УДК 681.3.07

И.А. Ушакова

Харьковский национальный экономический университет, Харьков

ОПРЕДЕЛЕНИЕ ТРЕБОВАНИЙ К ПРОГРАММНОМУ ПРОДУКТУ НА ОСНОВЕ ПРОТОТИПОВ

Успешное завершение IT-проектов зависит в значительной мере от точности определения требований к программному обеспечению на начальной фазе разработки проекта. Основными причинами сложности определения требований являются необходимость учета пожелания большого числа потенциальных заинтересованных лиц, многообразие типов требований, сложная иерархическая структура требованиями, трассировка требований. Применение прототипов и особенно раскадровки для определения требований при выполнении крупных IT-проектов на начальной фазе разработки позволит повысить точность определения требований, свести к минимуму необходимость внесения изменений на последующих фазах разработки проекта. Рассмотрены и проанализированы основные виды прототипов для определения требований к программному обеспечению, наиболее распространенные инструментальные средства раскадровки требований, методические подходы к выполнению раскадровок.

***Ключевые слова**: программное обеспечение, определение требований к программному обеспечению, прототип, раскадровка требований, RUP.*

Введение

Определение требований к программному обеспечению (ПО) в IT-проектах является очень важной задачей начальной фазы разработки проекта. Правильно определенные требования являются гарантией того, что система будет удовлетворять требованиям лиц, заинтересованных в ее разработке, так называемых стейкхолдеров (stakeholders). Для больших систем количество требований может быть весьма значительным и при этом требования могут меняться, кроме того, они часто связаны между собой. Причем всегда существует вероятность того, что некоторые из требований заинтересованных лиц не будут реализованы или неправильно реализованы.

Основная проблема, связанная с определением требований, заключается в том, что выявить окончательные требования до внедрения программного продукта бывает очень не просто, так как процесс выявления требований зачастую протекает неформально. Кроме того, требования собираются из разных источников, что так же затрудняет процесс их однозначного толкования и отслеживания изменений.

Исследования, проводимые в этой области, направлены в основном на точность, представление, методы моделирования и верификации требований. В то время как проблеме улучшения определения требований уделяется не достаточно внимания [7].

Обзор литературных источников. Анализ публикаций, касающихся определения требований к программному продукту, показал, что основными причинами сложности определения требований являются следующие [6, 9]:

– при определении требований к ПО необходимо учесть пожелания большого числа потенциальных заинтересованных лиц, что является характерным для таких проектов;

– многообразие требований разных типов. Каждый тип требования должен быть описан с определенной степенью детализации;

– требования могут представлять собой сложную иерархическую структуру, которую нужно поддерживать;

– требованиями различных типов могут быть связаны между собой, поэтому необходимо отслеживать трассировку требований, то есть устанавливать изменять и удалять взаимосвязи между ними.

Указанные причины приводят к тому, что требования меняются в ходе выполнения проекта, что по нашему мнению в первую очередь связано с недостаточной степенью их определения в начале проектирования.

Основными проблемами процесса определения требований являются следующие [4]:

– организация сбора, анализа и оценка идей для сбора требований. Решение этих проблем оказывает влияние на весь процесс разработки проекта, воздействуя на информацию, коммуникации, и инструментальные средства. Основная проблема при этом заключается в координации действий персонала, организации сотрудничества. Многие современные компании используют глобальную цепочку поставок программного обеспечения, которая включает как различные подразделения внутри компании, так и подразделения, обособленные географически и по временным поясам, что значительно затрудняет их координацию;

– масштабируемость практических навыков. Неформальные методы выявления требований, которые сотрудники часто выбирают при работе над небольшими проектами, обычно дают неутешительные

результаты при работе над более крупными проектами. Кроме того, проекты, в которых большинство сотрудников работают в удаленном режиме, сильно отличаются от проектов, выполняемых коллективом с компактным размещением сотрудников;

– необходимая степень документирования и потребность в инструментальных средствах может варьировать в зависимости от масштаба проекта. Многие компании используют в качестве инструментальной среды при выявлении требований текстовые документы Word, презентации PowerPoint диаграммы Visio, электронные таблицы Excel. Последние могут использоваться для отслеживания требований и мест их возникновения. Такой инструментарий может использоваться в небольших проектах. При выполнении крупных или множественных проектов подобный инструментарий является малоэффективным;

