САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

На правах рукописи

Романовский Константин Юрьевич

МЕТОД ПОВТОРНОГО ИСПОЛЬЗОВАНИЯ ДОКУМЕНТАЦИИ СЕМЕЙСТВ ПРОГРАММНЫХ ПРОДУКТОВ

05.13.11 – Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей

АВТОРЕФЕРАТ
диссертации на соискание ученой степени
кандидата физико-математических наук

Санкт-Петербург
2010

Работа выполнена на кафедре системного программирования математико-механического факультета Санкт-Петербургского государственного университета.

Научный руководитель: кандидат физико-математических наук, доцент
КОЗНОВ Дмитрий Владимирович

Официальные оппоненты: доктор технических наук, профессор,
ЛИСС Александр Рудольфович
(ОАО «Концерн «Океанприбор»)

кандидат технических наук, профессор
КОТЛЯРОВ Всеволод Павлович
(Санкт-Петербургский государственный политехнический университет)

Ведущая организация: Московский государственный университет приборостроения и информатики

Защита диссертации состоится "15" апреля 2010 г. в 14:00 часов на заседании совета Д 212.232.51 по защите докторских и кандидатских диссертаций при Санкт-Петербургском государственном университете по адресу: 198504, Санкт-Петербург, Петродворец, Университетский пр., 28, математико-механический факультет, ауд. 405.

С диссертацией можно ознакомиться в Научной библиотеке Санкт-Петербургского государственного университета по адресу: 199034, Санкт-Петербург, Университетская наб., д. 7/9.

Автореферат разослан "___" _____ 2010 года

Ученый секретарь диссертационного совета,
доктор физико-математических наук,
профессор Даугавет И.К.

Общая характеристика работы

Актуальность проблемы

Разработка семейств программных продуктов (СПП, Product Lines) – это промышленный подход разработки ПО, обеспечивающий эффективную совместную разработку и продвижение на рынке группы программных продуктов сходного назначения. Суть подхода заключается в плановом повторном использовании различных активов разработки, таких как исходный код, тесты, документация, инструменты разработки, организационные процедуры и т. д. Еще Парнас заметил, что создавать линейки продуктов часто бывает целесообразнее и экономичнее, чем разрабатывать отдельные продукты. В конце 90-х годов XX века сформировались два научных центра, вокруг которых сосредоточены основные исследования в области разработки СПП – Институт Программной Инженерии[1] Университета Карнеги-Мелон в США и Европейский Институт Программного Обеспечения[2] в Европе. Ежегодно проводится несколько международных конференций, посвященных вопросам разработки СПП, издано несколько сотен научных статей по этой тематике, ведутся десятки исследовательских проектов. Многие крупные компании, такие как Nokia, Hewlett Packard, Motorola и др. успешно используют методы разработки СПП на практике.

Существующие на сегодняшний день подходы в этой области обеспечивают повторное использование различных активов разработки, главным образом кода, тестов, организационных процедур. В то же время для полноценной промышленной разработки СПП требуется в дополнении к работающему ПО выпускать также и документацию. Документация СПП может иметь много общего – одноименные документы для разных продуктов семейства (например, руководство пользователя), описывая одинаковую, общую для разных продуктов функциональность, очевидно, содержат многочисленные повторы. Для эффективной разработки и эволюции подобной документации требуются средства поддержки повторного использования повторяющихся фрагментов, что позволяет сократить сроки разработки и существенно повысить качество документации. Однако в области разработки СПП отсутствуют методики разработки документации, ориентированные на повторное использование, а существующие методы и средства разработки документации, такие как DITA, DocBook,

[1] SEI – Software Engineering Institute, http://www.sei.cmu.edu.

[2] ESI – European Software Institute, http://www.esi.es.

FrameMaker, поддерживают возможности повторного использования, но не позволяют гибко настраивать переиспользуемые фрагменты текста под особенности конкретного продукта и слабо поддерживают средства визуального планирования структуры сложных пакетов документации. Все это существенно затрудняет их применение при разработке документации для СПП.

Таким образом, создание формального метода разработки документации СПП, поддерживающего визуальное проектирование документации и вариативное (т.е. допускающее модификации в зависимости от контекста использования) повторное использование – актуальная на сегодняшний день задача, решение которой позволит увеличить эффективность разработки промышленных СПП.

