УДК 681.518:658.512

Н.В. ТКАЧУК, Р.А. ГАМЗАЕВ

Национальный технический университет «ХПИ», Харьков, Украина

ОСОБЕННОСТИ ПРОЦЕССОВ УПРАВЛЕНИЯ ТРЕБОВАНИЯМИ В ГИБКИХ МЕТОДОЛОГИЯХ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Рассмотрены некоторые общие проблемы управления требованиями (УТ) при разработке программного обеспечения (ПО) с использованием гибких (agile-) методологий. На примере анализа процессов УТ в методологии SCRUM показана целесообразность применения кибернетических принципов для построения эффективных схем УТ в виде многоконтурной системы управления с обратной связью, в которой для формирования управляющих воздействий используются соответствующие метрики качества ПО. Определены принципы для дальнейшей модельно-инструментальной реализации предложенного подхода.

Ключевые слова: *управление требованиями, agile-методологии, программная кибернетика, метрики программного обеспечения.*

1. Актуальность проблем управления требованиями при использовании agile-технологий разработки программного обеспечения

В настоящее время многие фирмы-разработчики программного обеспечения (ПО) переходят к использованию так называемых гибких методологий разработки (agile- methodologies). Прежде всего это видно на примере таких крупных компаниях как IBM где, по заявлению ее ведущих специалистов, около 25% проектов уже выполняется с использованием этого подхода [1]. С другой стороны, сходная тенденция прослеживается и в практике работы Интернет-бирж по разработке ПО, где заказчики даже небольших проектов часто вместе со знанием средств разработки, наличия опыта работы, выдвигают такие требования к исполнителям как понимание ими agile-методологий, умение работать в agile-команде и т.д.

Основными причинами актуальности agile-подходов следует считать несоответствие традиционных моделей жизненного цикла (ЖЦ) разработки ПО: прежде всего, таких как, каскадная и спиралевидная [2], новым реалиям процессов разработки ПО, а именно: появление «быстрых» инструментальных средств разработки (Rapid Application Development - RAD), рост сложности и изменчивости (непредсказуемости) требований к ПО в таких областях как Web-приложения, мобильные системы и т.п. (см., напр., в [3]). При этом оказалось, что попытки решить эти проблемы путем прямого применения в программной инженерии принципов конструирования и реализации сложных технических систем оказались неэффективными, поскольку сам процесс разработки ПО является менее формализуемыми, недетерминированным, с присутствием весьма большого влияния человеческого фактор [3,4]. Все это потребовало перехода к более гибким, адаптивным моделям процессов ЖЦ ПО.

Сам термин «agile» впервые появился в 2001 году, когда международным консорциумом разработчиков ПО был сформулирован программный документ “The Manifesto for Agile Software Development” [5]. Он содержит 12 базовых принципов, которые характеризуют эти подходы. В частности, они определяют, что

a) личности разработчиков и степень эффективности их взаимодействия важнее для успеха выполнения проекта, чем сами процессы и инструменты разработки ПО;

b) корректно спроектированное и работающее ПО важнее, чем наличие его полной документации;

c) постоянное сотрудничество с заказчиком проекта по разработке ПО важнее, чем формальное выполнение контрактных обязательств;

d) адекватная и быстрая реакция на изменения в требованиях к ПО важнее, чем жесткое следование первоначальному плану выполнения проекта,

и ряд других положений [5].

На сегодняшний день существует несколько agile-подходов к управлению процессом разработки ПО, например: eXtreme Programming (XP), ASD (Adaptive Software development), SCRUM и некоторые другие [4]. Каждый из них имеет свои преимущества и недостатки, и целесообразность выбора того или иного зависит от ряда характеристик конкретного проекта, что само по себе, является интересной и важной научно-технической задачей. При этом подавляющее большинство авторов работ по

исследованию особенностей применения agile-подходов отмечают, что именно задача управления требованиями (requirements management) является наиболее сложной и важной в любой из agile-методологий, поскольку эффективность ее решения определяет в целом, при прочих равных условиях, степень адаптивности всего процесса разработки ПО.

