УДК 004

ОБЗОР МЕТОДОЛОГИЙ РАЗРАБОТКИ КОРПОРАТИВНЫХ ИНФОРМАЦИОННЫХ СИСТЕМ

Мутигуллин А.С., Прасолова Е.А.

ФГБОУ ВО «Магнитогорский технический университет им. Г.И. Носова», Магнитогорск, e-mail: mutigullin.art9@yandex.ru

В статье приводится подробный обзор наиболее популярных методологий разработки корпоративных информационных систем. Выбранная тема считается актуальной на сегодняшний день, так как в последнее время интерес к корпоративным информационным системам постоянно растет. Если вчера они притягивали внимание узкого круга руководителей, то сейчас проблемы системной автоматизации разных видов деятельности крупных коммерческих и некоммерческих организаций стали актуальными практически для всех. Охарактеризовано это не только положительным развитием экономики, но и тем, что к настоящему времени предприятия обладают значительным опытом использования программных продуктов различного класса. В статье приводится описание тяжелых и гибких методологий, в результате сравнительного анализа формируется вывод относительно того, что при разработке КИС целесообразно отдавать предпочтение тяжелым методологиям, поскольку в них четко прописаны основные этапы разработки программного обеспечения, а также предусматривается разработка большого количества проектной документации. Гибкие методологии, которые по своей природе более быстрые в силу меньшей формализации процессов разработки программных средств, при этом могут использоваться для настройки и адаптации отдельных модулей КИС.

Ключевые слова: корпоративная информационная система, методология, разработка, гибкие методологии, строгие методологии

REVIEW OF METHODOLOGIES OF DEVELOPMENT OF CORPORATE INFORMATION SYSTEMS

Mutigullin A.S., Prasolova E.A.

Nosov Magnitogorsk State Technical University, Magnitogorsk, e-mail: mutigullin.art9@yandex.ru

The detailed review of the most popular methodologies of development of corporate information systems is provided in article. The chosen subject is considered relevant today as recently interest in corporate information systems constantly grows. If yesterday they attracted attention of a narrow circle of heads, then now a problem of system automation of different types of activity of large commercial and non-profit organizations, became relevant practically for all. It is characterized not only by positive development of economy, but also the fact that so far the enterprises have considerable experience of use of software products of various class. The description of heavy and flexible methodologies is provided in article, as a result of comparative analysis a conclusion concerning the fact that when developing CIS it is expedient to give preference to heavy methodologies as the main development stages of the software are accurately registered in them is formed and also development of a large number of the project documentation is provided. Flexible methodologies which by the nature faster owing to smaller formalization of processes of development of software, at the same time can be used for control and adaptation of the CIS separate modules.

Keywords: corporate information system, methodology, development, flexible methodologies, rigorous methodologies

На современном этапе развития экономики Российской Федерации отмечается укрупнение коммерческих организаций, рост предприятий крупного бизнеса, автоматизация которых требует разработки сложных корпоративных информационных систем (КИС). КИС – это большие сложные программные системы, состоящие из многих взаимодействующих компонент и обеспечивающие основные (производственные) и вспомогательные бизнес-процессы [1–3]. При разработке такого рода систем возникает целый ряд существенных проблем, связанных с высокой их сложностью [4, 5]. Это, прежде всего, ограниченность ресурсов и часто изменяющиеся требования. У заказчика тоже достаточно много ролей, каждая из которых выдвигает свои требования, смотрит на процесс разработки корпоративных программных систем со своей точки зрения, имеет собственное мнение, приоритеты и ожидания. Остроту этих проблем призваны сгладить методологии разработки программного обеспечения, которые превращают создание программного продукта в упорядоченный процесс, с помощью которого можно сделать работу программиста более прогнозируемой и эффективной. Под методологией разработки КИМ будем понимать «Методология – подход к созданию и сопровождению информационных систем в виде жизненного цикла ИС, представляющий его в виде последовательности стадий, каждая из которых разбита на этапы, и выполняемых на них процессов [6].

