BULLETIN OF NATIONAL ACADEMY OF SCIENCES
OF THE REPUBLIC OF KAZAKHSTAN
ISSN 1991-3494
Volume 2, Number 354 (2015), 172 – 182

FEATURES SOME OF THE EXISTING IT MANAGEMENT METHODOLOGY IT-PROJECT

A. S. Zhumakhanova

S. Seifullin Kazakh agrotechnikal university, Kazakhstan.
E-mail: anarzh@mail.ru

Keywords: IT-project, information technology, program product, MSF, PERT, Agile, PMBoK, Oracle model

Abstract. IT-projects are special kind of innovative projects and has specific features. This article investigated the key features of the implementation of IT-projects and generated recommendations for effective management. Today, there are project management methodology as PMBOK, SCRUM, Agile, PRINCE2 and others. However, application of the principles recorded in these standards are not fully solve the problem of effective project management in the field of information technology. For more of their adaptation to the reality and practice of IT-companies.

УДК 004.4

ОСОБЕННОСТИ НЕКОТОРЫХ СУЩЕСТВУЮЩИХ МЕТОДОЛОГИИ УПРАВЛЕНИЯ ИТ-ПРОЕКТАМИ

А. С. Жумаханова

Казахский агротехнический университет им. С. Сейфуллина, Казахстан

Ключевые слова: ИТ-проект, информационные технологий, программный продукт, модель, MSF, PERT, Agile, PMBoK, Oracle.

Аннотация. ИТ-проекты являются особенной разновидностью инновационных проектов и имеются специфические черты. В данной статьи исследованы ключевые особенности осуществления ИТ-проектов и сформированы рекомендации для эффективного управления ими. На сегодняшний день существуют методологии управления проектами как PMBOK, SCRUM, Agile, PRINCE2 и другие. Но, применение принципов записанных в этих стандартах полностью не решает проблемы эффективного управления проектами в области информационных технологий. Необходима дополнительная их адаптация к действительности и практике ИТ-компании.

Современное состояние бизнеса требует от создателей программного обеспечения (ПО) разработки программных продуктов высокого качества в рамках отведенного бюджета и в срок. В создании программных продуктов, как правило, принимают участие различные специалисты, которые объединяются в команды. Команды могут включать сотрудников организации разработчика и заказчика, привлекаемых временных специалистов и субподрядчиков. Члены команды разработчиков ПО могут территориально находиться в одном или разных местах (распределенная разработка). Эффективное решение задач создания качественного ПО предполагает использование инструментальных средств, методик и технологий управления процессами жизненного цикла программных систем: формирования требований, моделирования и проектирования, разработки, тестирования, построения и развертывания систем.

Рациональная организация процессов разработки программных систем описывается в стандартах (международных, государственных, корпоративных), которые часто называют методо-

логиями разработки ПО. Методологии создания ПО обычно разрабатываются ведущими производителями программных систем и их сообществами с учетом особенностей программных продуктов, а также сферы внедрения. Методологии описывают подходы к организации рациональной стратегии и возможному набору процессов создания ПО.

В настоящее время все большее распространение получают гибкие методологии разработки программного обеспечения, где основное внимание сосредоточено на создании качественного продукта, а не подготовку исчерпывающей документации по проекту. При этом акцент делается на организацию эффективного управления командой. Как отмечает Эрих Гамма: «… ключ к своевременной поставке продукта - не процессы, а люди». Самоорганизация и целеустремленность команды разработчиков позволяет создавать высококачественные программные продукты в сжатые сроки.

Инструменты управления жизненным циклом приложений во многом способствуют успешности программных проектов. Компания Microsoft предоставляет разработчикам гибкий инструментарий для управления жизненным циклом приложений – ALM Visual Studio и Team Foundation Server. Традиционные средства разработки программ в Visual Studio дополнены средствами архитектурного проектирования и тестирования.

Цель данной исследовательской работы – представить основные положения командной разработки программного обеспечения, управления жизненным циклом приложений, гибкой методологии создания программных систем, а также возможностей инструментария VisualStudio2013 и TeamFoundationServe для управления жизненным циклом приложений.

MICROSOFT SOLUTIONS FRAMEWORK – методология разработки программного обеспечения, предложенная корпорацией Microsoft. MSF опирается на практический опыт Microsoft и описывает управление людьми и рабочими процессами в процессе разработки решения.

Базовые концепции и принципы модели процессов MSF:

- единое видение проекта – все заинтересованные лица и просто участники проекта должны чётко представлять конечный результат, всем должна быть понятна цель проекта;
- управление компромиссами – поиск компромиссов между ресурсами проекта, календарным графиком и реализуемыми возможностями;
- гибкость – готовность к изменяющимся проектным условиям;
- концентрация на бизнес-приоритетах – сосредоточенность на той отдаче и выгоде, которую ожидает получить потребитель решения;
- поощрение свободного общения внутри проекта;
- создание базовых версии – фиксация состояния любого проектного артефакта, в том числе программного кода, плана проекта, руководства пользователя, настройки серверов и последующее эффективное управление изменениями, аналитика проекта.