Цель диссертационной работы

Целью работы является создание формального метода разработки документации СПП, обеспечивающего поддержку проектирования документации с вариативным повторным использованием.

Научно-методическая база исследования

Данное исследование проводилось на стыке двух областей: первая область – это методы повторного использования и разработки СПП, вторая область – это подходы и средства создания технической документации. В работе использовался универсальный метод повторного использования Бассета-Ерзабека [10, 16], который адаптируется для разработки документации, и подход для анализа и проектирования общих и различающихся свойств продуктов СПП – диаграммы возможностей (Feature Diagrams) [13], – а также общие концепции визуального моделирования ПО [8]. В работе активно использовался подход единого исходного представления пакетов документов (Single Sourcing) и идея использования XML для внутреннего представления документации [17] .

Научная новизна

На сегодняшний день существует ряд методов разработки СПП [8, 10, 12, 15], однако в них отсутствует поддержка разработки документации. С другой стороны, имеется ряд методов и средств разработки документации [14, 18], часть из которых в той или иной степени поддерживают повторное использование. Но эти подходы не обеспечивают поддержку вариативности общих активов, т.е. возможности модифицировать их по требованиям конкретного продукта. Поддержка вариативности хорошо

проработана в методе фреймов Бассета-Ерзабека [10, 16], который, однако, изначально ориентирован на поддержку повторного использования программного кода и не применялся для задач разработки документации. В данной диссертационной работе ставится задача восполнить пробел между методами разработки СПП и подходами разработки документации. В результате исследования был предложен метод разработки технической документации DocLine, включающий в себя оригинальный язык разработки документации DRL, процесс разработки документации, а также инструментальный пакет. Язык DRL, в дополнение к традиционному XML-представлению документации, предусматривает также и визуальную нотацию, предназначенную для проектирования и изучения схемы повторного использования документации. Визуальная нотация использует концепцию нескольких взглядов на разрабатываемую систему[3], которая не использовалась при разработке документации – существующие XML-средства разработки технической документации (DITA, DocBook и др.) поддерживают только простые визуализаторы XML-структуры. Предложенный процесс разработки документации включает в себя рефакторинг документации – это достаточно традиционный инструмент в разработке ПО и, в частности, СПП, однако в области разработки документации рефакторинг пока не использовался.

Практическая ценность

В рамках работы был разработан пакет инструментальных средств, который интегрирован в широко распространенную в промышленном программировании среду разработки приложений Eclipse IDE[4], что позволяет эффективно использовать метод в коммерческих проектах разработки СПП. При этом инструментальные средства предлагаются с открытым исходным кодом и опираются на свободно-распространяемое ПО, что позволяет легко расширять их под особенности конкретных задач.

Апробация работы и публикации

Результаты диссертации докладывались на международной конференции по методам системного программирования (3rd IFIP TC2 Central and East European Conference on Software Engineering Techniques CEE-SET 2008, Брно, Чехия), на семинарах Института системного программирования Российской академии наук, Московского госу-

[3] Данная концепция широко используется в программной инженерии – например, она лежит в основе языка визуального моделирования UML.

[4] http://www.eclipse.org.

дарственного университета печати, кафедры системного программирования и НИИ ИТ математико-механического факультета СПбГУ.

Предложенный в работе метод был апробирован на документации двух промышленных семейств программных продуктов, результаты апробации опубликованы в работах [3, 4]. Первая апробация была выполнена при разработке документации семейства программно-аппаратных систем управления телевизионным вещанием (ООО «Фирма «ДИП», несколько продуктов, объем документации около 50 страниц). Документация была успешно создана с помощью DocLine, кроме того, был разработан и опробован механизм расширения DocLine по требованиям конкретного проекта – была реализована возможность автоматизированного включения в документацию и обновления иллюстраций, выполненных в Microsoft Visio. Более масштабная апробация была выполнена на документации промышленного семейства телекоммуникационных систем, разрабатываемых ЗАО «Ланит-Терком», – было переработано около 300 страниц документации и выделено 38 повторно-используемых фрагментов текста, создано 40 точек расширения.

Основные результаты диссертации изложены в семи научных работах [1-7]. Из них три работы [1-3] опубликованы в журналах из перечня ВАК. Работы [1,2,4,6,7] написаны в соавторстве.