Целью исследования является обзор современных методологий разработки про-

граммного обеспечения, а также определение области их применения в процессе разработки КИС. Предполагается рассмотреть как крупные методологии, предназначенные специально для создания корпоративных систем, так и методологии, которые работают в тех условиях, когда крупные методологии оказываются слишком громоздкими. К числу первых методологий мы отнесем Rational Unified Process (RUP) и Microsoft Solutions Framework (MSF). К числу гибких методологий относится все, что касается гибких (Agile) – методологий: это популярная в настоящее время методология Scrum, eXtreme Programming (XP) и Agile [7].

В исследовании использовались преимущественно теоретические методы: изучение литературы по проблеме исследования, сравнительный анализ методологий, обобщение. В качестве материалов исследования выступили научные статьи и книги, посвященные проблеме построения корпоративных информационных систем.

В результате исследования было выявлено, что MSF и RUP характеризуются достаточно жесткими процессами, большим количеством артефактов, большим количеством документации, которая создается на каждом этапе, а также достаточно сложным командным взаимодействием, в ряде случаев это масштабирование команды команд. «Тяжелые» методологии достаточно хорошо подходят для создания КИС, поскольку они продуцируют большое количество артефактов. Мелкие методологии, гибкие, подходят значительно меньше и только в ряде случаев.

В основе методологии RUP лежат процессы, причем следует отметить такие особенности методологии, такие особенности подхода, как архитектурную центричность, а также основу на Use Cases – сценариях использования, и итеративность.

Существуют фазы разработки – четыре крупных периода (inception, elaboration, construction и transaction, или начало, исследование, конструирование и передача). Внутри каждой из этих фаз может существовать некоторое количество итераций по инкрементальной подготовке, последовательному доведению решений и артефактов до того вида, когда они могут быть приняты, в том числе и заказчиками.

При этом на первом этапе дела есть высокоуровневые требования и общая концепция программного продукта, но еще нет детальных спецификаций. На втором этапе происходит архитектурное проектирование. На третьей стадии – конструирование или разработка, где и происходит кодирование, тестирование и сборка, разработка всей необходимой документации, для того чтобы релиз в начальном варианте мог быть передан заказчику. На четвертой стадии происходит последовательное уточнение, доработка, переработка, сведение и фиксация всей необходимой документации и кода, для того чтобы продукт был полнофункциональным и был готов к передаче заказчику. Так происходит базовый процесс разработки. Поскольку RUP – методология строгая, существуют вполне определенные критерии выхода, как из каждой фазы, так и из каждой итерации. Существуют вполне определенные артефакты и вполне определенные метрики, которые описывают степень их готовности. На первой стадии это основные высокоуровневые требования к системе. На втором этапе – архитектурный проект и, соответственно, диаграммы, которые описывают программный продукт, если на первом этапе это были Use-Case. На третьем этапе – программное решение в виде кода и документации. И на четвертом этапе – полный релиз, готовый к передаче заказчику [1, 7, 8].

Методология MSF основана на гибкой процессной модели и включает в себя командную разработку. Масштабирование, команды команд – это достаточно важные составляющие MSF. Нужно сказать, что MSF поддерживает полный жизненный цикл разработки, т.е. он включает в себя как MSF (Microsoft Solutions Framework), так и MOF (Microsoft Operations Framework), которые объединяют процессы концептуализации, создания, внедрения, сопровождения, расширения, развития программных продуктов.

