УДК 681

ПРОБЛЕМА ВЫБОРА МЕТОДОЛОГИИ РАЗРАБОТКИ ИНФОРМАЦИОННОЙ СИСТЕМЫ ВУЗА

Петрова А. Н., Еськова А. В., Лошманов А. Ю.

ФГБОУ ВПО «Комсомольский-на-Амуре государственный технический университет», Комсомольск-на-Амуре, Россия (681013, г. Комсомольск-на-Амуре, пр. Ленина, 27), e-mail: petrovaan2006@yandex.ru

Работа посвящена анализу методологий разработки и управления ИТ-проектов и их применимости к задаче разработки информационной системы (ИС) вуза. Выбор методологии определяет основные этапы разработки, организацию взаимодействия с заказчиком, принципы работы команды, распределение ролей и ответственности в ней, выбор приоритетов и принципы управления ими. Большинство методологий включает несколько практик, некоторые из них позволяют совместное использование. В статье приведены описания нескольких наиболее популярных и перспективных методологий: Microsoft Solutions Framework, Dynamic Systems Development Method, Экстремальное программирование, Scrum, Featuredrivendevelopment. Указаны их особенности, достоинства и недостатки, возможности их применения к решению поставленной задачи и трудности, с которыми можно столкнуться при их использовании. В результате сделан выбор в пользу комбинации нескольких методик и практик из семейства Agile.

Ключевые слова: информационные системы, методы, методология управления проектом.

PROBLEM SELECTION METHODOLOGY INFORMATION SYSTEM DEVELOPMENT SCHOOL

Petrova A. N., Eskova A. V., Loshmanov A. YU.

Federal State-financed Educational Institution of Higher Professional Learning «Komsomolsk-na-Amure State Technical University», Komsomolsk-na-Amure, Russia (681013, Komsomolsk-na-Amure, Lenina, 27), e-mail:petrovaan2006@yandex.ru

The paper analyzes the development methodologies and management of IT projects and their applicability to the problem of development of information systems (IS) of the university. The methodology identifies the main stages of the development, organization of interaction with the customer, the principles of team work, roles and responsibilities in it, the choice of priorities and principles of management. Most methodologies includes several practices, some of them allow sharing. The article describes several of the most popular and promising methodologies: Microsoft Solutions Framework, Dynamic Systems Development Method, Extreme Programming, Scrum, Feature driven development. Are their characteristics, advantages and disadvantages, their applicability to the task and the difficulties that may be encountered when using them.As a result, opted for a combination of several techniques and practices of the family of Agile.

Keywords: information systems, practices, project management methodology.

Введение.

При создании ИС важным шагом является выбор методологии управления разработкой проекта [2]. Согласно анализу ежегодного опроса института управления проектами за 2012 год [3], (опрошено более тысячи управляющих проектами) выявлено, что

в условиях продолжающегося кризиса для поддержания конкурентоспособности, организации будут стремиться быть гибкими, что увеличит использование итеративных мер и инкрементных методов, типа Agile и Extreme.

Гибкие (Agile) методологии разработки – это не одна конкретная практика, а семейство процессов разработки, которое определяется AgileManifesto [1]. Agile не включает практик, а определяет ценности и принципы, которыми руководствуются успешные команды. Манифест определяет следующие *ценности:* «Люди и взаимодействие важнее процессов и инструментов. Работающий продукт важнее исчерпывающей документации. Сотрудничество с заказчиком важнее согласования условий контракта. Готовность к изменениям важнее следования первоначальному плану» [4].

Реализация данных приоритетов осуществляется через принципы управления разработкой ИТ проекта. Все гибкие методологии объединяют общие *принципы*:

– Заказчику программный продукт должен быть поставлен как можно быстрее. Для достижения этого принципа большой проект делится на небольшие части и реализуется в первой версии в максимально короткие сроки (от 2-х до 6 недель). Затем версии дорабатываются, расширяя функционал очередного релиза. Таким образом, создание проекта в целом является итерационной процедурой.

– Тесное взаимодействие с заказчиком. Заказчик – это конечный пользователь, он, как правило, участвует в нескольких этапах создания проекта, а не только в конце его реализации.