В работе [1] Романовскому К.Ю. принадлежат основные идеи языка DRL, а также им выполнена формальная спецификация DRL. Кознов Д.В. предложил идею использовать визуальное моделирование для планирования повторного использования документации. В работе [2] Романовскому К.Ю. принадлежит идея идеального (сверху-вниз) процесса разработки документации и архитектура пакета инструментальных средств, а также разработка и формализация описания метода в целом. Ему же принадлежит разработка инструментальных средств. Кознов Д. В. предложил идею «легковесного» («гибкого», снизу-вверх) процесса разработки документации. В работе [4] Романовский К.Ю. написал раздел «Архитектура средств автоматизации». В работе [6] Романовский К.Ю. разработал и формально специфицировал операции рефакторинга, спроектировал средства их программной поддержки, а также разработал и описал примеры. Кознов Д.В. предложил идею рефакторинга документации. Минчин Л. реализовал операции рефакторинга в составе пакета DocLine. В работе [7] Романовский К.Ю. написал основной текст статьи и добавил новые операции рефакторинга. Кознов Д.В. предложил уточнение «гибкой» модели разработки документации.

Исследование поддержано Российским Фондом Фундаментальных Исследований (гранты РФФИ 08-01-00716-а, 08-07-08066-з). Разработка инструментальных средств поддержана Фондом содействия развитию малых форм предприятий в научно-технической сфере (программа СТАРТ).

На пакет программных средств получено свидетельство о государственной регистрации программы для ЭВМ №2008612676, выданное 28 апреля 2008 г. Федеральной службой по интеллектуальной собственности, патентам и товарным знакам.

Структура и объем диссертации

Диссертация состоит из введения, шести глав (со сквозной нумерацией разделов, рисунков и таблиц), заключения, списка литературы и приложения. Текст диссертации изложен на 110 страницах и содержит 20 рисунков и 2 таблицы. Список литературы содержит 58 наименований.

Содержание работы

Во **введении** показывается актуальность выбранной темы исследований, излагается научный контекст диссертационной работы, а также приводится краткое описание предложенного в работе метода DocLine. DocLine охватывает весь жизненный цикл разработки документации от проектирования до публикации итоговых документов и поддерживает плановое адаптивное повторное использование документации. В DocLine, следуя сложившейся традиции, явно выделяется исходное и целевое представление документации. Целевое представление – это документы в привычных для пользователя форматах, таких как PDF для печатных документов, HTML для электронных, или HTML Help для справочной системы. Целевое представление может быть в любой момент получено из исходного автоматически – эта операция называется публикацией (наподобие получения исполняемого кода ПО из исходного кода с помощью компиляции). Для исходного представления документации в DocLine предлагается оригинальный проблемно-ориентированный язык DRL (Documentation Reuse Language). Язык DRL имеет две нотации – графическую (DRL/GR – Graphic Representation) и текстовую (DRL/PR – Phrase Representation). Графическое представление служит для проектирования структуры повторного использования документации (т.е. планирования повторного использования). Текстовое представление позволяет описать в виде XML-представления варианты конфигурирования повторно используемых компонент и конкретные конфигурации для порождения конечных документов. DRL/PR интегрирован с известным форматом DocBook для реализации фор-

матирования текста. В работе реализована поддержка двух целевых форматов – PDF и HTML.

Помимо языка метод DocLine также определяет эталонные модели процесса разработки документации: проактивную и «гибкую». В проактивном процессе сначала проводится проектирование схемы повторного использования и разработка адаптивных повторно-используемых компонент, а затем на основе созданной инфраструктуры создаются пакеты документов для конкретных продуктов. В «гибком» процессе сначала создаются требуемые документы, а затем по мере необходимости документация подвергается рефакторингу.

Для практического применения DocLine реализован пакет инструментальных средств, встроенный в интегрированную среду разработки приложений Eclipse.

В **первой** главе описан контекст исследований и существующие подходы разработки СПП и документации, в том числе подходы, на которые опирается DocLine. В результате обзора делаются следующие выводы. Существует ряд зрелых подходов к разработке документации, часть из которых поддерживает повторное использование (DITA [14], DocBook [18], FrameMaker [17]). Также есть ряд общих методов повторного использования, в которых хорошо проработана адаптивность, но нет возможностей, нужных для разработки документации (подход Бассета-Ерзабека [10, 16]). Имеются и методы моделирования общих и различных свойств систем, которые могут быть применены к задаче разработки документации (диаграммы возможностей [13]). Однако нет единого метода разработки документации, поддерживающего проектирование сложной документации и адаптивность повторного использования. Для документации СПП эти аспекты являются ключевыми.