– интеграция информации при использовании таких инструментальных средств Word, PowerPoint Visio, Excel является ограниченной. Данные инструменты могут поддерживать встраивание и обновление определенной информации, но при этом имеют ограниченные возможности при работе с реляционными данными. Теряется возможность отслеживания промежуточных результатов, затрудняет повторное использование полученных ранее результатов;

– необходимость поддерживать работу с требованиями в диалоговом режиме. Инструментальные средства Word, PowerPoint Visio, Excel не поддерживать работу в диалоговом режиме. При использовании данных инструментов, можно создавать репозитории для артефактов требований, использовать дискуссионные форумы или электронную почту для решения проблем. инструментальных средств Word, PowerPoint Visio, Excel при этом все равно создается разрозненная информация, которую еще необходимо собрать и каталогизировать для получения контекста.

Применение раскадровок можно использовать в качестве инструментального средства для определения требований. Кроме того, раскадровки могут быть с успехом использованы проектировщиками, разработчиками, тестировщиками и архитекторами, а также дизайнерами пользовательских интерфейсов [5].

Постановка задачи

Актуальность темы. Важность поставленной проблемы определяется тем, что ошибки и неточности в определении требований приводят к необходимости внесения изменений в проект. Это приводит к необходимости исправления в архитектуры системы, программного кода, документации и т.д. Все это требует дополнительных финансовых и трудовых затрат. Это, с одной стороны, отражается на финансовом положении компании-разработчике, а также ее имидже. С другой стороны, организация-заказчик так же несет дополнительные финансовые затраты, имеет риск получить продукт не вовремя или не в полной мере отвечающий ее требованиям.

К типичным причинам срыва сроков и превышения бюджета проектов можно отнести следующие:

– требования заказчиков не удалось полностью определить;

– требования не были четко сформулированы;

– в процессе работы над проектом не удалось отследить изменения требований.

Мировой опыт разработки программных систем показывает, что устранение ошибок в требованиях на стадии сопровождения готового ПО обходится в 200 раз дороже, чем на стадии спецификации требований [6]. Поэтому ошибки в требованиях, которые были обнаружены на более поздних фазах проекта требуют дополнительных финансовых затрат, которые составляют 30 — 40% бюджета проекта.

В табл. 1 представлен анализ основных причин неуспешно выполненных IT-проектов [4].

Следовательно, основной причиной неудачно выпущенного ПО является плохая организация процесса определения и управления требованиями. Поэтому в условиях постоянно растущего спроса на IT-проекты проблема определения требований является важной задачей.

Таблица 1

Причины неуспешно выполненных IT-проектов

Причина	Процент IT-проектов
Редко выполняемый код	12
Код написан, но никогда не выполнялся	45
Неудачное завершение проекта вследствие плохого определения и управления требованиями	60 – 70

Цель данного исследования заключается в анализе существующих методов определения требований к программному обеспечению, а также инструментальных средств, поддерживающих эти методы и использовании их при определении требований на ранних этапах разработки проекта.

Объектом исследования является процесс определения требований к ПО в IT-проектах.

Предметом исследования являются методы и инструментальные средства определения требований ПО.

Основная часть

Определение требования к программному обеспечению. Процесс определения требований к ПО базируется на понятии требования. Рассмотрим определение этого понятия в с точки зрения программной инженеприи.

В стандарте (IEEE Standard Glossary of Software Engineering Terminology, 1990), разработанным Институтом инженеров по электротехнике и электронике программное требование (Software Requirement) определяется следующим образом:

1. Условие или возможность, необходимая пользователю для решения задач или достижения цели.

2. Условие или возможность, которой должна обладать система или ее компонент, соответствующие договору, стандарту, спецификации или другому официальному документу.

3. Документированное представление условия или возможности перечисленные в предыдущих пунктах.

Наиболее фундаментальной проработанной методологией в области программной инженерии является Rational Unified Process (RUP). В старой редакции RUP ппрограммное требование определяется как возможность, которую кто-либо ожидает от данного ПО. В последней редакции RUP под требованием к ПО подразумевается спецификация наблюдаемого поведения системы, например, входных и выходных данных, функций или атрибутов системы или атрибутов внешней среды системы [1].