– Изменение требований заказчиком позволяет создать более качественный продукт, в большей степени удовлетворяющий его. Этот принцип достигается за счет выполнения первых двух. Поскольку получив продукт на ранней стадии, возможны изменения требований к информационной системе, а постоянная связь с заказчиком позволяет быстрее реагировать на изменения его требований и оперативно решать вопросы, возникающие при разработке ИТ проекта.

– Команда разработчиков должна быть мотивированной командой профессионалов, постоянно общающихся и стремящихся к повышению эффективности своей работы.

Семейство Agile насчитывает более десятка методик, рассмотрим наиболее популярные из них и проанализируем применения их к поставленной задаче.

Методология Microsoft Solutions Framework

Даже методология разработки, предложенная корпорацией Microsoft – MSF [7],

предлагает методику управления ИТ проектом с учетом принципов Agile. При разработке *модели процессов* MSF стремились соединить свойства каскадной и спиральной моделей. Модель включает следующие основные фазы: выработка концепции, планирование, разработка, стабилизация, внедрение. Фаза внедрения длится до момента, когда продукт начинает давать отдачу. Для каждой фазы определяется результат и зоны ответственности. Предлагается интерактивный подход к процессу разработки. Решение считается не законченным, пока не внедрено. Стратегия версионирования предполагает сначала реализацию основного функционала, а затем доработку дополнений в новых версиях.

Проектная группа состоит из шести ролевых кластеров, каждый из которых имеет свою зону ответственности: управление программой, разработка, тестирование, управление выпуском, удовлетворение заказчика, управление продуктом. Численность команды составляет от трех до десяти человек. Должность менеджера проекта отсутствует, а ответственность несут лидеры каждой из ролей.

Dynamic Systems Development Method

Метод разработки динамических систем (DSDM) [5] основан на концепции быстрой разработки приложений (RAD). Представляет собой итеративный и инкрементный подход, в котором важную роль играет участие пользователя. DSDM обычно используют для проектов ИС, со сжатыми сроками и бюджетами. Основные методики DSDM:

• Тайм-боксинг предполагает разделение всего проекта на части, каждая из которых имеет свой бюджет, срок выполнения и требования, которые были распределены по принципу MoSCoW. Поскольку время и бюджет фиксированы, то можно поменять только требования. Если проект не укладывается в график или бюджет, то требования с наименьшим приоритетом опускаются. Это соответствует принципу 20/80.

• Метод MoSCoW предоставляет путь распределения объектов по приоритетам. Must – требование должно удовлетворять экономическим нуждам. Should – следует ли выполнять это требование, если от него не зависит успех проекта. Could – нужно ли оставить это требование, если оно не действует на деловую потребность проекта. Would – можно ли отложить выполнение требования, если ещё есть время.

• Прототипирование – создание прототипов системы во время разработки на ранних этапах. Она позволяет выявить недостатки в системе и позволяет будущим пользователям протестировать её.

• Тестирование – проведение тестирования на каждой итерации.

• Моделирование с целью визуализировать в виде диаграмм той стороны системы, над которыми идёт работа.

К достоинствам можно отнести быстрое получение результата, малый объем документирования, раннее тестирование. К недостаткам также можно отнести неприменимость DSDM к проектам, где не применим принцип 20/80. К последним относятся системы, в которых важна безопасность данных. ИС вуза должна оперировать персональными данными, и вопрос безопасности является одним из важнейших. Поэтому применение при разработке ИС вуза методологии DSDM возможно лишь частично.

Экстремальное программирование

Рассмотрим основные идеи экстремального программирования (XP) [9], перенося их на процесс проектирования:

• *Разработка через тестирование*. Выделяют тестирование модулей и приемочное тестирование. Разработчик проверяет правильность кода путем выполнения всех тестов для модуля, а удовлетворение всех требований заказчика является приемочным тестированием. При проектировании объектом тестирования является разрабатываемая модель системы (объектная, процессная, субъектная), которая тестируется на полноту, неизбыточность, непротиворечивость и адекватность. Если в XP сначала пишется тест, а затем код по нему, то в проектировании сначала формулируются функциональные требования и информация, описывающая объект, а затем проектируется модель им удовлетворяющая.

• *Игра в планирование*. Присутствует разграничение ответственности: заказчик отвечает за принятие бизнес-решений (они фиксируются в виде функциональных требований), а разработчик принимает только технические решения. Такой подход можно полностью перенести на процесс проектирования.

