УДК 512.086

Н. А. БАЖЕНОВ, аспирант НТУ «ХПИ»

МОДЕЛИ УПРАВЛЕНИЯ КАЧЕСТВОМ СБОРА ТРЕБОВАНИЙ К ПРОГРАММНОМУ ОБЕСПЕЧЕНИЮ НА ОСНОВЕ ТЕКСТОВ СПЕЦИФИКАЦИЙ

В статті пропонуються підхід до керування якістю процесу розробки програмного забезпечення, побудований на підвищенні ефективності функціонування одного з етапів життєвого циклу програмного продукту – процесу збору вимог. Були виділені мети функціонування даного етапу, поставлені основні завдання, виявлені механізм взаємодії з іншими етапами життєвого циклу. Запропоновано концепцію керування якістю даного процесу на основі всебічного контролю якості вхідної й вихідної інформації, ресурсів і самого процесу.

В статье предлагается подход к управлению качеством процесса разработки программного обеспечению, основанный на повышении эффективности функционирования одного из этапов жизненного цикла программного продукта – процесса сбора требований. Были выделены цели функционирования данного этапа, поставлены основные задачи, выявлены механизм взаимодействия с другими этапами жизненного цикла. Предложена концепция управления качеством данного процесса на основе всестороннего контроля качества входной и выходной информации, ресурсов и самого процесса.

In the article there has been proposed approach to quality management of software development process, based on the efficiency function increasing of one of the software product life cycle stages – requirements elicitation process. The purposes of functioning of the given stage have been established, the primary goals are put, the mechanism of interaction with other stages of life cycle is revealed. The concept of process quality management on the basis of total quality management of the input and output information, resources and the process is offered.

Введение. В настоящее время процесс разработки программного обеспечения представляет собой целый ряд организационных, технологических и технических групп задач [1, 2]. Эти группы задач формируют собой жизненный цикл (ЖЦ) программной системы (ПС), начиная от замысла создания такой системы, и заканчивая снятием с эксплуатации. Согласно [2], «*Модели жизненного цикла системы обычно разделяют на последовательные периоды реализации – стадии и этапы*». В данной работе предлагается рассмотреть один из таких этапов – процесс сбора требований к программному обеспечению.

В связи с постоянно возрастающей сложностью разрабатываемых программных систем, а также с повышающимися при этом ресурсоёмкостью и функциональными возможностями, на первый план выходит проблема обеспечения качества таких систем. Причём качество конечного продукта зависит не только от достижения определённых характеристик, заранее заданных заказчиком системы, но и от качества самого процесса разработки, состоящего из всех стадий жизненного цикла системы. Предлагается рассмотреть аспекты качества процесса разработки на примере стадии сбора требований к программному обеспечению путём обработки спецификаций требований на естественном языке.

В разделе 1 описывается процесс сбора требований как одна из стадий ЖЦ ПС. Раздел 2 посвящен аспектам управления качеством стадии сбора требований. В разделе 3 рассмотрены основные выходные данные, являющиеся целью функционирования процесса сбора требований, а также возникающие при этом задачи. В 4-м разделе приводится схема функционирования процесса и основные функциональные блоки.

1. Место процесса сбора требований в ЖЦ. В соответствии с комплексом международных стандартов, регламентирующих жизненный цикл программных систем (ISO 12207 [3], ISO 15288 [4], ISO 9126-1 [5], ISO 25030 [6], SWEBOK [7]), процесс сбора требований к программному обеспечению рассматривается как одна из дисциплин процесса разработки ПО (рис. 1).

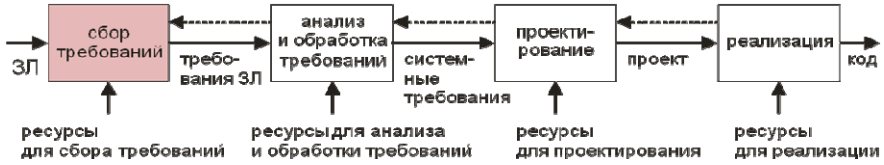

Рис. 1. Процесс сбора требований в ЖЦ

Таким образом, объектом данного исследования является стадия технической группы процессов жизненного цикла на примере процесса сбора требований к программному обеспечению. Целью исследования ставится повышение эффективности управления качеством сбора требований к программному обеспечению путём разработки и исследования моделей и информационных технологий поддержки этапов жизненного цикла программных систем. Цель исследования основывается на комплексном использовании методов компьютерной лингвистики, искусственного интеллекта, теории автоматизированной обработки естественного языка, а также на теории многокритериальной оптимизации и принятия решений.