С другой стороны, имеющиеся методы и средства могут дать техническую базу для нового метода и позволяют сосредоточить основные усилия на главном – на поддержке адаптивного повторного использования документации. Так при реализации DocLine использовались готовые открытые технологии – DocBook для описания форматирования текстов и публикации документов в различных целевых форматах и Eclipse GMF для реализации графических средств проектирования структуры сложных пакетов документации.

Во **второй** главе описывается язык DRL. Графическая нотация DRL/GR определяет три вида диаграмм – главную диаграмму, диаграмму вариативности и диаграмму продукта. Главная диаграмма (Рис. 1) определяет состав продуктов семейства и набор *информационных продуктов*, представляющих собой шаблоны целевых документов,

таких как руководство пользователя или справочник. Здесь и далее в качестве примеров приводятся фрагменты описания телекоммуникационных систем, подготовленные в ходе апробации DocLine в ЗАО «Ланит-Терком».

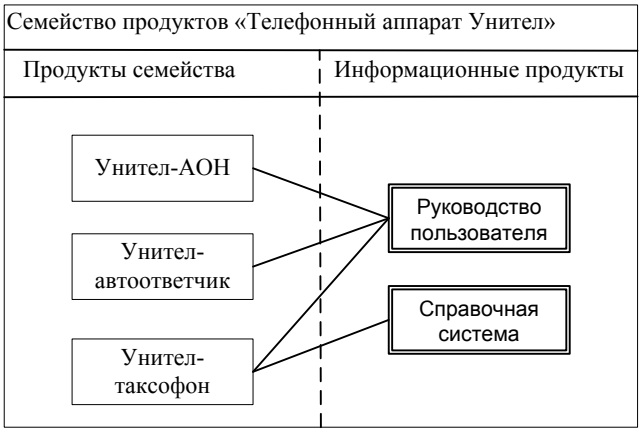

Рис. 1. Главная диаграмма.

Информационные продукты состоят из *информационных элементов*, представляющих собой контекстно-независимые повторно-используемые фрагменты текста. Каждый информационный элемент также может включать другие информационные элементы. Эти включения могут быть обязательными или не обязательными, а также их можно объединять в группы двух видов: OR-группы, допускающие включение любого набора элементов из группы (такие группы используются для перечисления «опций», которые могут в любом сочетании войти в разные продукты, например описание входящих и исходящих вызовов), и XOR-группы, допускающие включение не более чем одного элемента (применяются для объединения взаимоисключающих вариантов, например стандарта определителя номера – европейский Caller ID или российский ГОСТ). Подобные варианты сочетания повторно-используемых блоков задаются на диаграмме вариативности (см. Рис. 2). Фактически, диаграмма вариативности описывает допустимые варианты адаптации общих активов в соответствии с требованиями контекста на уровне блоков текста (информационных элементов).

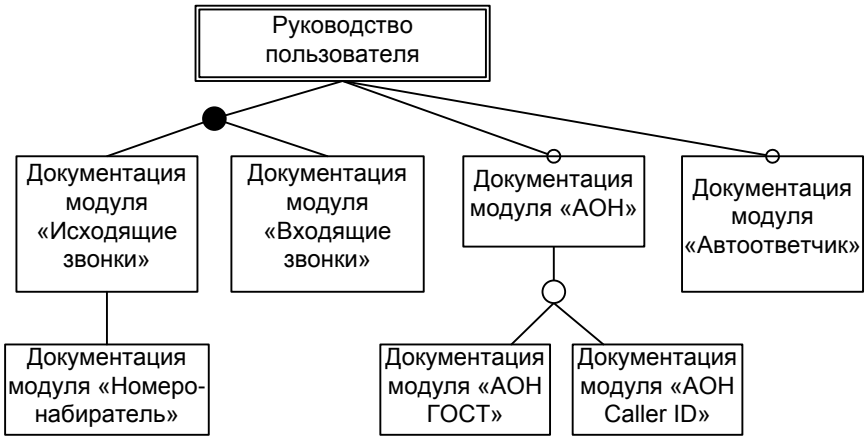

Рис. 2. Диаграмма вариативности.