• *Парное программирование*. Код модуля создается парой программистов совместно, заменяя друг друга. Затем пары меняются, таким образом, получается, что все разработчики свободно владеют кодом и могут вносить в него улучшения (рефакторинг), и интеграция происходит постоянно, без затруднений. При переносе подхода на проектирование в качестве кода модуля выступает разрабатываемая модель или ее декомпозиция. Рефакторингом является процесс изменения содержания модели, не изменяя ее функций и содержащейся информации. Благодаря парному проектированию достигается коллективное владение

моделями системы и выработка стандарта проектирования, т.е. нотации, точки зрения проектирования, правила представления информации, внесения изменения, интерфейсов и т.д.

• *Частые небольшие релизы*. В качестве релиза при разработке может выступать эскизный проект для конечного пользователя.

В отличие от других методологий, в XP заказчик должен быть всё время на связи и доступен для вопросов и предполагается взаимозаменяемость членов команды.

Достоинством будет являться совместное владение фактически всеми моделями проекта, а при проектировании это важно. Однако возможность одного члена команды изменить модель без согласования с остальными является минусом. В XP эта проблема решается тестированием, но при проектировании ошибки выявить труднее.

Scrum

Другим вариантом гибкой методологии управления разработкой информационной системы является **Scrum** [8]. Он, как Agile методология, является итерационным. Итерация в скрам называется *спритом*. Все пожелания заказчика (функциональные требования) хранятся в резерве проекта, отсортированные по степени важности. Перед началом каждого спринта обсуждается список задач и функций, которые должны быть в нем реализованы, таким образом формируется резерв спринта. Во время выполнения спринта требования и цели спринта не могут быть изменены, но его можно остановить, если цели перестали быть актуальны или невозможно их выполнить в срок.

Гибкость достигается за счет краткости спринтов – 2–4 недели. Сотрудничество с заказчиком выполняется на совещании при выборе списка задач для очередного спринта. В ходе работы над проектом в скрам предусмотрены несколько типов *совещаний* команды разработчиков. В начале спринта определяется резерв спринта и технические вопросы его реализации. Ежедневно выполняется отчет о результатах работы и возникших проблемах. По завершении спринта проводится демонстрация результатов и анализ работы команды.

В скрам-методологии заложено шесть ролей, которые можно разделить на две группы: разработчики (скрам-мастер, владелец проекта, команда) и пользователи (клиенты, управляющие персоналом, и эксперты-консультанты).

К достоинствам методологии можно отнести систему организации труда, быстрые сроки разработки, тесную и регламентированную работу с заказчиком. К недостаткам –

отсутствие в фазах создания проекта определения общей модели, поэтому, реализовав часть системы, может наступить момент, когда потребуется изменение большой части созданного для реализации оставшегося функционала.

Feature driven development

В разработке, управляемой функциональностью (FDD) [6] выделяют ряд процессов:

- *Разработка общей модели.* Проводится анализ круга решаемых задач, описание системы, ее внешнего окружения, ее составных частей, их анализ и взаимодействие. Анализ выполняется группами, затем обсуждается коллективно, и проводится экспертиза результата. Получаемая в итоге модель принимается за основу, которая может быть изменена в ходе работы

- *Составление списка необходимых функций системы.* Этот список составляется на предыдущем этапе. Исходная область декомпозируется на предметные области, описываются бизнес-процессы, происходящие в них. В результате составляется список функций (свойств), представленных в виде (действие; результат; объект). Функция либо является свойством, то есть функцией, реализуемой за две недели, либо разбивается на несколько свойств.

- *Планирование работ над каждой функцией.* Выполняется распределение владения классами и организация свойств в группы среди ведущих программистов.

- *Проектирование функций.* Ведущие программисты и разработчики составляют диаграммы последовательности для каждого свойства, выбранного на текущие две недели. Создаются образы классов и методов, происходит рассмотрение дизайна.

- *Реализация функций*, их тестирование и включение в основной проект.

Для контроля процесса разработки проекта необходимо протоколировать разработку каждого свойства. FDD выделяет шесть последовательных этапов для каждой функции (свойства). Первые три (анализ области, дизайн, проверка дизайна) полностью завершаются в процессе проектирования, последние три (код, проверка кода, включение в сборку) – в процессе реализации. На каждом этапе показывается процент его выполнения.