Зависимость стоимости внесения изменений в разрабатываемое ПО в зависимости от фазы его жизненного цикла. В основе жизненного цикла разработки ПО в соответствии методологией RUP лежит итерационная модель разработки, включающая следующие фазы: Начало (Inception), Проектирование (Elaboration), Построение (Construction), Внедрение (Transition). Мировой опыт разработки программных систем показывает, что устранение ошибок в требованиях на фазе внедрения готового ПО обходится в 200 раз дороже, чем на начальной фазе при спецификации требований [6]. Поэтому ошибки в требованиях, которые были обнаружены на более поздних фазах проекта требуют дополнительных финансовых затрат, которые составляют 30 – 40% бюджета проекта.

Преимущества точного определения требований на начальной фазе разработки проекта. Точное определение требований позволяет компании получить следующие преимущества[4, 10]:

– сокращение объемов переработки кода, связанной с неучтенными или неправильно определенными требованиями;

– ускорение выхода продукта на рынок, вследствие повышения производительности работы над проектом,

– повышение качества выполнения проекта.

Важность перечисленных преимуществ показывают современные исследования. Задержка выпуска программного продукта на полгода может снизить прибыльность проекта до одной трети. В тоже время ускорение выпуска продукта на один месяц дает потенциальное повышение его прибыльности более чем на 10 процентов.

Прототипирование – как область, способствующая успешному определению требований. Прототип — это работающая модель программы с неполным функционалом. Он обычно содержит графический интерфейс пользователя и выглядит как настоящая программа, однако активизация элементов интерфейса не приводит к получению результата.

Прототипирование программного обеспечения (от англ. prototyping) — этап разработки ПО, в процессе которого создается его прототип.

С помощью прототипов можно увидеть элементы реальной программной системы, поработать с ними до ее создания. Использование прототипов на ранних стадиях выполнения проекта позволяет уточнить и справить ошибки в определении требований на основе обратной связи с заинтересованными лицами.

Цели прототипирования. Основными целями прототипирования являются [2]:

уточнение нечетких требований к системе;

выбор альтернативного решения;

анализ осуществимости ПО.

Исходя из этого основная цель прототипирования заключается в том, чтобы показать заинтересованным лицам как будет выглядеть и работать программа без затрат на ее создание. Согласованный с заинтересованными лицами прототип программы можно использовать для создания окончательного варианта программы.

Классификация прототипов. В настоящее время традиционно все виды прототипов принято классифицировать следующим образом (рис. 1):

– по назначению – горизонтальные и вертикальные;

– по глубине проработки кода – одноразовые и эволюционные.

Рис. 1. Классификация прототипов ПО

Горизонтальный или поведенческий прототип (horizontal prototype, behavioural prototype) имитирует интерфейс пользователя, не затрагивая при этом логику обработки и базу данных. Такие прототипы обычно используются для прояснения неясных или многоальтернативных требований.

Вертикальный или структурный прототип (vertical prototype, structural prototype) включает как интерфейс пользователя, так и реализацию всех уровней ее реализации. Назначение таких прототипов – это анализ применимости системы, проверка ее архитектурных концепций.

Одноразовый или исследовательский прототип (throwaway prototype, exploratory prototype) создается, когда нужно быстро смоделировать некоторые аспекты и компоненты системы. Такой создаваться быстро, без проработки вопросов повторного использования кода, его качества и т.п.

Эволюционный прототип (evolutionary prototype) – это наиболее проработанный прототип, который создается, как первое приближение системы, на основе которого впоследствии будет создана сама система.

Особый интерес в связи с проблемой определения требований представляет классификация прототипов в зависимости от типа используемых инструментальных средств: электронные прототипы; раскадровки.

Электронный прототип (electronic prototype) основан на использовании языков программирования высокого уровня абстракции, таких как Java, Perl, Python, Haskell и т.п.

Раскадровка (storyboard) – это логическое и концептуальное описание функциональных возможностей системы для определенного сценария, включая необходимое взаимодействие между системой и ее пользователями. В качестве инструментальных средств раскадровки требований используются Microsoft Word, Microsoft Visio, Microsoft PowerPoint, IBM Rational Requirements Composer, Expression Blend SketchFlow.