Для более детальной настройки общих активов в DRL/PR предлагается механизм конфигурирования информационных элементов. Для того, чтобы была возможность конфигурировать информационный элемент в соответствии с особенностями конкретного контекста использования, в нем необходимо предусмотреть точки расширения в тех местах, в которых предполагается вариативность. Пример информационного элемента в нотации DRL/PR, содержащего точку расширения, представлен ниже.

<***infelement*** *id=CallerIdent>После получения входящего вызова телефон запрашивает в сети информацию о звонящем абоненте и затем*

<***nest*** *id=AONOptions>отображает на экране номер абонента*</***nest***>.

</***infelement***>

XML-теги <***infelement/***> и <***nest/***> задают информационный элемент и точку расширения соответственно.

Для упорядочивания терминологии в документах DRL/PR предлагает конструкцию *словарь*, включающую в себя набор пар (имя, значение). В произвольной точке документа можно подставить значение элемента словаря, указав его имя. В словарь могут быть включены такие элементы, как название продукта, версия, набор поддерживаемых операционных систем и т.п. – все они много раз используются в тексте и склонны меняться. Также DRL/PR предлагает обобщение словарей – каталоги, которые предназначены для унификации представления в тексте однородных элементов, более сложных, чем термины. *Каталог* – это набор элементов-кортежей (имя, атрибут 1, атрибут 2, ...). Так, каталог команд пользовательского интерфейса может содержать коллекцию команд интерфейса, имеющих имя, пиктограмму, описание, «горячую клавишу», текст всплывающей подсказки, и т.д.

Для преобразования переиспользуемой документации в документ для конкретного продукта в DRL/GR имеются диаграммы продукта, позволяющие описать, какие именно информационные элементы включаются в тот или иной продукт. На Рис. 3 (а) и (б) представлены две диаграммы продуктов для руководства пользователя, описанного на Рис. 2.

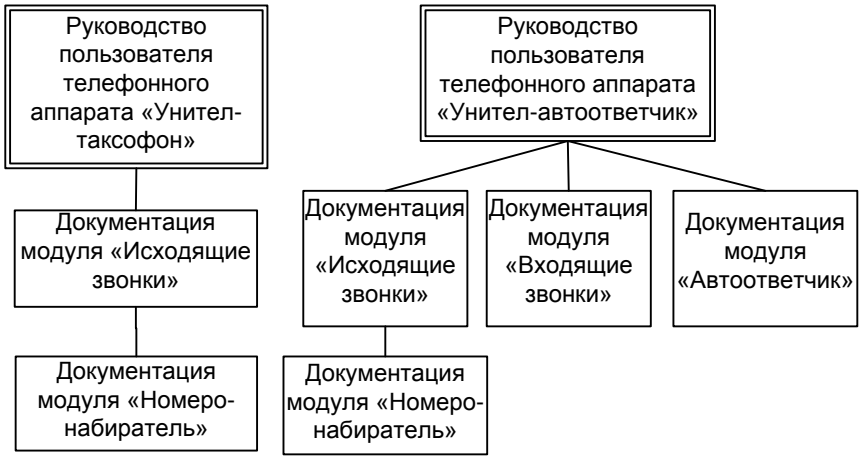

Рис. 3. Диаграмма продукта.

Для настройки конфигурируемых информационных элементов служит специальная конструкция DRL/PR – *адаптер*. Адаптер специфицирует набор изменений, которые необходимо внести в информационный элемент, чтобы получить требуемый для конкретного контекста текст. Изменения могут вноситься только в предусмотренные точки расширения, с точками расширения допустимы следующие операции: замена содержимого, добавление текста перед/после точки расширения. Пример адаптера в нотации DRL/PR приведен ниже.

<***adapter*** *infelemrefid="CallerIdentRef”>*

<***replace-nest*** *nestid=AONOptions>проговаривает номер абонента*</***replace-nest***>

</***adapter***>

Тег **<*adapter/*>** задает сам адаптер, тег <***replace-nest/***> задает изменение типа «вставить вместо».

В **третьей** главе описывается процесс разработки документации, а также определяется процесс рефакторинга документации и описываются типовые операции рефакторинга. В работе технических писателей можно выделить два вида деятельности – разработку повторно-используемых активов документации и разработку документации конкретных продуктов. В проактивном варианте процесса создание документации начинается с разработки повторно-используемой документации (см. Рис. 4).