2. Некоторые подходы к инженерии требований в agile-методологиях и их недостатки

Управление требованиями (УТ) представляет собой совокупность процессов по выявлению потребностей заказчика проекта, проведению их анализа и проверки (валидации), и, в конечном итоге, составление спецификаций для разработки соответствующего проектного решения. В традиционных моделях ЖЦ для этого предназначена первая фаза проекта: этап сбора и анализа требований, после чего они считаются вполне определенными и на их основе начинается этап непосредственного проектирования ПО. Но уже в таких современных методологиях как, например, RUP (Rational Unified Process) [4] процессы УТ рассматриваются как неотъемлемая составная часть практически всех остальных этапов разработки ПО, таких как: кодирование (implementation), тестирование (testing) и развертывание (deployment) ПО. При этом RUP предполагает создание на всех этих этапах определенного набора спецификаций требований, как правило, в виде совокупности UML-диаграмм: диаграммы прецедентов (use case diagram), последовательностей (sequence diagram), устойчивости (robustness diagram), которые затем являются основой для создания других проектных документов: диаграмм классов (class diagram), диаграмм компонентов (component diagram), на основе которых, в конечном итоге, и создается работающий программный продукт.

В agile-методологиях, принимая во внимание их базовые принципы (a)-(d), также предполагается, что управление постоянно изменяющимися требованиями к ПО должно происходить непрерывно, в адаптивном режиме, что означает необходимость соответствующим образом быстро изменять проектные решения, конфигурацию средств разработки, а, возможно, и сам состав команды разработчиков проекта [4]. При этом в большинстве работ по анализу особенностей agile-подходов отмечается, что основными технологиями, которые позволяют обеспечить эти свойства процессов УТ являются [3-5]:

1) *итеративность* - т.е. представление всего процесса разработки в виде той или иной последовательности весьма коротких циклов (итераций) по созданию очередной версии ПО, что позволяет эффективнее реализовывать изменения в требованиях;

2) *трассировка требований* – это технология установления связей между различными артефактами жизненного цикла требований, например, их текстовыми спецификациями, диаграммами вариантов использования ПО и фрагментами исходного кода, с целью отследить степень влияния изменений в требованиях на характеристики всего проекта в целом.

Рассмотрим более подробно процессы УТ на примере одного из наиболее распространенных и хорошо структурированных agile-подходов, а именно методологии SCRUM [6], общая логическая схема которого показана на рис. 1.

Первоначально требования формируются в виде каталога требований к продукту (Product Backlog - PB), который составляется заказчиком проекта либо его полномочным представителем (*Product Owner - PO*) совместно с командой разработчиков проекта (*SCRUM Team - ST*). Затем эти требования приоритизируются *PO* - с точки зрения их важности для бизнес-логики всей системы, и оцениваются группой *ST* - относительно трудоемкости их реализации. При этом новые требования могут добавляться в *PB* в ходе всего проекта.

Сам процесс разработки ПО происходит итеративно, в начале каждой итерации (*Sprint - S*) из каталога требований *PB* формируется их некоторое

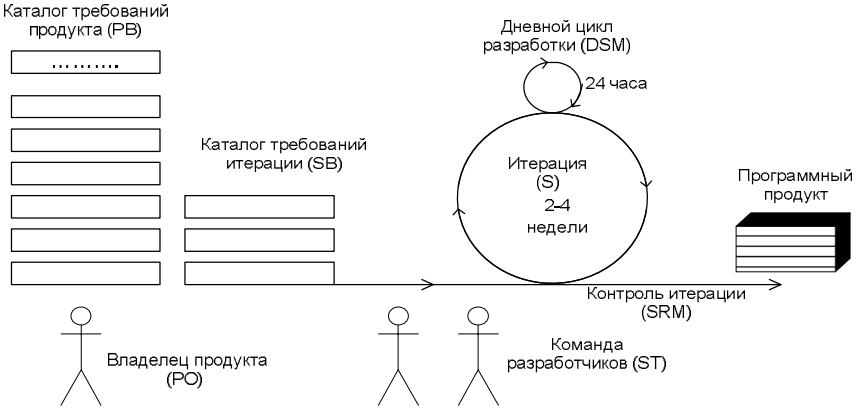


Рис. 1. Логическая схема основных этапов и артефактов SCRUM

подмножества, которое фиксируется в т.н. журнале спринта (*Sprint Backlog - SB*). В рамках данной итерации требования, зафиксированные в *SB*, не могут быть изменены по решению извне (напр., по решению *PO*). Но они могут быть скорректированы по решению участников *ST*, если в ходе итерации в требованиях из *SB* найдены такие ошибки, которые могут помешать ее успешному завершению, либо если в ходе итерации представления о свойствах проектируемой системы значительно расширились, что является поводом для коррекции требований в *SB*. Для оперативного управления процессом разработки в ходе текущей итерации *S* ежедневно проводится краткое рабочее совещание (*Daily SCRUM Meeting - DSM*), в ходе которого все участники *ST* (при необходимости, и с участием *PO*) обсуждают текущее состояние проекта и корректирую свои представления о ходе реализации требования из *SB*.