MSF предлагает проверенные методики для планирования, проектирования, разработки и внедрения успешных IT-решений. Благодаря своей гибкости, масштабируемости и отсутствию жестких инструкций MSF способен удовлетворить нужды организации или проектной группы любого размера. Методология MSF состоит из принципов, моделей и дисциплин по управлению персоналом, процессами, технологическими элементами и связанными со всеми этими факторами вопросами, характерными для большинства проектов.

Visual Studio Team System решаются следующие задачи:

- повышение предсказуемости успеха проекта.
- рост производительности труда команды разработчиков за счет понижения сложности процессов проектирования и реализации современных сервисно-ориентированных решений.
- обеспечение сотрудничества команды путем интеграции коммуникационных и прочих средств, используемых ее членами.
- расширение возможностей командной работы за счет обеспечения удаленным пользователям доступа к надежному, защищенному и масштабируемому окружению.
- предоставление команде разработчиков гибкого и расширяемого коммуникационного решения с настраиваемыми сервисами.

Ключевыми членами проектной команды являются: руководитель проекта, архитектор, разработчик и тестировщик. Каждую из этих ролей может исполнять как один человек, так и несколько.

Второстепенные участники проекта, являющиеся профессионалами в области информационных технологии, также по достоинству оценят использование Team System, так как эта система облегчит им взаимодействие с остальными членами команды.

Модели MSF:

- модель процессов MSF;
- модель проектной группы MSF;
- дисциплина управления проектами MSF;
- дисциплина управления рисками MSF;
- дисциплина управления подготовкой MSF.

Научному руководителю проектов, выполняемых в составе команды, необходимо также решать следующие задачи:

- психологическая совместимость;
- методы подбора групп;
- сроки введения группового проектирования, первоначальная подготовка;
- распределение ролей, ротация ролей.

Управление изменениями проекта по разработке программного обеспечения ориентировано на анализ влияния изменений свойств и функций конечного программного обеспечения в процессе реализации проекта. Управление изменениями тесно связано с управлением требованиями, так как бизнес-аналитики и разработчики программного обеспечения, выявив изменения в потребностях и требованиях заказчика, способны перестроить или улучшить дальнейшую реализацию проекта. Однако каждое изменение или нововведение способно так или иначе повлиять на сроки проекта или его бюджет, поэтому очень важно провести предварительную оценку рисков [1].

Agile направление в MSF ориентируется на небольшие команды (5-6 человек), предполагает, что информация о разрабатываемом продукте не просто выясняется в процессе разработки, а может и будет изменяться по ходу. Таким образом, первая рабочая версия системы должна быть создана как можно раньше, а сам продукт фактически проявляется из прототипов путем повторения итераций в цикле разработки.

Методология MSF содержит весьма много элементов, в частности:

- рекомендованные процессы создания IT-проектов;
- структуру итераций;
- роли членов команды;
- шаблоны документов (Excel, Word);
- шаблоны Microsoft Project;
- отчеты;
- портал проекта (шаблон сайта SharePoint).

MSF for Agile Software Development ориентирован на использование итеративной и эволюционной модели процесса разработки и основан на сценариях использования. MSF for Agile Software Development выделяет 7 ролевых групп (рисунок 1) [2]:

- Управление программой (program management)
- Архитектура продукта (architecture)
- Разработка (development)
- Тестирование (test)
- Управление выпуском (release operations)
- Удовлетворение потребителя (user experience)
- Управление продуктом (product management)

Ролевые группы и 6 ролей (рисунок 2):

- менеджер проекта (project manager) – ролевая группа Управление программой
- архитектор (archrect) – ролевая группа Архитектура
- разработчик (developer) – ролевая группа Разработка
- тестер (tester) – ролевая группа Тестирование
- релиз-менеджер (release manager) – ролевая группа Управление выпуском
- бизнес-аналитик (business analyst) – ролевые группы Управление продуктом и Удовлетворение потребителя