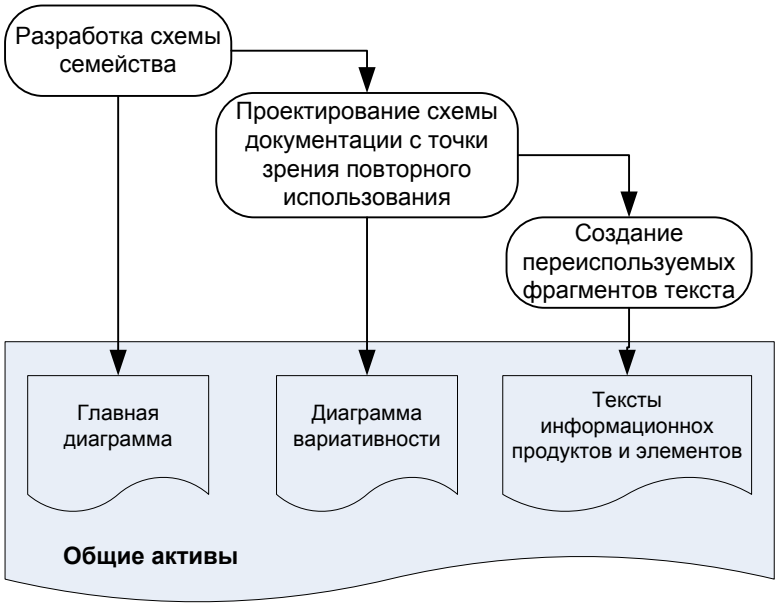

Рис. 4. Схема процесса разработки повторно-используемой документации.

Когда общие активы документации созданы, на их основе порождается документация конкретных продуктов (см. Рис. 5). В случае, когда разрабатывается документация первого продукта, процесс имеет линейную структуру – дорабатываются необходимые общие активы и создается документация продукта. При добавлении к семейству последующих продуктов и модификации общих активов потребуются дополнительные действия – ведь от добавления нового документа существующие не должны меняться. Это означает, что вместе с модификацией общих активов необходимо внести соответствующие изменения в документацию существующих продуктов, при этом существующая документация (точнее, ее конечное, видимое читателями представление – PDF, HTML и т.д.) не должна измениться, в то время как ее внутреннее DRL-представление претерпевает значительные изменения. Такое изменение общих активов является *рефакторингом* документации.

Рефакторинг встраивается также и в «гибкий» вариант процесса разработки ПО. Согласно этому процессу сначала разрабатывается документация для первого продукта без учета повторного использования. Это соответствует «гибкому» подходу к разработке СПП, когда в начале делается необходимый минимум инфраструктурной работы и основное внимание уделяется тому, что нужно заказчику ПО. Далее, в процессе разработки документации следующих продуктов из существующей документации выделяются общие активы и на их основе создаются новые документы. При этом существующая документация подвергается рефакторингу, т.е. сама документация перестраивается для использования общих активов при сохранении ее конечного вида.

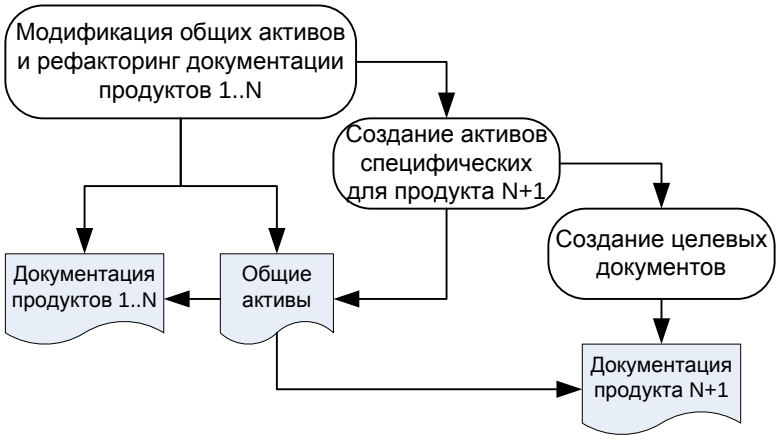

Рис. 5. Схема процесса разработки документации продуктов.