Раскадровки делят на три типа:

пассивные раскадровки, в виде истории, рассказанной пользователю. Она включает схемы копии экранов, презентации PowerPoint и формы выходной информации т.п. Аналитик играет роль системы, которая сводится к рассказу пользователю о том, как будет работать система;

активные раскадровки используют средства анимации или автоматизации. Например, с помощью автоматического показа слайдов, анимации, фильмов. Применяются для показа типового поведения системы;

интерактивные раскадровки, позволяющие пользователю получить опыт работы с системой. Данный тип раскадровки представляет собой электронный одноразовый горизонтальный прототип

Состав раскадровки. в Rational Requirements Composer. Одним из наиболее мощных средств разработки раскадровки требований на сегодняшний день является Rational Requirements Composer [5]. Rational Requirements Composer позволяет заинтересованным лицам описать свои потребности аналитикам, которые на основе этих потребностей определяют требования к системе, осуществляют проверку соответствия требований поставленным бизнес-задачам и осуществляют с ними обратную связь.

Основными компонентами данного прототипа являются: персонаж, сценарий использования и непосредственно раскадровка.

Персонаж - это вымышленный типичный пользователя системы. Его назначение - «оживление» пользователя системы, посредством описания его индивидуальных характеристик. пользователя посредством предоставления. Персонаж может быть связан с несколькими раскадровками.

Сценарий использования описывает взаимодействие персонажа с системой при выполнении определенной задачи. Если при разработке используются варианты использования (use case) сценарий использования обычно представляет собой полный или частичный сценарий вариантов использования (use case scenario).

Раскадровка – это представление сценария использования по кадрам. В каждом кадре этого сценария имеется описание действий, приводящих к появлению следующего кадра. Раскадровка включает «истории» - подробное прохождение по линейному сюжету, представленное в виде расположенных на временной шкале (timeline) графических кадров с образцами данных. По существу, раскадровка – это последовательность кадров, каждый из которых конкретизирует возможность пользователя в соответствующей ситуации. В состав раскадровки входит список кадров, средство просмотра временной шкалы и сами кадры. Кадры – это фактически отдельные экземпляры эскизов внутри раскадровки.

Определение с помощью раскадровок. Раскадровки могут применяться уже на фазе Inception (Начало) для выявления того, какие возможности заинтересованное лицо желает получить. По результатам создания раскадровки должны быть сформулированы требования (с возможностью отследить процесс «в обратном направлении» вплоть до конкретного элемента раскадровки), подлежащие согласованию с заинтересованными лицами. Затем, эти требования должны быть включены в соответствующий набор требований, из которого первоначально появилась потребность в раскадровке. Типичный шаблон для создания раскадровки включает предварительные условия и выходные условия, описываемые в следующих разделах.

Предварительные условии.я. До проведения раскадровки необходимо выполнить следующее:

сформулировать проблему,

сформулировать цели раскадровки,

выбрать типа раскадровки,

идентифицировать фазы жизненного цикла разработки,

задать уровня точности раскадровки,
определить роли заинтересованных лиц,
определить ограничения.

Выходные условия. После завершения раскадровки необходимо выполнить валидацию созданной раскадровки, определить необходимость в любых последующих работах.

Выводы

Анализ основных причин сложности определения требований, таких как необходимость учета пожелания большого числа потенциальных заинтересованных лиц, многообразие типов требований, сложная иерархическая структура требованиями, трассировка требований показал, что вследствие недостаточной степени определения требований на начальных этапах разработки проекта, требования могут изменяться в ходе выполнения последующих его фаз.

Основной причиной неудачно выпущенного ПО является плохая организация процесса определения и управления требованиями. Решением задачи определения требований на начальной фазе разработке проекта может быть использование прототипов и особенно раскадровки требований.

Наиболее мощным инструментальным средством рскадровки требований в настоящее время является программный продукт в Rational Requirements Composer, поддерживаемый методологией RUP. В сочетании с другими инструментальными средствами, поддерживающими управление требованиями, данный продукт может решить задачу работы с требованиями при разработке крупных IT-проектов.

Список литературы

1. IBM Rational Unified Process (RUP), 2008 [Электронный ресурс]. – Режим доступа к ресурсу: http://www.ibm.com.

2. Вигерс К. Разработка требований к программному обеспечению / К. Вигерс. – М.: Издательско-торговый дом "Русская Редакция", 2004. – 576 с.