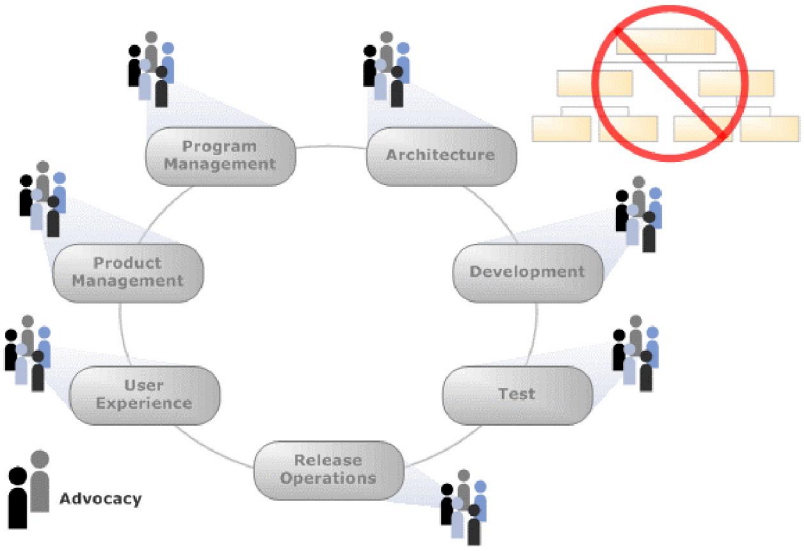

Рисунок 1 – Модель команды в MSF 4

Рисунок 2 – Модель команды в MSF 4.0 – роли

Для каждой ролевой группы, помимо зоны ответственности, определены заинтересованные стороны, как внутри, так и вне команды, с которыми группа должна взаимодействовать и чьи интересы представлять/отстаивать при принятии решений.

Качество программного продукта определяется по нескольким критериям и качественный программный продукт должен отвечать функциональным и нефункциональным требованиям. В жизненном цикле приложения качество должно отслеживаться на всех его этапах. Тестирование программного продукта позволяет на протяжении всего жизненного цикла ПО гарантировать, что программные проекты отвечают заданным параметрам качества. На протяжении всего жизненного цикла разработки ПО применяются различные типы тестирования. Инструментарием тестировщика в VisualStudio 2013 является MicrosoftTestManager идиспетчер виртуальной среды LabManagement. Для улучшения качества кода программных приложений применяют рефакторинг.

Рассмотрим, какие инструменты могут потребоваться команде разработчиков на различных этапах разработки приложения.

Определение требований. Определение требований – это задача бизнес-аналитиков, которые осуществляют предпроектное обследование, общаясь с заказчиком и потенциальными пользователями и выясняя их проблемы и потребности. Результатом обследования обычно является документ, называемый в нашей стране техническим заданием и содержащий сведения о назначении продукта, набор требований к нему и описание границ проекта [2].

Какие инструменты требуются бизнес-аналитику? Потребность в инструментах определяется масштабом проекта, требованиями заказчика к оформлению документации, стандартами, принятыми в той или иной компании-разработчике. Бизнес-аналитик может обойтись и текстовым процессором (например, Microsoft Word), изложив требования в виде текста. Однако в последнее время описание требований принято иллюстрировать UML- и IDEF0-диаграммами (рисунок 3), описывающими сценарии взаимодействия пользователя с продуктом, порядок передачи сообщений от одних объектов к другим, взаимодействие объектов друг с другом, потоки работ и изменение состояний объектов. Для создания таких диаграмм можно применять Microsoft Visio, Rational Rose, Rational XDE, Borland Together, для создания диаграмм в стандарте IDEF0 наиболее активно используется CA AllFusion Process Modeler (ранее носивший название BPwin) [3].

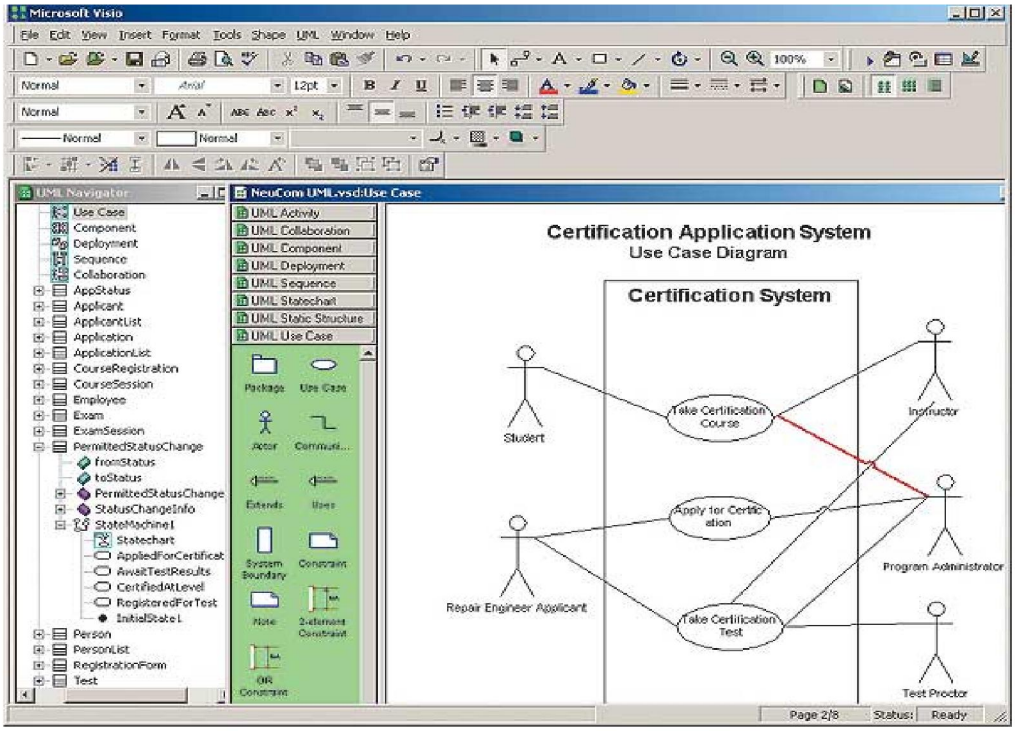

Рисунок 3 – Пример UML-модели (Microsoft Visio)