Для облегчения рефакторинга документации DocLine предлагает ряд типовых автоматизированных операций рефакторинга, предназначенных для создания общих активов и их настройки, а также вспомогательные операции, такие как автоматизированные переименования различных вхождений одной и той же конструкции.

В **четвертой** главе описывается архитектура инструментального пакета, а также особенности его реализации. Пакет состоит из следующих компонент.

- Редакторы.
  - Графический редактор DRL/GR.
  - Текстовый XML-редактор DRL/PR, включающий систему автоматизированного рефакторинга документации.
  - Механизм циклической разработки, поддерживающий целостность различных представлений документации.
- Менеджер публикации.
  - Модуль трансляции, преобразующий DRL-тексты в формат DocBook.
  - Модуль диагностики ошибок, выполняющий проверку корректности текстов на DRL, в том числе проверку корректности генерируемого DocBook-кода средствами пакета DocBook.
- Модуль сопряжения с DocBook, выполняющий генерацию конечных документов в форматах HTML и PDF основываясь на инструментарии DocBook.

Первая версия пакета была реализована на платформе Microsoft.NET в виде отдельного приложения (менеджер публикации и модуль сопряжения с DocBook). Далее была создана архитектура пакета для платформы Java/Eclipse: графический редактор DRL/GR был разработан с помощью технологии Eclipse GMF, а текстовый редактор базируется на стандартном открытом XML-редакторе Eclipse. Стоит также отметить алгоритм валидации документации, состоящий из трех этапов – проверки по схеме DRL, контроля ссылочной целостности и проверки по схеме DocBook.

В **пятой** главе описывается апробация предложенного метода. Результаты апробации обсуждаются и делаются выводы о применимости метода. В целом выяснилось, что применение DocLine при наличии модели вариативности СПП, практически, ничем не отличается по трудоемкости от применения, например, DocBook, однако при этом DocLine добавляет средства планирования и механизм адаптивного повторного использования. Основная сложность применения DocLine – необходимость техническому писателю владеть основами XML. Основные вопросы о возможности практического применения DocLine относятся к применимости подобных XML-технологий в принципе. Однако в западных университетах уже готовят технических писателей (специальность «технические коммуникации», technical writing), которые изучают XML-технологии. Такие XML-технологии как DocBook и DITA начинают активно применяться в индустрии, есть также сведения об их применении и в России. Кроме того, в России распространена практика использования языка TeX для верстки печатных изданий, а TeX намного сложнее, чем DocLine.

Затраты на внедрение DocLine оправданы, если требуется разработать ряд похожих и активно эволюционизирующих (дорабатываемых и исправляемых) документов. В такой ситуации DocLine позволяет уменьшить количество точечных изменений текста за счет однократного изменения повторно используемых активов. В том случае, когда документов мало и/или они статичны (разрабатываются один раз и навсегда), или же нет высоких требований к качеству документации, применение DocLine не даст преимуществ перед DocBook, однако сохранит возможность организации адаптивного повторного использования документации в будущем, если на каком-либо этапе развития СПП ситуация изменится.

В **шестой** главе приводится анализ новизны диссертационной работы.

В **заключении** сформулированы следующие основные **результаты работы**.

- Создан и формально специфицирован новый язык разработки документации DRL, включающий возможности визуального проектирования документации, средства для задания адаптивного повторного использования фрагментов документации в XML-формате, средства задания форматирования текста (интеграция с форматом DocBook).
- Предложены две эталонные модели процесса разработки документации СПП: проактивная («сверху-вниз») и «гибкая» («снизу-вверх»).
- Разработаны операции рефакторинга документации, позволяющие создавать и настраивать общие активы в процессе эволюции документации.

- Предложена архитектура пакета инструментальных средств для разработки документации СПП, выполнена реализация пакета на платформах Microsoft.NET (первая версия) и Java/Eclipse (вторая версия).
- Проведена апробация метода и инструментальных средств на документации реальных промышленных продуктов: .NET-версия применена для разработки документации семейства телевещательных систем (ООО «Фирма ДИП»), Eclipse-версия – для разработки документации ПО семейства телефонных станций (ЗАО «Ланит-Терком»).

В **Приложении** представлено формальное описание синтаксиса языка DRL.