В конце каждой итерации S проводиться т.н. демонстрационный митинг (*Sprint Review Meeting - SRM*), в ходе которого: 1) *ST* представляет *PO* полученные результаты работы, 2) происходит редактирование общего каталога требований *PB* с учетом состояния требований в журнале текущего спринта *SB*; 3) затем выполняется планирование выполнения новых требований для последующей итерации *S*.

При анализе процессов, представленных схемой на рис. 1, возникают, в частности, следующие проблемы:

- как приоритизируются требования в каталоге *PB*?
- каким образом определяется эффективная длительность итерации *S* для конкретного проекта?
- какие количественные метрики для обеспечения качества получаемого ПО применяются при этом?

Для решения этих задач необходимо перейти от качественных схем представления взаимосвязей основных этапов и артефактов в agile-методах, один из примеров которых показан на рис. 1, к их формализованному представлению, а затем – и к целенаправленному применению количественным метрик для определения эффективных значений их технологических параметров и получаемых результатов разработки ПО.

3. Кибернетический подход к управлению требованиями в agile-процессах разработки ПО

В соответствии с общей современной тенденцией применения принципов теории управления при решении задач программной инженерии, которая проявилась в появлении в этой области такого нового направления как *программная кибернетика* (software cybernetics) (см. напр., в [7]), рассмотрим такой подход к постановке и решению задачи УТ в agile-методологиях.

Следует отметить, что такие попытки уже предпринимаются, и, в частности, в работе [8] предлагается рассматривать основные процессы и артефакты методологии SCRUM в виде функциональной схемы некоторой системы автоматического управления (САУ), в которой выделены три контура управления: 1) контур дневного цикла разработки (DSM), 2) контур управления всей итерацией (SB / SRM), 3) контур управления всем циклом разработки программного продукта (PB), который, в свою очередь, может быть разбит на несколько его выпусков (версий). Соответствующая структурная схема такой системы показана на рис. 2, а.

Однако, такое представление еще не позволяет сделать вывод о том, как именно реализуются механизмы обратной связи в соответствующих контурах управления и каким образом при таком подходе обеспечивается качество конечного программного продукта. С целью устранения этих недостатков представляется целесообразным модифицировать схему на рис. 2, а, с учетом следующих принципиальных положений, а именно:

а) объектом управления в соответствующем контуре схемы САУ всегда является *набор (каталог) требований*, которые должны быть реализованы в конечном программном продукте при выполнении очередной итерации процесса разработки;

б) в механизмах обратной связи в этих контурах управления должны использоваться соответствующие *количественные метрики*: метрики исходного кода, метрики состояния требований и метрики качества программного продукта.

Модифицированная таким образом схема взаимодействия основных процессов и артефактов методологии SCRUM показана на рис. 2, б.

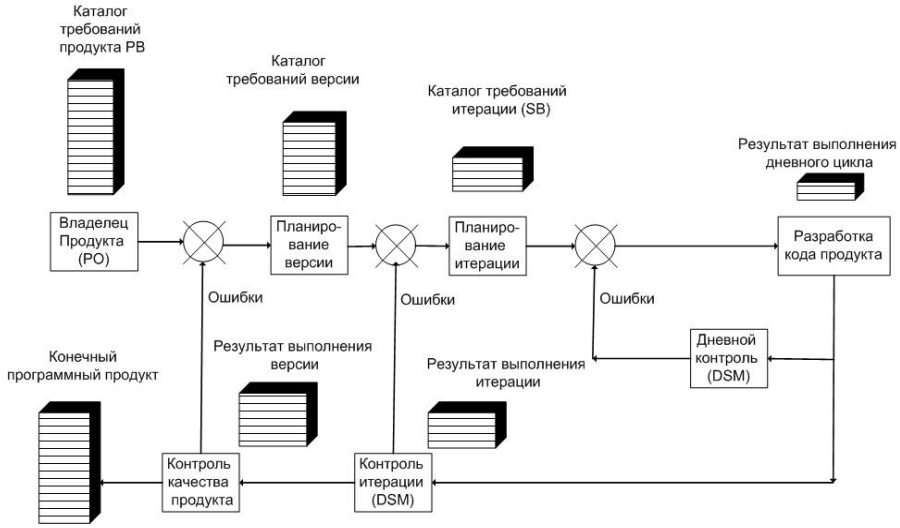
Следует отметить, что на ней, помимо сформулированных выше принципиальных положений (А)-(В), отражено и то обстоятельство, что во время выполнения каждой очередной итерации по реализа ции требований из каталогов PB и SB на вход контура управления могут поступать новые требования. Выбор соответствующих метрик для их использования в этой схеме возможен на основании подходов и рекомендаций, изложенных в таких исследованиях как, например, [9, 10].

