**Санкт-Петербургский государственный университет**
**Математико-механический факультет**

*Кафедра информатики*

# Поддержка вариативности в семействах Web-приложений интенсивной обработки данных

Дипломная работа

Головина Валерия Геннадьевна
542 группа

«Допустить к защите»
Зав. кафедрой:
д.ф.-м.н., профессор Косовский Н. К.

Научный руководитель:

к.ф.-м.н., доцент, зав. лаб. CASE-технологий
мат.-мех. факультета СПбГУ Кознов Д.В.

Рецензент:

ст. преп. Смирнов М. Н.

Санкт-Петербург
2011

**St. Petersburg state university**
**Faculty of Mathematics and Mechanics**

*Chair of Computer science*

# Software development in the problem of simultaneous software and related documentation development on the basis of the product lines

Graduate paper

Golovina Valeria
542 group

"Approved by"
Head of the chair:

PhD, Professor N. Kosovskyi

Scientific advisor:

PhD, Associate Professor, Head of the CASE- technology laboratory
at the faculty of Mathematics and Mechanics SPbSU D. Koznov

Reviewer:

Senior Lecturer M. Smirnov

Saint-Petersburg
2011

# Оглавление

# Введение

В связи с активной разработкой разнообразных программных продуктов на сегодняшний день очень актуальна идея повторного использования кода. При разработке ряда приложений, имеющих некоторую общую функциональность, такой подход дает возможность переиспользовать общие компоненты, а не создавать множество похожих приложений с нуля. Это позволяет повысить качество кода и сэкономить ресурсы. Одним из методов, реализующих данный подход, является метод создания приложений на основе семейства программных продуктов, предложенный еще 1976 году Дэвидом Парнасом (Parnas, 1976) и активно развивающимся в последние 15 лет. Семейство программных продуктов (software product lines) – это набор программных систем, разрабатываемых на основе общих активов с помощью хорошо определенного метода повторного использования (Clements P., Northrop L., 2002).

Одним из способов реализации семейства программных продуктов является создание некоторого обобщенного приложения с отмеченными точками вариативности, на базе которого при разрешении этих точек вариативности строятся конечные продукты.

Для создания точек вариативности приложения необходимо, чтобы это приложение создавалось с использованием модельно-ориентированного подхода. Такая модель является удобным уровнем абстракции для построения вариативности в отличие от исходного кода.

Для работы с документацией семейства программных продуктов на кафедре системного программирования СПбГУ была создана технология DocLine (Д.В.Кознов, К.Ю.Романовский, 2008). В общем случае эта технология позволяет разрабатывать и сопровождать пакеты электронных документов, которые имеют значительные повторы. DocLine предоставляет возможность контестоно-зависимого, настраиваемого повторного использования фрагментов документов, что позволяет вносить изменения только в один актив и автоматически распрстранять их во многие места документации.

На сегодняшний день рынок Web-приложений является одним из наиболее активно развивающихся. Своей популярностью Web-приложения обязаны распространению Интернета, развитию Web-браузеров, отсутствию необходимости установки и простоте

использования. Современные Web-приложения, практически, не уступают в функциональности и качестве пользовательского интерфейса настольным (desktop) приложениям.

Для создания Web-приложений с помощью модельно-ориентированного подхода был создан язык WebML (Ceri, et. al., 2003). Чтобы создать отдельное Web-приложение с помощью WebML необходимо спроектировать ряд моделей. CASE-пакет WebRatio, реализующий этот язык, предоставляет графические инструменты для создания моделей и генерации по ним целевого кода.

Проект WebMLDoc (Голубев, 2010), (Дорохов, 2010) основывается на DocLine и посвящен трассировке изменений Web-приложений в изменения их пользовательской документации для семейств Web-приложений. В качестве средств разработки Web-приложений был выбран язык WebML и среда WebRatio, так как эти средства позволяют моделировать приложение с последующей генерацией кода по этим моделям, а работать с вариативностью удобно на модельно уровне. Во-первых, потому что оперировать модельными абстракциями удобнее, чем конструкциями кода на языке программирования (больший уровень абстракции, есть графическое представление и т.д.). Во-вторых, такое оперирование практично, когда модельным конструкциям однозначно соответствуют фрагменты кода (то есть имеется полная генерация кода по моделям и нет кода, который дописывается в стороне от моделей).

В работе (Голубев, 2010) разработан эскиз подхода для построения и разрешения модели вариативности при разработке семейства программных продуктов на основе *обобщенного проекта WebRatio* (исходный проект в среде разработки WebRatio, содержащий общие активы для разрабатываемого семейства Web-приложений). В работе (Дорохов, 2010) представлен метод для создания связей между отдельным приложением, разрабатываемым в WebRatio, и его пользовательской документацией в формате DocLine. Цель такой «привязки» документации в дальнейшем автоматическом прослеживании разделов документации, требующих корректировки, по изменениям в приложении (далее этот процесс будет называться *трассировкой*). Дальнейшим шагом по разработке проекта WebMLDoc является интеграция средств, представленных в (Голубев, 2010) и (Дорохов, 2010), а также доработка как самих подходов, так и их программных реализаций.

# Постановка задачи

Цель данной дипломной работы – расширить подход (Голубев, 2010), добавив в него поддержку иерархии гипертекстовой модели, расширить функциональность графического редактора, а также предоставить средства интеграции с результатами (Дорохов, 2010). Для достижения этих целей были сформулированы следующие задачи.