3. Определение требований с IBM Rational Requirements Composer 2010 [Электронный ресурс]. – Режим доступа к ресурсу: http://www.interface.ru/home.asp?artId=23530

4. Определение требований как основа эффективности поставки программного обеспечения [Электронный ресурс]. – Режим доступа к ресурсу: http://www.ibm.com/developerworks/ru/library/r-raw/index.html - сайт IBM.

5. Применение раскадровок (Storyboard) в инструменте IBM Rational Requirements Composer [Электронный ресурс]. – Режим доступа к ресурсу: http://www.ibm.com/developerworks/ru/library/r-1118_zhuo/ - сайт IBM.

6. Роль процесса Управления Требованиями при разработке сложных программных систем. Практика применения методологии IBM RUP и инструмента IBM Rational RequisitePro [Электронный ресурс]. – Режим доступа к ресурсу: http://cmcons.com/articles/upravlenie_trebovanijami_instrument_ibm_rational_r/rol_protsessa_upravlenija_trebovanijami_pri_razrabotke_slozhnykh_programmnykh_sistem_praktika_primenenija_metodologii_ibm_rup_i_instrumenta_ibm_rational_requisitepro/

7. Руководство по управлению требованиями VS TFS 2010 [Электронный ресурс]. – Режим доступа к ресурсу: http://ashamray.wordpress.com/2010/07/27/ vsts_2010 _ req_management/

8. Ушакова І.О. Визначення функціональних вимог до системи на основі розкадровувань / І.О. Ушакова // Системи обробки інформації: зб. наук. пр. – Х.: ХУПС, 2009. – Вип. 7 (79). – С.147-148.

9. Ушакова І.О. Трасування вимог в Use Cases проектах / І.О. Ушакова // Управління розвитком: зб. наук. статей. – Х.: Вид. ХНЕУ, 2008. – № 14. – С. 37-39.

10. Ушакова І.О. Практичні аспекти управління вимогами при розробленні програмних систем / І.О. Ушакова, Л.В. Лощина // Управління розвитком: зб. наук. статей ХНЕУ. – Х., 2007. – № 7. – С. 64-66.

Поступила в редколлегию 4.10.2010

Рецензент: д-р экон. наук, проф. А.И. Пушкарь, Харьковский национальный экономический университет, Харьков.

ВИЗНАЧЕННЯ ВИМОГ ДО ПРОГРАМНОГО ПРОДУКТУ НА ОСНОВІ ПРОТОТИПІВ

І.О.Ушакова

Успішне завершення IT-проектів залежить значною мірою від точності визначення вимог до програмного забезпечення на початковій фазі розроблення проекту. Основними причинами складності визначення вимог є необхідність обліку побажання великого числа потенційних зацікавлених осіб, різноманіття типів вимог, складна ієрархічна структура вимог, трасування вимог. Застосування прототипів і особливо розкадровки для визначення вимог при виконанні великих IT-проектів на початковій фазі розроблення дозволить підвищити точність визначення вимог, звести до мінімуму необхідність внесення змін на наступних фазах розробки проекту. Розглянуто та проаналізовано основні види прототипів для визначення вимог до програмного забезпечення, найбільш поширені інструментальні засоби розкадровки вимог, методичні підходи до виконання розкадровок.

Ключові слова: *програмне забезпечення, визначення вимог до програмного забезпечення, прототип, розкадровка вимог, RUP.*

DETERMINATION OF SOFTWARE REQUIREMENTS IS TO PRODUCT ON BASIS OF PROTOTYPES

I.O. Ushakova

Successful completion of IT-projects depends largely on the accuracy of the software requirements for the initial phase of project development. The main causes of difficulty in determining the requirements are the need to consider the wishes of a large number of potential stakeholders, a variety of types of requirements, a complex hierarchical structure of requirements, trace requirements. The use of prototypes, and especially the storyboard to determine the requirements for the implementation of large IT-projects in the initial phase of development will improve the accuracy of the claims, to minimize the need for changes in subsequent phases of project development. Reviewed and analyzed the main types of prototypes to determine the requirements for software, the most common tools storyboard requirements, methodological approaches to the implementation of storyboards.

Keywords: *software, definition of software requirements, prototype, storyboard requirements, RUP.*