Какой именно инструмент подходит для данного проекта, во многом зависит от постановки процессов разработки. Если бизнес-аналитику нужно только проиллюстрировать проектную документацию, то выбор инструмента не принципиален – лишь бы можно было нарисовать с его помощью ту или иную диаграмму. Если же бизнес-аналитик должен не просто сделать иллюстрацию, а создать модель для последующего использования системными аналитиками, разработчиками и специалистами по тестированию на последующих этапах проекта, то выбор инструмента определяется его способностью к интеграции со средствами разработки, которые со значительной степенью вероятности будут использоваться в качестве инструментария реализации кода. Из наиболее технологически совершенных современных пар «средство разработки – средство моделирования» стоит отметить такие, как Visual Studio.NET – Rational XDE, Visual Studio. NET – Borland Together, Borland JBuilder – Borland Together.

Помимо текстовых редакторов и средств моделирования, бизнес-аналитики могут применять средства управления требованиями, позволяющие хранить структурированный и систематизи-

рованный список требований к продукту в какой-либо базе данных и обращаться к нему на последующих этапах, связывая требования с реализующими их составными частями продукта и тестами, проверяющими соответствие продукта требованиям, а также корректно отслеживать изменения в требованиях, которые возникают практически в каждом проекте, как бы тщательно ни проводилось исследование, и влияние этих изменений на результаты последующей работы. Из наиболее известных сегодня средств управления требованиями следует отметить RequisitePro (IBM/Rational), DOORS (Telelogic) и CaliberRM (Borland) [3].

Таким образом, на этапе определения требований нужны средства подготовки документов, а во многих случаях и средства управления требованиями, моделирования бизнес-процессов и UML-моделирования.

Проектирование. За этапом определения требований, завершающимся утверждением технического задания, следует этап проектирования.

Осуществляется он, как правило, архитекторами приложений, принимающими решения относительно архитектуры и составных частей создаваемого решения, а также технологий их реализации, и системными аналитиками, осуществляющими проектирование данных и классов приложения, а иногда и прототипирование пользовательских интерфейсов.

Результатом этого этапа обычно является документ, часто называемый техническим проектом и содержащий диаграммы классов, модели данных, прототипы пользовательских интерфейсов. Обычно на этом этапе к модели, созданной бизнес-аналитиками, добавляются диаграммы классов создаваемых приложений и диаграммы развертывания создаваемого решения, а также диаграммы, описывающие логическую модель данных и их физическую структуру для выбранной СУБД.

Для создания диаграмм классов и диаграмм развертывания используются вышеперечисленные инструменты UML-моделирования. Для проектирования данных обычно применяют такие инструменты, как CA AllFusion Data Modeler (бывший ERwin), Sybase Power Designer, Oracle Designer, а также аналогичные инструменты компаний Embarcadero и Popkin Software. Для СУБД, разработанных компанией Microsoft, можно достаточно успешно применять и Microsoft Visio. Перечисленные инструменты позволяют создать как минимум скрипт для генерации базы данных, а различия между ними заключаются в способах управления генерацией серверного кода, связанного с созданием триггеров и хранимых процедур (рисунок 4).

Прототипирование пользовательского интерфейса, если таковое на данном этапе производится, может потребовать применения либо средства рисования изображений форм будущего приложения (например, Microsoft Visio или графических редакторов), либо непосредственно средств разработки приложений.

Таким образом, на этапе проектирования нужны средства моделирования и проектирования данных и UML-моделирования, а в некоторых случаях – средства создания изображений форм.

Разработка продукта. На этапе собственно разработки создается код приложения в соответствии с техническим проектом, в том числе и серверный код, реализующий функциональность, отсутствующую в модели данных. На этом этапе основным инструментом, обязательным к применению, является средство разработки приложений. Выбор средства разработки определяется в первую очередь платформой (Windows, .NET, Java/J2EE, Linux/UNIX) и архитектурой (приложения с графическим интерфейсом, консольные приложения и службы, Web-приложения) и в настоящее время достаточно разнообразен (рисунок 5). Средства разработки Java/J2EE-приложений производят компании IBM, Oracle, Borland, средства разработки Windows-приложений – Microsoft, Borland, Sybase, средства разработки .NET-приложений – Microsoft и Borland, средства разработки приложений для Linux – Borland и некоторые другие компании.