FDD построен путем синтеза наиболее известных и успешных практик: объектное моделирование области, разработка по функции, индивидуальное владение классом (кодом), команда по разработке функций (свойств), проверка кода, конфигурационное управление, регулярная сборка, обозримость хода работ и результатов.

Рассматриваемый метод вобрал лучшие практики в области современный программной

инженерии. Он позволяет решать крупные задачи за счет наличия процесса разработки модели, в то же время относится к технологии Agile, неся ее преимущества – быстрая разработка и итерационность. К недостаткам можно отнести больший объем документации, чем в других Agile методологиях, команда разработчиков в данном методе больше, взаимодействие ее членов менее интенсивное, и индивидуальное владение кодом в отличие от коллективного владения кодом в большинстве Agile методик.

Заключение

Сравнивая описанные методологии и оценивая их применимость к процессу разработки информационной системы вуза, можно сказать следующее. Для успешной работы Agile-методов должно быть организовано взаимодействие проектной группы, заказчика и менеджмента – это одна из составляющих успеха, но не всегда к такой работе готовы пользователи и руководство, в этом один из недостатков Agile-методологий. Гибкие методы позволяют быстро создать первую версию продукта, имеющего законченный функционал и обладающий самодостаточностью, которую можно предоставить заказчику для тестирования. Однако не всегда решение большой задачи сводится к решению набора малых. Так, можно потерять общие свойства системы.

Поэтому при разработке проекта ИС вуза выбрана стратегия на комбинированное использование нескольких методик. Это должна быть Agile методология, то есть должна быть итерационной, инкрементной, поддерживающей тесную связь с заказчиком. Наиболее применимой из рассмотренных является FDD методология с добавлением в процессе проектирования и реализации практик из других методов (XP, Scrum).

Список литературы

1. Гибкие технологии [Электронный ресурс] // ru.wikipedia.org: информ.-справочный портал URL:http://ru.wikipedia.org/wiki/Agile

2. Попов, А. В. Объектно-ориентированный анализ, проектирование и программирование информационной системы университета / А. В. Попов, А. Л. Григорьева, А. Ю. Лошманов // Современные проблемы науки и образования (эл. журнал). – 2012. – № 6.

3. Пульс профессии руководителя проекта от PMI / гибкие технологии управления бизнес-процессами [Электронный ресурс]. – Режим доступа:

http://gibtech.ru/blog/discus?entry_id=131(дата обращения 06.12.2012).

4. Agile-манифест разработки программного обеспечения [Электронный ресурс]. – Режим доступа: http://agilemanifesto.org/iso/ru/(дата обращения 06.12.2012).

5. DSDM [Электронный ресурс] // ru.wikipedia.org: информ.-справочный портал URL: http://ru.wikipedia.org/wiki/DSDM (дата обращения 06.12.2012).

6. FDD [Электронный ресурс] // ru.wikipedia.org: информ.-справочный портал URL:http://ru.wikipedia.org/wiki/Feature_driven_development (дата обращения 06.12.2012).

7. MSF [Электронный ресурс] // ru.wikipedia.org: информ.-справочный портал URL: ru.wikipedia.org/ wiki/Microsoft_Solutions_Framework (дата обращения 06.12.2012).

8. Scrum [Электронный ресурс] // ru.wikipedia.org: информ.-справочный портал URL:http://ru.wikipedia.org/wiki/Scrum (дата обращения 06.12.2012).

9. XP [Электронный ресурс] // ru.wikipedia.org: информ.-справочный портал URL: http://ru.wikipedia.org/wiki/ ExtremeProgramming (дата обращения 06.12.2012).

Рецензенты:

Биленко Сергей Владимирович, доктор технических наук, доцент, помощник ректора по информатизации учебного процесса ФГБОУ ВПО «Комсомольский-на-Амуре государственный технический университет», г. Комсомольск-на-Амуре.

Одиноков Валерий Иванович, доктор технических наук, профессор, заведующий кафедрой «Прикладная математика и информатика» ФГБОУ ВПО «Комсомольский-на-Амуре государственный технический университет» , г. Комсомольск-на-Амуре.