной интеграции и в процессе сборки системы, что дает стабильный фундамент для дальнейшей работы.

На этом фундаменте XP строит эволюционный процесс проектирования, основанный на реорганизации кода системы в течение каждой последующей итерации. При этом проектируется только та функциональность, которая

РЕПОЗИТОРИЙ БГУКИ

относится к текущей итерации, а любые будущие потребности не учитываются. Получившийся в результате процесс требует от разработчиков дисциплины, и в то же время сочетает ее с высокой адаптивностью. Такое удивительное сочетание позволяет предположить, что XP является наиболее развитой адаптивной методологией.

Список используемой литературы:

1. Адаптивная разработка (ASD) по Джиму Хайсмиту. [Электронный ресурс] – Режим доступа: http://www.cyberguru.ru/programming/programming-theory/coding-methodology-new-page13.html – Дата доступа: 22.04.2010.

2. Бек, Кент. Экстремальное программирование. – СПб.: Питер, 2002. – 80с.

РЕПОЗИТОРИЙ БГУКИ