Помимо средств разработки на этом этапе иногда требуются различные дополнительные библиотеки компонентов, предназначенные для применения на конкретной платформе или вместе с определенными средствами разработки. Такие компоненты поставляет большое количество независимых разработчиков, а также многие поставщики коммерческих продуктов, на основе которых предполагается создание решений другими разработчиками (примерами таких продуктов являются многие геоинформационные системы, CAD-приложения и некоторые серверные продукты).

Естественно, что при создании решений на базе какого-либо серверного продукта (СУБД, J2EE-сервера, сервера обмена сообщениями и т.д.) на данном этапе следует иметь этот продукт. Во

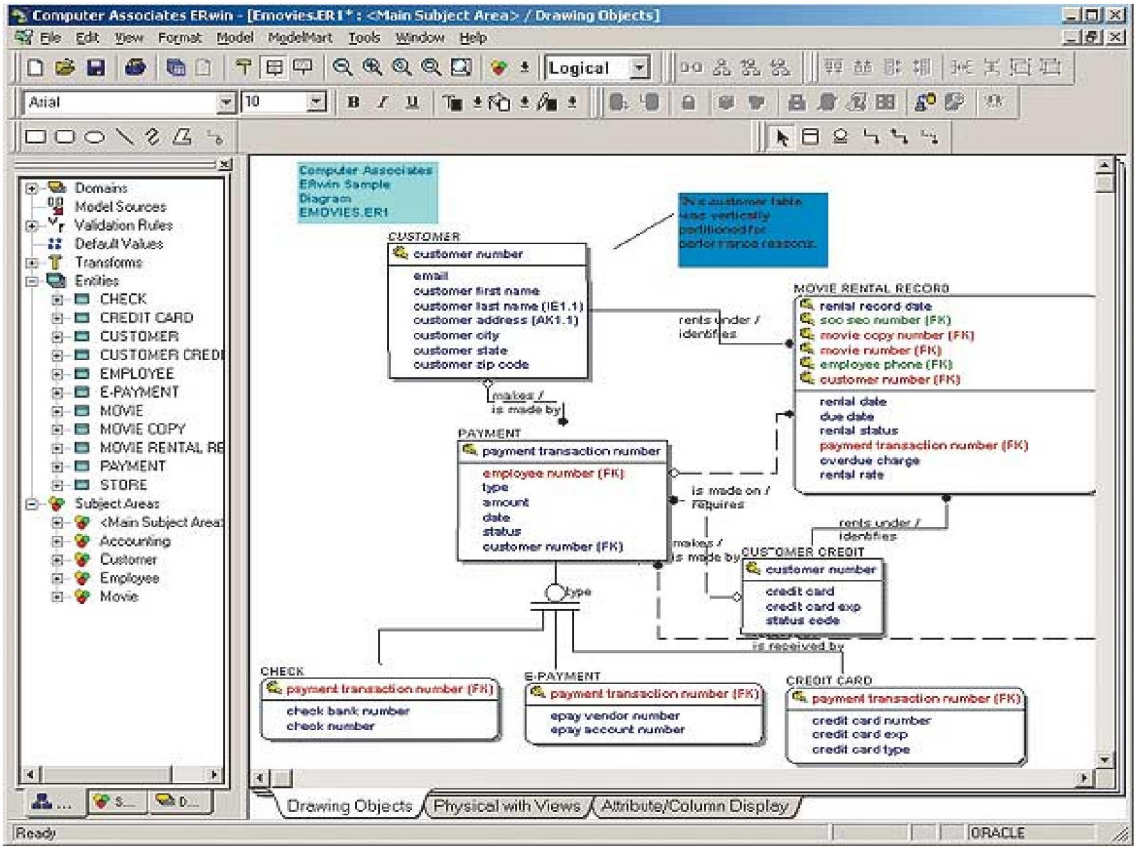

Рисунок 4 – Пример логической модели данных (CA AllFusion ERwin Data Modeler)