Наиболее сложными для количественного измерения из этих метрик являются метрики качества требований (requirements metrics), поскольку для их определения в большинстве случаев необходимо использовать не только такие хорошо структурированные документы как, например, UML- или SADT – диаграммы и т.п., но и текстовые спецификации требований. На качественном уровне оценка таких

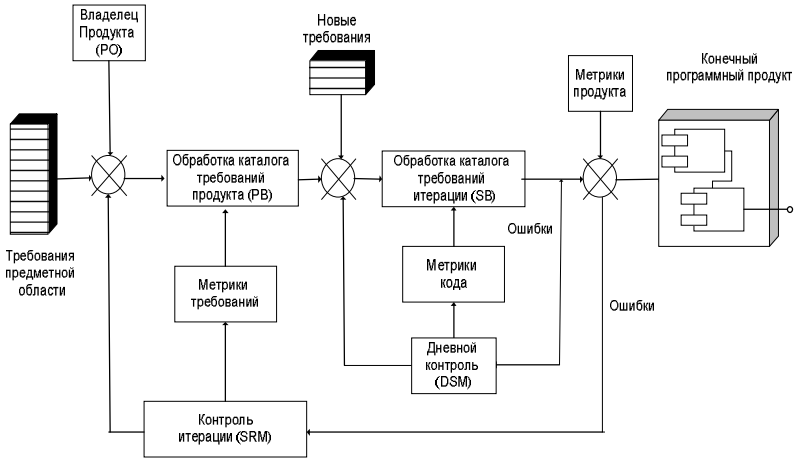
текстовых документов должна определить на каждой итерации *S* следующие метрические значения для совокупности требований в их каталоге *PB*:

- полноту описания требований (requirements completeness metric);
- степень их изменчивости (requirements volatility metric);
- уровень трассировки требований (requirements traceability metric).

Очевидно, что для автоматизации процессов определения и анализа этих показателей необходима разработка соответствующих CASE-средств, которые должны использоваться в контурах схемы управления, представленной на рис. 2, б.



а



б

Рис. 2. Представление методологии SCRUM в виде САУ:
а – схема из работы [8]; б – модифицированная схема

Выводы

Таким образом, в данной работе на примере методологии SCRUM проанализированы некоторые общие качественные особенности процессов управления требованиями в agile-подходах к разработке ПО и предложен кибернетический подход к интерпретации этой проблемы в виде схемы САР с обратной связью. На основе этой схемы можно сделать следующие выводы: поскольку в agile-подходах нужно постоянно учитывать новые изменения в требованиях, анализировать их и создавать проектные спецификации итеративно, в условиях ограниченного ресурса времени, в тесном взаимодействии с заказчиком проекта, для эффективного решения этой задачи необходимо:

1) при спецификации требований использовать такие формы их представления, которые максимально понятны как разработчику системы так и ее заказчику (владельцу требований), и которые позволяют использовать знание-ориентированные методы принятия решений при анализе и валидации требований;

2) для автоматизации процессов трассировки требований необходимо интегрировать функциональность существующих систем управления требо-

ваниями (requirements management system - RMS) и средств визуальной разработки ПО (integrated development environment - IDE) в единый инструментальный комплекс agile-разработки;

3) для обеспечения качества получаемого программного продукта и снижения затрат на его разработку применять как продуктовые так и процессные количественные метрики.