2. Управление качеством процесса сбора требований. Для управления качеством дисциплины жизненного цикла предлагается использовать процессный подход, или так называемый цикл Дёминга Plan-Do-Check-Act [8]. Осуществляется управление измерениями качества – качеством входа, выхода, ресурсов, необходимых для данного этапа и самим процессом сбора требований (рис. 2).

Таким образом, получаем четыре задачи системной оптимизации для управления качеством входной информации, выходной информации, ресурсов и процесса сбора требований. Выделим критерии контроля качества по этим четырём задачам локальной оптимизации.

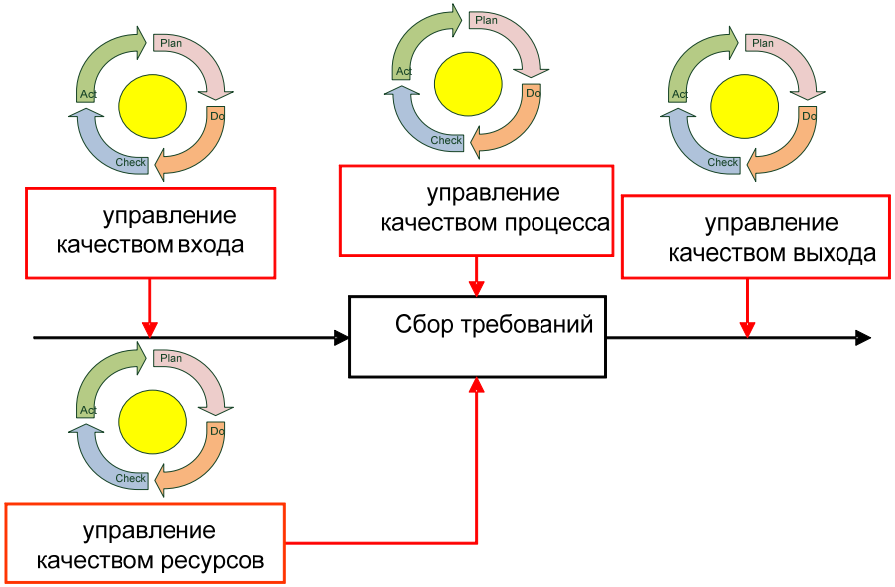

Рис. 2. Процесс сбора требований в ЖЦ

Контроль качества входной информации:

- Качество документов спецификаций требований. Включает в себя контроль качества по специализированным критериям, отсеивание нерелевантных документов, грубую фильтрацию (например, использование фильтров ключевых слов и фраз [9]).

Контроль качества выходной информации:

- Качество получаемой модели требований.
- Синхронизация управление качеством с этапом анализа и обработки требований.
- Поддержка возможности обратной связи.

Контроль качества ресурсов:

- Качество трудовых ресурсов.
- Качество финансовых ресурсов.
- Качество материальных ресурсов.
- Качество информационных ресурсов.

Качество процесса:

- Внутренние характеристики процесса.
- Управление различными вариантами реализации.

3. Цели и задачи процесса сбора требований. На вход дисциплины сбора требований поступают текстовые документы, выражающие спецификации требований к системе, нужды и пожелания заинтересованных лиц (ЗЛ), заказчиков, потребителей. Процесс данной дисциплины должен проанализировать данную неструктурированную информацию и преобразовать её во множество требований ЗЛ, которое отражает взаимодействия между элементами системы, между системой и внешними интерфейсами, и эксплуатационные характеристики системы, удовлетворяющие нуждам ЗЛ.

Связь с дисциплиной анализа и обработки требований осуществляется по данным и обратной связи. В качестве этих данных выступает выходная информация дисциплины сбора требований, которая затем поступает на вход следующей дисциплине жизненного цикла системы. На преобразовании «вход-выход» осуществляется контроль качества информации по заранее заданным соответствующим метрикам. Обратное управляющее и корректирующее воздействие возможно как с последующей дисциплины, так и с остальных дисциплин жизненного цикла – проектирование, реализация. Согласно стандарту ISO 12207[3] выходная информации данной дисциплины должна содержать следующую информацию:

- Требуемые характеристики системы и условия использования сервисов.
- Системные ограничения.
- Соотношения между требованиями потребностями ЗЛ.
- Информация для определения системных требований.
- Информация для проверки конечных сервисов заданным характеристикам.
- Информация по поставке системы.