Прежде всего, важнейшим фокусом этой методологии является ориентация именно на бизнес, требования заказчика. Поэтому первое, что требуется – это партнерство с клиентом. При этом клиент понимается достаточно широко. Это может быть не обязательно конечный заказчик, но это может быть целый ряд людей, которые называются стейкхолдерами, или людей, которые вносят свой капитал в создание и развитие программного продукта. Одной из ценностей методологии является открытая коммуникация. Суть её состоит в том, что на самом деле представление о продукте, особенно на уровне первоначальной идеи, первоначальной концепции с точки зрения разработчика и с точки зрения заказчика, могут весьма существенно отличаться. При этом на стороне разработчика существует порядка 15 и даже более ролей, каждая из которых на самом деле имеет свое видение и свой взгляд на продукт и на программный

проект. Поэтому для того, чтобы разработка продукта, необходимого заказчику, была предсказуема и надежна, нужна открытая коммуникация, нужно постоянное взаимодействие и нужно строить общее видение – vision. Совместное видение – это, таким образом, третий принцип. Четвертый принцип – качество как работа каждого, как ежедневная необходимость создавать некую ценность, т.е. документацию, программный код и т.д., которые будут положены в основу будущего продукта. Еще один важный принцип – быть адаптивным и приспосабливаться к изменениям, за счет чего происходит постоянный мониторинг рисков, общение с заказчиком, и, таким образом, создается ценность, а внедрение делается привычкой. Microsoft основывает свою методологию не просто на разработке продукта и передаче заказчику, но и на внедрении, доводке и сопровождении [9–11].

Сравнительная характеристика строгих методологий разработки корпоративных систем представлена в табл. 1.

Гибкая методология разработки *Scrum* предназначена для адаптивного управления команды разработчиков и создания продуктов в условиях высоких рисков и неопределенностей. Её особенность заключается, прежде всего, в обеспечении адаптивности, в фокусе на команду, управлении командами не только разработки, но и поддержки. Подход Scrum может быть использован для масштабируемых команд, для управления достаточно большими проектами. В этом случае он называется Scrum of Scrums [12].

Scrum, как и любая методология, включает набор практических приемов, методов, средств, технологий для решения задач проектирования и реализации программного обеспечения в адаптивном формате и при наличии гибкой, самоорганизующейся команды. Методология Scrum, как и любая другая гибкая методология, является во многом достаточно неформальной. Команда постоянно находится во взаимодействии, отслеживает свой производственный процесс в ходе кратких устных совещаний, которые не приводят к созданию большого количества документов. Из всего этого можно сделать вывод, что методология Scrum, как и методологии гибкой разработки в целом, не вполне хорошо предназначены для создания больших корпоративных систем с четкой документацией. Но тем не менее в условиях существенных проектных рисков, в условиях больших неопределенностях, в условиях кризиса, в условиях жесткой экономии по срокам и стоимости эти методологии могут оказаться приемлемыми.

К основным принципам *экстремального программирования* как подхода к гибкой разработке программных систем относится, прежде всего, обратная связь. Эта обратная связь поступает как от заказчика, так и от системы, она может поступать и от других разработчиков. Здесь критично время. Если обратная связь, если реакция заказчика получена слишком поздно, разработчики не могут провести грамотное ранжирование требований и начинают отставать от графика разработки [13], поэтому обратная связь хороша вовремя. Контакт с заказчиком должен быть непрерывным, поддерживаться постоянно и даже при том, что итерации, т.е. фрагменты времени, которые необходимы для разработки некоего нового дополнения к программной системе, зачастую возникают еженедельно, разработчики должны поддерживать связь с заказчиком практически непрерывно.

Методология XP подразумевает простоту. Речь идет о том, что программа или ее фрагмент должны быть изначально разработаны настолько просто, включать настолько минимальное количество строк и настолько предельно краткое описание того, что должно быть сделано, в том числе и в комментариях, чтобы можно было с одного взгляда понять, каким образом этот код устроен, даже если какой-то конкретный разработчик в создании его участия и не принимал. Наконец, инкрементное изменение. Это очень важный пункт. Разработка ведется последовательным добавлением новой функциональности.