Решение этих задач рассматривается авторами как перспективные направления дальнейших исследований в этой области.

Литература

1. Ambler S. Strategies for Scaling Agile Software Development [Электронный ресурс]/ S. Ambler. – Режим доступа: http://www.dama-michigan.org/21%20Scott%20Ambler%20-%20Agile%20Data%20Workshop.pdf.

2. Соммервил И. Инженерия программного обеспечения: пер. с англ. / И. Соммервил. – М.: Изд. дом «Вильямс», 2002. – 624 с.

3. Коваль Г.І. Удосконалення процесу розробки сімейств програмних систем елементами гнучких технологій / Г.І. Коваль, А.Л. Колесник, К.М. Лаврiщева // Проблемы программирования. – К.: НАН України. – 2010. - № 2-3 (спец. вып.). - С. 261-270.

4. Амблер С. Гибкие технологии: экстремальное программирование и унифицированный процесс разработки. Библиотека программиста / С. Амблер. – СПб.: Питер, 2005. – 412 с.

5. The Manifesto for Agile Software Development. [Электронный ресурс]/ М. Борисов. – Режим доступа: http://agilemanifesto.org.

6. Борисов М. SCRUM – гибкое управление разработкой [Электронный ресурс] / М. Борисов. – Режим доступа: http://www.osp.ru/text/print/302/4220063.html.

7. A Control-Theoretic Approach to the Management of Software System Test Phase / S.D. Miller, R.A. DeCarlo, A.P. Mathur, J.W. Cangussu // Journal of Systems and Software; Special section on Software Cybernetics. – November 2006. – № 11 (79). – P. 1486–1503.

8. Anderson D.J. Agile Management for Software Engineering / D.J. Anderson // Prentice Hall, 2003. – 302 p.

9. Sulaiman AgileEVM – earned Value Management in SCRUM Projects / Sulaiman, Barton, Blackburn // Proc. of Agile-2006, 23-28 July 2006. – P. 349-360.

10. Kunz M. Software Metrics for Agile Software Development / M. Kunz, R.R. Dumke, N. Zenker // Proc. of 19th Australian Conference on Software Engineering (ASWEC-2008), March 26-28.2008. – P. 1022-1034.

Поступила в редакцию 22.11.2010

Рецензент: д-р техн. наук, проф., декан факультета информатики и управления, И.П. Гамаюн, Национальный технический университет «Харьковский политехнический институт», Харьков.

ОСОБЛИВОСТІ ПРОЦЕСІВ УПРАВЛІННЯ ВИМОГАМИ В ГНУЧКИХ МЕТОДОЛОГІЯХ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

М.В. Ткачук, Р.О. Гамзаєв

Розглянуто деякі загальні проблеми управління вимогами (УВ) при розробці програмного забезпечення (ПЗ) з використанням гнучких (agile-) методологій. На прикладі аналізу процесів УВ в методології SCRUM показано доцільність застосування кібернетичних підходів для побудови ефективних схем УВ у вигляді багатоконтурної системи управління із зворотним зв'язком, в якій для формування управляючих параметрів використовуються відповідні метрики якості ПЗ. Визначені принципи щодо її подальшої модельно-інструментальної реалізації.

Ключові слова: управління вимогами, agile-методології, програмна кібернетика, метрики програмного забезпечення.

SOME FEATURES OF REQUIREMENTS MANAGEMENT IN AGILE SOFTWARE DEVELOPMENT METHODOLOGIES

M.V. Tkachuk, R.O. Gamzayev

Some common problems of requirements management (RM) in agile software development methodologies are considered. By the example of RM-analyzing in SCRUM methodology the advisability of cybernetics approach applying is shown to represent a RM-scheme in form of multiple-loop feedback control system, where appropriate software metrics for generation of control actions are used. The principles for its further elaboration of models and tools are formulated.

Key words: requirements management, agile-methodologies, software cybernetics, software metrics.

Ткачук Николай Вячеславович – д-р техн. наук, проф., проф. кафедры автоматизированных систем управления, Национальный технический университет «ХПИ», Харьков, Украина, e-mail: tka@kpi.kharkov.ua.

Гамзаев Рустам Александрович – ассистент кафедры автоматизированных систем управления, Национальный технический университет «ХПИ», Харьков, Украина, e-mail: Rustam.Gamzayev@gmail.com.