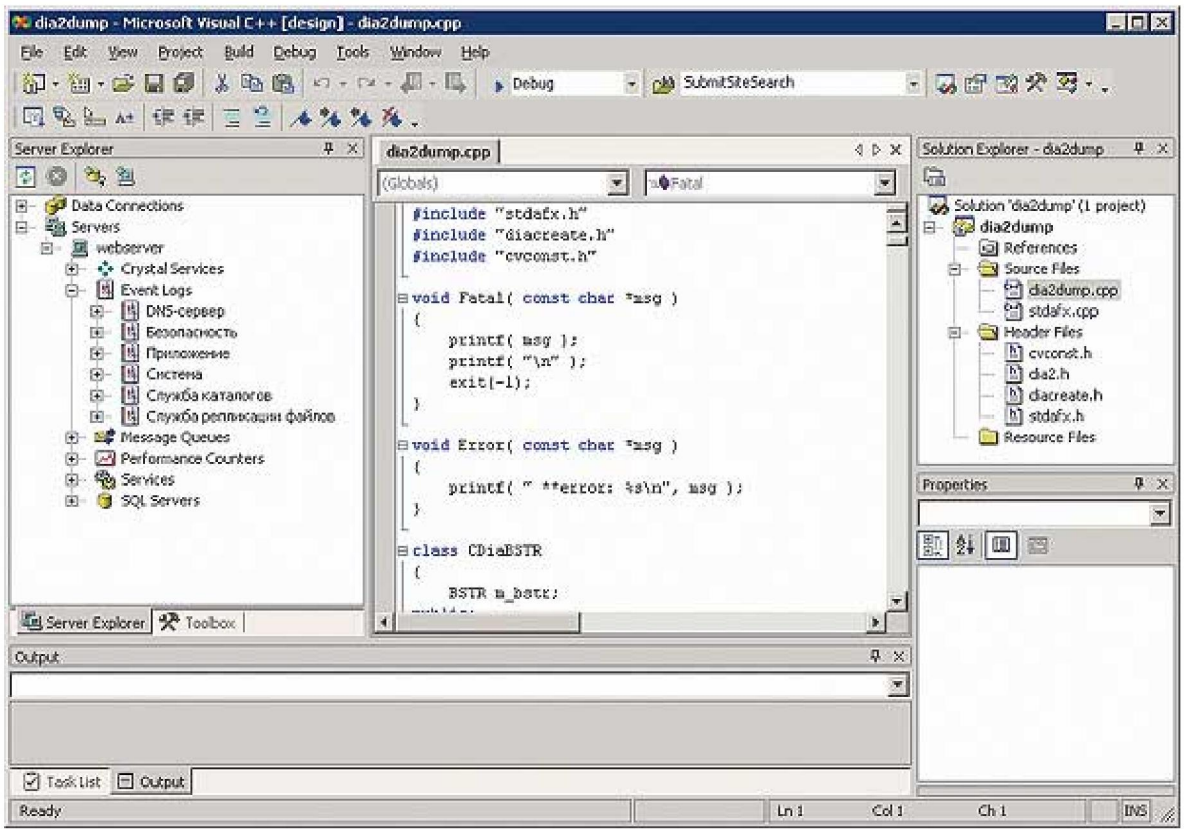

Рисунок 5 – Создание кода приложения (Microsoft Visual Studio.NET)

многих случаях специально для разработчиков решений производители таких продуктов выпускают версии, предназначенные для разработки и отладки приложений, но не для промышленной эксплуатации.

На этапе разработки приложения средства моделирования тоже применяются, особенно в том случае, когда они могут осуществлять не только генерацию кода на различных языках программирования, но и поддерживать обратное проектирование, создавая диаграмму классов на основе готового приложения либо позволяя синхронно редактировать и код, и модель. Функция синхронного изменения кода и модели существенно упрощает многие процессы, сопровождающие собственно разработку, так что если есть возможность выбора инструментов, то стоит обратить внимание на ее поддержку (например, синхронное изменение кода и модели поддерживают некоторые редакции инструмента UML-моделирования Borland Together).

Нередко на этапе создания приложений применяются средства оптимизации кода и его отладки. Такие инструменты могут входить в состав средств разработки или поставляться отдельно.

Тестирование и оценка качества. При тестировании продукта проверяется его соответствие требованиям, и в зависимости от этих требований осуществляется определение методик тестирования, создание тестов и выбор соответствующих инструментов.

Тестировщик (tester) выявляет и устраняет все неполадки в продукте и дает окончательное разрешение на его выпуск. Он также оценивает соответствие набора реализованных в продукте функций общей концепции и области действия проекта.

Тестирование должно включать в себя не только проверку кода. Тестировать надо функциональные спецификации, систему обеспечения производительности, пользовательские интерфейсы, планы внедрения и используемую терминологию. Тестер обеспечивает то, что все особенности и задачи будут известны до выпуска версии продукта, разрабатывает стратегию тестирования и планы тестирования для каждой из фаз проекта.

Планы и процедуры тестирования для клиент-серверных систем должны быть комплексными. Еще более комплексными они должны быть в случае событийно-ориентированного программирования, нескольких сетевых транспортов и целевых серверов, задач администрирования данных и баз данных и т.д.

Очень важно различать тестирование и контроль качества. Тестирование сосредоточено на проекте и оперирует деталями и техникой работы.

В современных проектах тестированию подвергаются и пользовательский интерфейс, и производительность приложений, и безопасность данных, и совместимость с различными операционными системами и приложениями. Для этого разработаны соответствующие инструменты, такие как утилиты тестирования баз данных, средства автоматического тестирования пользовательского интерфейса, средства нагрузочного тестирования, средства тестирования ошибок исполнения, средства тестирования безопасности приложений.

Как правило, подобные инструменты содержат средства создания сценариев, согласно которым производится автоматизированное тестирование, то есть многократное выполнение набора действий с одновременным сбором статистики отказов, времени выполнения операций и иных сведений, связанных с оценкой качества продукта.