Таблица 1

Сравнительная характеристика строгих методологий разработки корпоративных систем

RUP	MSF
Подходит для больших и очень больших проектов	Подходит для больших и очень больших проектов
Поддерживает разные модели ЖЦ	Гибкая и масштабируемая методология, построена на итеративной модели разработки
Базируется на широком использовании UML	Важный аспект подхода – синхронизация и стабилизация
На всех стадиях используются программные метрики	Четко определяются результаты по каждой контрольной точки

Таблица 2

Сравнительная характеристика гибких методологий разработки корпоративных систем

Методологии	Преимущества	Недостатки
Scrum	Простые практики и артефакты менеджмента	Минимальное руководство во всех дисциплинах кроме менеджмента
	Самоуправляемая команда, решающая свои проблемы	
	Эволюционный подход к требованиям и разработке, адаптивное поведение	
	Активное участие клиента	
	Сосредоточенность, прозрачность	Нет четко определенных проектных документов
	Легко сочетается с другими методами	
	Коммуникация, взаимное обучение, общее построение ценностей	
	Укрепление команды на ежедневных митингах	
EP	Много полезных и легко усваиваемых техник	Требуется заказчик в команде
	Участие заказчика	Зависимость от устного общения
	Эволюционный подход	Сильная связанность XP практик, они эффективны только в совокупности
	Кодировщики оценивают трудозатраты, расписание следует этим оценкам	Отсутствие стандартного способа описания дизайна
	Важность взаимодействия	Некоторые разработчики не любят парного программирования
	Важность качества	Отсутствие определенных артефактов
	Уточнение требований к системе на основе приемочных тестов от заказчика	Простой дизайн может приводить к отсутствию архитектурной проработки
	Четкие метрики	
	Частые ревизии	
Agile	Гибко отслеживать часто меняющиеся требования к ПО без выхода за рамки бюджета проекта	Может привести к низкому качеству продукта
	Максимально эффективно вносить изменения и новые идеи заказчика в разрабатываемый продукт	Риск никогда не достигнуть завершения проекта
	В короткие сроки внедрить работающий функционал и оптимизировать его в процессе использования	
	Создать поле для постоянного взаимодействия команды разработки и пользователей	Могут возникнуть проблемы с расширяемостью продукта
	Обеспечить постоянный контроль за разработкой и внедрением со стороны заказчика	

Agile несет в себе несколько принципов. Это итерации фиксированной длины – короткая и быстрая эволюционная и итерационная разработка ограничивает по времени все, что только возможно. Два уровня планирования, более крупные релизы и более мелкие итерации. Жесткое соблюдение плана итераций, уточнение требований по мере необходимости, частое и раннее тестирование, т.е. тесты до кода или вместе с кодом. Заказчик работает в команде проекта, происходит непрерывное обучение и адаптация. Это очень гибкий и адаптивный подход, подразумевающий достаточно небольшое количество стандартных артефактов и метрик.

Процессы в Agile двухуровневые, итеративные, адаптивные. Под них можно подогнать достаточно большое количество процессов и достаточно широкое количество, широкий спектр процессов определяется этим подходом. Количество итераций заранее неизвестно, поэтому работать, скажем, в концепции fixed price, когда представляется целесообразным или возможным указать некую сумму бюджета, в которую мы совершенно точно уложимся, в рамках Agile зачастую очень сложно.

Процессы в Agile в определенной мере гибкие, настраиваемые и очень сильно зависят от того, какого рода команда ведет разработку, какие приоритеты у этой команды, какие ценности она перед собой ставит. Простые практики и инструменты. Все остальное можно уточнить, улучшить, усовершенствовать посредством рефакто-

ринга, если будет признано, в том числе и заказчиком, что действительно эта функциональность необходима. Эмпирический процесс, а не предопределенный; его во многом формируют и заказчик, и разработчик, команда разработчиков. Эволюционная и инкрементная поставка, частые и быстрые релизы, команда как сложная адаптивная система, в которую входят менторы, наставники, тренеры, консультанты и другие роли, которые способствуют формированию команды, сплочению команды и работе этой команды как единого адаптивного организма. Ограниченные во времени итерации, при этом никаких изменений внутри итераций не может быть. Инкрементная и эволюционная поставка, постоянное взаимодействие с заказчиком – вот такие приоритеты у этой методологии [13, 7].