Публикации по теме диссертации из перечня ВАК

1. Романовский К.Ю., Кознов Д.В. Язык DRL для проектирования и разработки документации семейств программных продуктов // Вестн. С.-Петерб. ун-та. Сер. 10: Прикладная математика, информатика, процессы управления. 2007. Вып. 4. С. 110–122.
2. Кознов Д.В., Романовский К.Ю. DocLine: метод разработки документации семейств программных продуктов // Программирование. 2008. №4. С. 1–13.
3. Романовский К.Ю. Разработка повторно-используемой документации семейства телефонных станций средствами технологии DocLine // Вестн. С.-Петерб. ун-та. Сер. 10: Прикладная математика, информатика, процессы управления. 2009. Вып. 2. С. 166–180.

Остальные публикации по теме диссертации

4. Кознов Д.В., Перегудов А.Ф., Романовский К.Ю., Кашин А, Тимофеев А. Опыт использования UML при создании технической документации // Системное программирование. Вып. 1. Сб. статей / Под ред. А.Н. Терехова, Д.Ю. Булычева. СПб.: Изд-во СПбГУ, 2005. С. 18–36.
5. Романовский К.Ю. Метод разработки документации семейств программных продуктов // Системное программирование. Вып. 2 / Под ред. А. Н. Терехова, Д. Ю. Булычева. СПб.: Изд-во С.-Петерб. ун-та, 2007. С. 191–218.
6. K. Romanovsky, D. Koznov, L. Minchin: Refactoring the Documentation of Software Product Lines // Proceedings of 3rd IFIP TC2 Central and East European Conference on Software Engineering Techniques CEE-SET 2008, Brno (Czech Republic), October 13-15, 2008. P. 157-170.

7. Кознов Д.В., Романовский К.Ю. Автоматизированный рефакторинг документации семейств программных продуктов // Системное программирование. Вып. 4 / Под ред. А. Н. Терехова, Д. Ю. Булычева. СПб.: Изд-во С.-Петерб. ун-та, 2009. С. 128–150.

Цитируемая литература

8. Кознов, Д.В. Основы визуального моделирования. М.: Изд-во Интернет-университета информационных технологий, ИНТУИТ.ру, БИНОМ, Лаборатория знаний. 2008. 248 с.
9. Atkinson C., Bayer J., Bunse C., et al. Component-Based Product Line Engineering with UML. Addison-Wesley Professional, 2001. 464 p.
10. Bassett, P. Framing software reuse – lessons from real world. Yourdon Press, Prentice Hall, 1997.
11. Bayer J., DeBaud, J.M., Flege, O., Knauber, P., Laqua, R., Muthig, D., Schmid, K. and Widen, T. PuLSE: A Methodology to Develop Software Product Lines // Proc. Symposium on Software Reusability, SSR'99, Los Angeles, May 1999. P. 122–132.
12. Clements, P., Northrop, L. Software Product Lines: Practices and Patterns. Boston, MA: Addison-Wesley, 2002. 608 p.
13. Czarnecki K., Eisenecker U., Generative Programming: Methods, Tools, and Applications. Reading, Mass.: Addison Wesley Longman, 2000. 864 p.
14. Day, D., Priestley, M., Schell, David A. Introduction to the Darwin Information Typing Architecture – Toward portable technical information // http://www-106.ibm.com/developerworks/xml/library/x-dita1/
15. Greenfield J., Short K., Cook S., Kent S. Software Factories: Assembling Applications with Patterns, Models, Frameworks, and Tools. Indianapolis, Indiana: Wiley Publishing, Inc., 2004. P. 696.
16. Jarzabek, S.; Bassett, P.; Hongyu Zhang; Weishan Zhang. XVCL: XML-based variant configuration language // 25th International Conference on Software Engineering, 2003. Proceedings. 3-10 May 2003. P. 810–811.
17. Marques. M. Single-sourcing with FrameMaker // TECHWR-L Magazine Online, http://www.techwr-l.com/techwhirl/magazine/technical/singlesourcing.html
18. Walsh N., Muellner L. DocBook: The Definitive Guide. O'Reilly, 2003.

Подписано к печати 05.03.10. Формат бумаги 60 х84 1/16.
Бумага офсетная.Гарнитура Таймс. Печать цифровая.Печ. л. 1,0.
Тираж 100 экз. Заказ 4628.
Отпечатано в Отделе оперативной полиграфии Химического факультета СПбГУ
198504, Санкт-Петербург, Старый Петергоф, Университетский пр., 26.
Тел.: (812) 428-40-43, 428-69-19