Основными производителями средств тестирования в настоящее время являются компании Compuware, Segue, Mercury Interactive, IBM/Rational, Borland. Каждая из них производит несколько инструментов автоматизированного тестирования, включая средства нагрузочного тестирования, проверки пользовательских интерфейсов, тестирования ошибок исполнения [5].

Качество – это полное удовлетворение согласованных требований потребителя, причем внутренние затраты организации должны быть как можно ниже. Качество программного продукта определяется по несколькимкритериям. Качественный программный продукт должен отвечать функциональным и нефункциональным требованиям, в соответствии с которыми он создавался, иметь ценность для бизнеса, отвечать ожиданиям пользователей [6].

В жизненном цикле управления приложениями качество должно отслеживаться на всех этапах жизненного цикла ПО. Оно начинает формироваться с определения необходимых требований. При задании требований необходимо указывать желаемую функциональность и способы проверки её достижения.

Существует несколько подходов к управлению качеством:

Контроль качества. При этом подходе предлагаются различные способы обнаружения недостатков «на выходе». Готовый продукт либо принимается, либо отправляется на доработку. Если обнаруживаются недостатки, исследуются причины с целью устранения проблем. Это реактивный подход к качеству: «Ждать, пока что-нибудь не испортится, а затем попытаться устранить неполадки».

Культура качества. При этом подходе предлагается строить взаимоотношения внутри организации на основании концепции «поставщик-потребитель». Вводятся понятия внутренних и внешних поставщиков и потребителей. Это проактивный (предупреждающее действие) подход. В соответствии с ним некачественный продукт просто не производится. Культура организации становится более адаптивной и лояльной к изменениям. Это крайне важно для нашей отрасли, которая отличается высокой динамичностью и конкуренцией.

Таким образом, процесс управления качеством можно определить как процесс организации производства, способного осуществлять проектирование, создание и доставку продукции, соответствующей определенному уровню ожиданий потребителей.

Для управления качеством, оно должно быть измерено. В основе измерения качества лежит определение его пяти элементов:

- спецификация продукта;
- соответствие ожиданиям потребителя;
- надежность продукта (как долго он будет работать?);
- стоимость (прямые и косвенные затраты по приобретению и владению продуктом);
- поставка (определяет сроки, когда потребитель получит продукт).

Эти концепции учтены в MSF и органично вписаны во все модели. И поэтому их сложно выделить как отдельные механизмы. Например, в соответствии с Моделью проектной группы, целью ее работы является создание качественного продукта.

Документирование. Документирование проекта осуществляется разными специалистами. Руководство пользователя обычно создается техническим писателем, и его основной инструмент – это текстовый редактор и какое-нибудь не слишком сложное средство обработки графических изображений, с помощью которого можно добавить к снимкам экрана стрелки, выноски и иные элементы, которые принято изображать на иллюстрациях подобных документов. Документацию же по развертыванию и сопровождению продукта, равно как и иные технические документы, обычно создают аналитики или специалисты по развертыванию. В этом случае им может потребоваться многое из того, что было перечислено выше, как-то: средства управления требованиями, инструменты моделирования данных и UML-моделирования – нередко значительная часть таких документов состоит именно из отчетов по моделям. При этом чем аккуратнее велась работа над моделью, тем проще создавать проектную документацию.

На основе руководства пользователя с помощью специальных инструментов обычно генерируются файлы справочной системы. В простейшем случае файл справочной системы можно создать с помощью Microsoft Word и утилит от Microsoft для создания help-файлов, включаемых в состав многих средств разработки, но при большом объеме работы нередко используются специализированные средства таких компаний, как Blue Sky Software, EC Software, JGsoft.

Внедрение и сопровождение. Перед внедрением продукта обычно создаются дистрибутивные приложения, облегчающие этот процесс, – именно их мы запускаем, устанавливая на компьютер тот или иной продукт. Для создания дистрибутивных приложения также применяются специализированные средства, лидерами рынка которых являются компании InstallShield Software и Wise Solutions. Нередко в состав средств разработки входят специализированные версии указанных продуктов, учитывающие их специфику (например, возможность включения в дистрибутив библиотек, входящих в состав данного средства разработки).

Ни один проект не обходится без человека, несущего за него ответственность и осуществляющего планирование деятельности всех специалистов и управление всеми процессами разработки. Хотя планировать работу можно и на бумаге, но в последнее время основным инструментом руководителя проекта (а в крупных проектах – менеджеров, отвечающих за составные части

проекта) служит какое-либо специализированное средство управления проектами. Лидером среди продуктов данной категории является семейство продуктов Microsoft Project.

Инструментарий TeamFoundationServer позволяет формировать и отслеживать требования к программной системе, связывать их с задачами и реализацией, распределять между членами команды, проводить построение программного продукта, управлять тестированием, проводить контроль версий, предоставлять средства коммуникации с членами команды и заказчиками, подготавливать многочисленные отчеты.