Так как объектом данного исследования является одна из технических стадий разработки программной системы, то организационные процессы жизненного цикла и соответствующие им аспекты не рассматриваются.

Таким образом, имеем взаимодействие дисциплины с другими этапами жизненного цикла на следующей схеме (рис. 3).

Для достижения необходимых результатов в виде перечисленной выходной информации перед дисциплиной ставятся следующие задачи:

- Идентификация ЗЛ. Определить отдельных ЗЛ или целые классы ЗЛ, вовлеченных в проект, на протяжении всего жизненного цикла.
- Идентификация требований:

1. Извлечь требований ЗЛ, описывающие потребности, желания, ожидания и предполагаемые ограничения, выявленные ЗЛ.
2. Определить системные ограничения, возникающие вследствие существующих соглашений, организационных и технических решений.
3. Определить типовой набор действий для выявления всех необходимых функций, которые соответствуют ожидаемым сценариям и условиям работы системы.

4. Определить возможные взаимодействия между пользователями и системой.

- Оценка требований. Проанализировать полное множество извлеченных требований.
- Согласование требований:

1. Решить проблемы и несогласованности в требованиях.
2. Обеспечить обратную связь с ЗЛ для подтверждения того, что потребности и ожидания были адекватно извлечены и представлены.
3. Предоставлять ЗЛ информацию о том, что их требования выражены корректно.

- Обеспечить фиксацию и запись требований:

1. Производить фиксацию требований ЗЛ в виде пригодном для управления требованиями на протяжении всего жизненного цикла системы и за его пределами.
2. Поддерживать трассируемость требований ЗЛ и исходных источников ЗЛ.

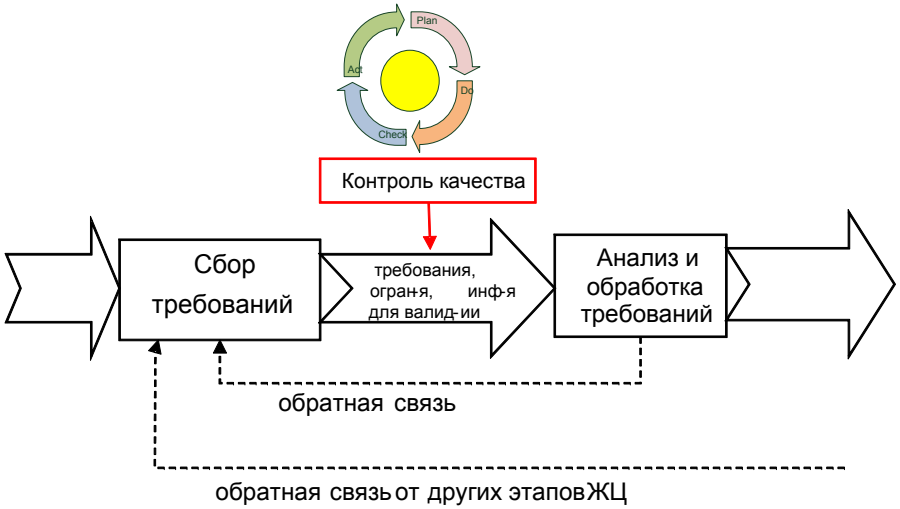

Рис. 3. Взаимодействие процесса сбора требований с другими стадиями ЖЦ

4. Функциональные элементы процесса сбора требований. Исходя из поставленных задач перед процессом сбора требований, можно выделить основные функциональные элементы процесса, а также потоки данных и управляющих воздействий. На рис. 4 представлена общая схема функционирования процесса сбора требований.