Сравнительная характеристика гибких методологий разработки корпоративных систем приводится в табл. 2.

В заключение стоит отметить, что залогом успеха любого проекта, в том числе и корпоративного, является не только сплоченность команды, но и строгое следование стандартам и использование специализированных инструментальных средств. Не существует универсальной методологии, которая сможет решить все проблемы любого заказчика раз и навсегда. Выбор методологии существенным образом определяется характером и масштабом проекта, теми задачами, которые ставит заказчик. Строгие, «тяжелые», методологии пригодны для разработки крупных КИС, в то время как гибкие методологии можно использовать для отдельных модулей КИС, при наличии каких-то сложных обстоятельств: кризисные условия, быстрая смена требований и приоритетов заказчиком, высокие риски проекта. Предполагается, что материалы исследования послужат основанием для дальнейшей работы по выбору методологии проектирования и разработки КИС сети магазинов розничной торговли.

Список литературы

1. Гаврилова И.В. Подходы к проектированию архитектуры корпоративной информационной системы негосударственного пенсионного фонда // Инновации, качество и сервис в технике и технологиях: сборник научных трудов 4-ой Международной научно-практической конференции Курск: Юго-Зап. гос. ун-т, 2015. С. 267–268.

2. Гаврилова И.В. Теоретические аспекты развития корпоративных информационных систем // Перспективное развитие науки, техники и технологий: материалы 3-й Междунар. научно-практической конференции. Курск: Юго-Зап. гос. ун-т, 2013. С. 267–268.

3. Чернова Е.В., Гаврилова И.В., Доколин А.С., Романова М.В. Информационные технологии в управлении непрерывностью бизнеса // Научное обозрение. Экономические науки. 2017. № 5. С. 46–50.

4. Ананьин В.И. Формирование архитектуры корпоративной информационной системы путем естественного отбора // Intelligent Enterprise. 2006. № 17 [Электронный ресурс]. URL: http://www. iemag.ru/articles/detail.php?ID=5232 (дата обращения: 11.12.2018).

5. Гаврилова И.В. Свободное программное обеспечение для управления бизнес-процессами // Теория и практика применения свободного программного обеспечения: сборник трудов участников Всероссийской молодежной конференции с элементами научной школы. Магнитогорск, 2011. С. 144–147.

6. Гаврилова И.В. Разработка приложений : учеб. пособие. М.: ФЛИНТА, 2012. 242 с.

7. Олейник П.П. Корпоративные информационные системы: учебник для вузов. Стандарт третьего поколения. СПб.: Питер, 2012. 176 с.

8. Попова И.В., Субочев А.В. Разработка обучаемой специализированной информационно-поисковой системы // Программные продукты и системы. 2011. № 3. С. 22.

9. Трофимов В.В., Ильина О.П., Трофимова Е.В., Кияев В.И., Приходченко А.П. Информационные системы и технологии в экономике и управлении: учебник для академического бакалавриата / Под ред. В.В. Трофимова. 4-е изд., перераб. и доп. М.: Издательство Юрайт, 2016. 542 с.

10. Нетесова О.Ю. Информационные системы и технологии в экономике: учеб. пособие для вузов, 3-е изд., испр. и доп. М.: Издательство Юрайт, 2017. 146 с.

11. Попова И.В., Зленко И.В., Попова Е.В. Информационная система «Наука в МаГУ»: опыт разработки // Теория и практика применения свободного программного обеспечения: сборник трудов участников Всероссийской молодежной конференции с элементами научной школы. Магнитогорск, 2011. С. 27–32.

12. Рыжко А.Л., Рыбников А.И., Рыжко И.А. Информационные системы управления производственной компанией: учебник для академического бакалавриата. М.: Издательство Юрайт, 2017. 354 с.

13. Гаврилова И.В. Подходы к интеграции автоматизированных информационных систем ФГБОУ ВПО «МаГУ» // Инновации, качество и сервис в технике и технологиях: сборник научных трудов 4-ой Международной научно-практической конференции. 2014. С. 49–52.