Основным средством разработки в VisualStudio2013 является интегрированная среда разработки (IDE).IDE-среда интегрирована со средствами модульного тестирования и обеспечивает возможности выявления неэффективного, небезопасного или плохо написанного кода, управление изменениями и модульное тестирование как кода, так и базы данных.

Важным аспектом создания качественного ПО является обеспечение нефункциональных требований, таких как удобство в эксплуатации, надежность, производительность, защищенность, удобство сопровождения. Надежность ПО определяет способность без сбоев выполнять заданные функции в заданных условиях и в течение заданного отрезка времени. Производительность характеризуется временем выполнения заданных транзакций или длительных операций. Защищенность определяет степень безопасности системы от повреждений, утраты, несанкционированного доступа и преступной деятельности. Удобство сопровождения определяет легкость, с которой обслуживается продукт в плане простоты исправления дефектов, внесения корректив для соответствия новым требованиям, управления измененной средой.

Управление жизненным циклом программного продукта помогает разработчикам целенаправленно добиваться создания качественного ПО, избегать потерь времени на переделку, повторное проектирования и перепрограммирование ПО.

Выбор проектных методологий, моделей ЖЦ ПО, метрик проектов и других инструментальных средств управления ПО представляет собой достаточно сложную задачу для компаний-разработчиков ПО и проектных команд. Гибкость позволяет делать команды кроссфункциональными и самоорганизующимися, улучшить процессы отслеживания заданий и управления проектом, улучшить обратную связь между заказчиком и разработчиками и повысить точность планирования.

Методология управления проектами полностью отлична от чисто технической методики, которая часто связана с большинством проектом. В реальной жизни существует множество аспектов проекта, которые лежат вне границ технических областей и которые необходимо организовывать с максимально возможными тщательностью и вниманием. То есть, чтобы достичь поставленных перед проектом целей при оптимальном использовании ресурсов и максимальном удовлетворении участников проекта, такие нетехнические аспекты проектов должны быть хорошо управляемы, а это во многом зависит от компетенции проектных менеджеров и команд управления проектами.

ЛИТЕРАТУРА

[1] PMBOK – Project Management Body of Knowledge, USA, 2005.

[2] Грекул В.И., Коровкина Н.Л., Куприянов Ю.В. Методические основы управления ИТ-проектами. Учебник. – М., 2010.

[3] Крылов Е.В., Острейковский В.А., Типикин Н.Г. Техника разработки программ. – М.: Высшая школа, 2008.

[4] Гвоздева Т.В., Баллод Б.А. Проектирование информационных систем. – М.: Феникс, 2009.

[5] Орлов С.А. Технологии разработки программного обеспечения. Учебник для вузов. 4-е издание. Стандарт третьего поколения. Учебник для вузов. – М.: Мир книг, 2012.

[6] Robert Happy. Project 2010 Project Management: Real World Skills for Certification and Beyond, John Wiley & Sons. – 2010. – 484 p.

REFERENCES

[1] PMBOK – Project Management Body of Knowledge, USA, 2005 (in English.).

[2] Grekul V.I., Korovkina N.L., Kupriyanov Yu. Methodical bases of management of IT projects. Textbook. Moscow, **2010** ((in Russ.)

[3] Krylov E.V., Ostreikovskaya V.A., Tipikin N.G. Technique development programs. M.: High School, **2008** ((in Russ.)

[4] Gvozdeva T.V., Ballod B.A. Information systems development. M.: Phoenix, **2009** ((in Russ.)

[5] Orlov S. Software development technology. Textbook for high schools. 4th edition. Third-generation standard. Textbook for high schools. M.: The world of books, **2012** ((in Russ.).

[6] Robert Happy. Project 2010 Project Management: Real World Skills for Certification and Beyond, John Wiley & Sons, **2010**, 484 p. (in English.).

АТ-ЖОБАЛАРЫН БАСҚАРУДЫҢ КЕЙБІР ӘДІСТЕМЕЛЕРІНІҢ ЕРЕКШЕЛІКТЕРІ

А. С. Жұмаханова

С. Сейфуллин атындағы Қазақ Агротехникалық университет, Қазақстан

Тірек сөздер: АТ-жоба, ақпараттық технологиялар, программный продукт, модель, MSF, PERT, Agile, PMBoK, Oracle.

Аннотация. Ақпараттық технологиялар (АТ) саласындағы жобалар - инновациялық жобалардың бір түрі, әрі олардың өзіне тән ерекшеліктері бар. Осы мақалада АТ-жобаларын жүзеге асырудың негізгі ерекшеліктері зерттеліп, оларды тиімді басқарудың әдістемелері ұсынылды. Қзіргі уақытта PMBOK, SCRUM, Agile, PRINCE2 және тағы басқа жобаларды басқару стандарттары бар. Бірақ ол стандарттардағы ұстанымдарды қолданып, АТ-жобаларын тиімді басқарудың мәселелерін толық шеше алмайды. Ол үшін оларды нақты жағдайға және АТ-компаниясының тәжірибесіне қарай бейімдеу қажет.

Поступила 20.03.2015 г.