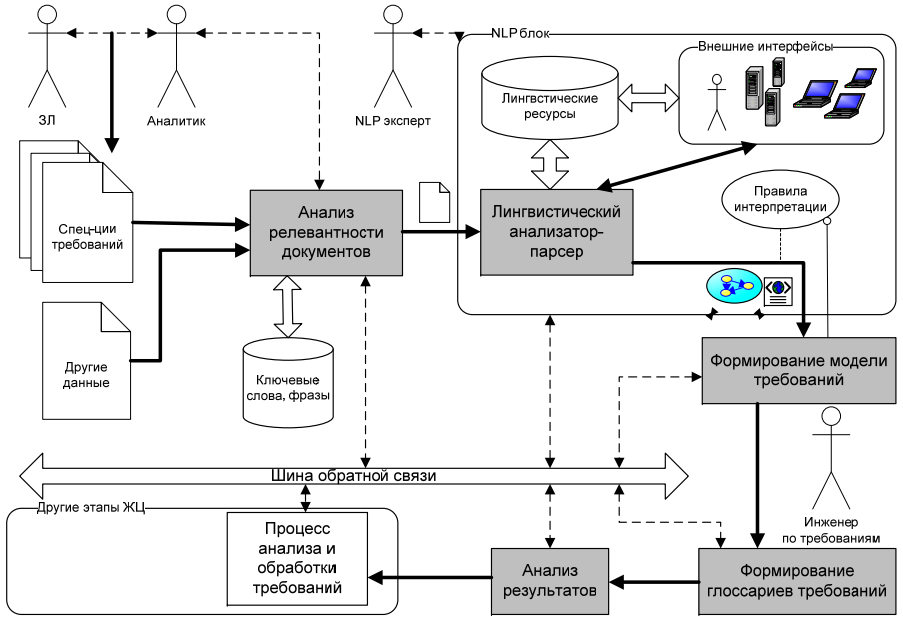

Рис. 4. Общая схема функционирования процесса сбора требований

На данной схеме показаны основные функциональные блоки серыми прямоугольниками, потоки данных сплошными стрелками, управляющие воздействия и обратные связи пунктирными стрелками, ресурсные взаимодействия фигурными стрелками.

На вход подсистемы сбора требований в блок анализа релевантности документов могут поступать следующие данные:

Спецификации требований, полученные в результате взаимодействия ЗЛ и аналитиков.

Другие документы, полученные от ЗЛ, или представляющие интерес для проектирования и создания программной системы.

Обратное взаимодействие от пары «ЗЛ–аналитик» о подтверждении/опровержении решения о релевантности документа/текста.

Управляющие и обратные воздействия от других элементов как процесса, так и ЖЦ в целом.

Анализ релевантности документов и их соответствие доменной специфичности разрабатываемой ПС определяется с учетом имеющихся ресурсов ключевых слов, фраз, и т.д. [9]. На выходе данного блока «фильтра грубой очистки» получается релевантный документ/текст, готовый к лингвистическому анализу.

Блок лингвистического анализа представляет собой конкретную реализацию одним из методов *Автоматизированной Обработки*

Естественного Языка (Natural Language Processing, NLP). Могут быть предложены следующие реализации:

- С помощью теории «Смысл-текст» [10] с использованием с использованием языкового онлайн-процессора «ЭТАП» [11].
- С помощью применения универсального структурированного языка Universal Network Language (UNL) [12] с использованием онлайн базы знаний UNL-KB [13].
- На основе теории контекстно-свободных грамматик с использованием статистического морфологического анализатора и основанного на правилах синтаксического парсера [14-16].
- На основе подхода «Ontological Semantics» [17].

При любой реализации внутри данного блока существует блок лингвистического анализа и парсинга, взаимодействующего с необходимыми ресурсами и внешними интерфейсами, конкретная реализация которых зависит от выбранного метода NLP. Связь с внешними интерфейсами может осуществляться путём взаимного обмена данными. Типичным набором задач лингвистического анализатора является [18]:

- Разбиение текста на отдельные лексемы (tokenization).
- Идентификация границ предложений.
- Морфологический анализ (POS tagging).
- Лемматизация (lemmatization).
- Присвоение всевозможных лингвистических категорий и свойств лексемам.
- Поверхностный синтаксический анализ и фразовая фрагментизация текста (chunking, shallow parsing).
- Идентификация вербальных категорий.
- Идентификация семантических ролей и связей.

Результатом работы данного блока является структурированный текст требований древовидного формата с корневой вершиной «текст», промежуточными узлами, характеризующими предложения и различные фразы, и с листьями, представленными отдельными лексемами с наборами присвоенных атрибутов.

Данный формат данных передаётся на блок формирования модели требований. Для сопоставления лингвистических элементов древовидного текста с конкретными концептами модели требований существует правила интерпретации (interpretation rules). Реализация конкретных правил зависит от выбора метода NLP, поэтому на схеме функционирования данный объект расположен в логической группе NLP блока. После формирования модели требований происходит заполнение глоссариев концептов требований по данной модели, в которой учитываются категории ЗЛ, множество их требований, выявленные системные ограничения, и другие данные.

Завершающим блок процесса сбора требований является анализ полученных результатов. Полученные глоссарии проверяются на предмет неопределенности, неполноты, противоречивости и других несоответствий. Отметим, что данная проверка является первичной, так как более глубокий анализ осуществляется на последующих этапах ЖЦ.

Все функциональные блоки имеют управляющее взаимодействия с т.н. «шиной обратной связи», по средствам которой осуществляется возврат на предыдущие этапы с соответствующим управляющим воздействием, и корректировочные настройки функционирования этапов.

Выводы. Таким образом, рассмотренный процесс сбора требований к программному обеспечению является одним из важнейших этапов ЖЦ ПС. И для обеспечения качества ПС необходимо осуществлять управление качеством на различных элементах этого процесса путём оценивания отклонения от эталонных характеристик. Реализацию блока лингвистического анализа следует выбирать с учётом удовлетворения необходимых показателей качества функционирования всего процесса сбора требований.

Список литературы: ***1.*** *Андон Ф. И., Коваль Г. И., Коротун Т. М. и др.* Основы инженерии качества программных систем.– К.: Академпериодика, 2007.– 680 с. ***2.*** *Липаев В. В.* Процессы и стандарты жизненного цикла сложных программных средств.– М.: СИНТЕГ, 2006.– 276 с. ***3.*** ISO/IEC Standard 12207:2008 System and software engineering – Software life cycle processes standard. ***4.*** ISO/IEC 15288:2008 Systems and software engineering – System life cycle processes. ***5.*** ISO/IEC 9126-1:2001 Software engineering – Product quality – Part 1: Quality model. ***6.*** ISO/IEC 25030:2007 Software engineering – Software product Quality Requirements and Evaluation (SQuaRE) – Quality requirements. ***7.*** ISO/IEC 19759:2005 Software Engineering – Guide to the Software Engineering Body of Knowledge (SWEBOK). ***8.*** *Репин В. В., Елиферов В. Г.* Процессный подход к управлению. Моделирование бизнес-процессов. – М.: РИА «Стандарты и качество», 2004. ***9.*** *Perkonigg M.* Linguistische Aspekte des Attempto Controlled English (ACE). Masterarbeit.– Klagenfurt: Alpen-Adria-Universität Klagenfurt, 2009. ***10.*** *Мельчук И.* Опыт теории лингвистических моделей Смысл-Текст.– М.: Наука, 1974. ***11.*** *Apresian J., Boguslavsky I., Iomdin L. etc.* ETAP-3 Linguistic Processor.– Moscow: Laboratory of Computational Linguistics Institute for Information Transmission Problems, 2002. ***12.*** *J. Cardeñosa, A. Gelbukh, E. Tovar (Eds.).* Universal Network Language: Advances in Theory and Applications – Research on Computing Science 12, 2005. ***13.*** *H. Uchida, M. Zhu.* UNL2005 for Providing Knowledge Infrastructure.– Chiba, Japan: SeC2005 Workshop 2005. ***14.*** *Баженов М. О.* Розробка алгоритмічного та програмного забезпечення обробки текстів на природній мові для підвищення ефективності застосування систем управління вимогами. Дипломна робота освітньо-кваліфікаційного рівня магістра.– Х.: НТУ «ХПІ», 2008.– 134 с. ***15.*** *Georg Weber.* NIBA<tag> Aspekte der Implementierung eines erweiterten Taggers für die automatische Textannotation in NIBA: Masterarbeit.– Alpen-Adria-Universität Klagenfurt, 2007. ***16.*** *Günther Fliedl, Christian Kop, Jürgen Vöhringer.* NIBA: Project & Toolset.– Alpen-Adria-Universität Klagenfurt, 2006. ***17.*** *Sergei Nirenburg, Victor Raskin.*– The MIT Press, 2004. ***18.*** *Christopher D. Manning.* Foundations of statistical natural language processing.– The MIT Press, 1999.

Поступила в редколлегию 15.12.09