- Изучить язык WebML, его гипертекстовую модель, продукт WebRatio, а также изучить технологию Eclipse GMF.
- Доработать подход (Голубев, 2010) для поддержки приложений с несколькими профилями сайтов и тематическими областями страниц.
- Расширить функциональность редактора WebMLDoc для работы с вариативными моделями.
- Предоставить средства интеграции с результатами (Дорохов, 2010).

# Глава 1. Обзор используемых подходов и технологий

## *Семейства программных продуктов*

Семейство программных продуктов (software product lines) – это набор программных систем, разрабатываемых на основе общих активов с помощью хорошо определенного метода повторного использования (Clements P., Northrop L., 2002). Этот подход был предложен Дэвидом Парнасом в 1976 году (Parnas, 1976) и активно развивается в настоящее время. Его использование позволяет снизить затраты на разработку, уменьшить время выхода на рынок и повысить качество кода за счет того, что единожды написанный актив может использоваться в нескольких продуктах семейства. Поскольку заранее известна область применения каждого актива, для разработчиков не ставится задача добиваться их полной универсальности. Многие крупные компании используют такой подход в повседневной практике.

Общими активами могут быть:

- требования и результаты их анализа;
- модель предметной области;
- архитектура программного обеспечения;
- результаты работ по улучшению производительности;
- документация;
- планы по тестированию, тест и тестовые данные;
- знания и навыки разработчиков;
- процессы, подходы и инструменты;
- бюджеты и планы работ;
- программные компоненты и сервисы (как исходные, так и бинарные коды).

## *Диаграммы возможностей*

Диаграмма возможностей (feature diagrams) – это набор возможностей и иерархических связей между ними с явно выделенным корнем иерархии, который называется концептом (concept) (Kang, K., et. al., 1990). Основная задача модели возможностей – формализовать общие и различные свойства продуктов семейства. Наряду с концептом ключевым понятием этих диаграмм является возможность (feature), то есть обособленное свойство системы, распознаваемое с точки зрения пользователя или разработчика. Иерархические связи между возможностями, а также между концептом и возможностями называются отношением включения. Пример диаграммы возможностей представлен на рис. 1.

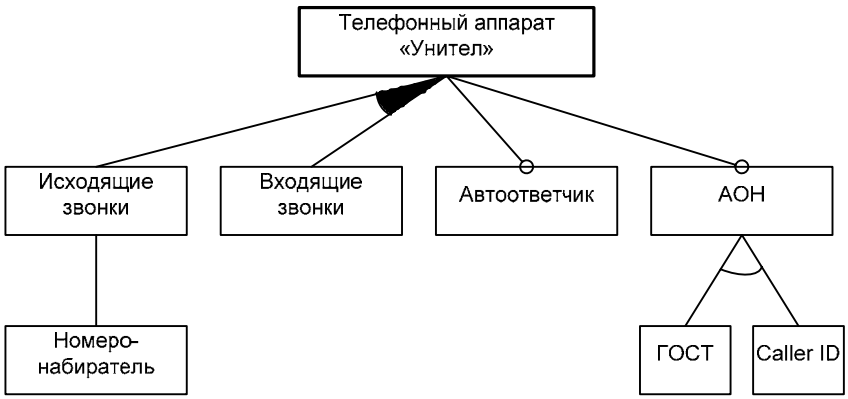


**Рисунок 1. Пример диаграммы возможностей.**

Прямоугольником изображается концепт или возможность. Линия, соединяющая концепт с возможностью, либо две возможности между собой, может иметь следующую дополнительную семантику:

- обязательное включение;
- необязательное включение, если линия помечена кружочком со стороны подвозможности.

Также возможны специальные значки на группе линий:

- черный сектор обозначает, что можно выбрать любое подмножество возможностей, в которые ведут линии, попавшие в данный сектор;

- незакрашенный сектор, оконтуренный снизу фрагментом окружности, обозначает, что можно выбрать только одну из возможностей, в которую ведет линия, попавшая в данный сектор.

## *Язык WebML*

WebML – язык моделирования Web-приложений, интенсивно работающих с большим количеством данных (data intensive systems). Он предоставляет инструменты для графического описания проектируемых процессов и приложений и подробно описан в работе (Ceri, et. al., 2003) .

Язык WebML состоит из четырех частей, каждая из которых является отдельной моделью характеристики проектируемого Web-приложения.

1. *Модель данных* (data model) описывает абстракцию организации данных.

2. *Модель запросов* (derivation model) позволяет расширить модель данных вычислимыми конструкциями (используется WebML/OQL). По сути, эта модель является текстовым языком запросов к модели данных.

3. *Гипертекстовая модель* (hypertext model) отвечает за описание пользовательского интерфейса Web-приложения. Она позволяет определить страницы, находящиеся на них элементы управления и навигацию между ними. Но не содержит в себе описание стилей внешнего представления интерфейса.

4. *Модель управления контентом* (content management model) расширяет гипертекстовую модель дополнительными конструкциями, такими как операции и транзакции. Это позволяет задавать поведение экранных форм, осуществлять вызовы различных операций и интеграцию с внешними сервисами.

## Гипертекстовая модель

Пользовательский интерфейс Web-приложения описывается в гипертекстовой модели.

Основные элементами этой модели являются профили сайта (site views), области (areas), страницы (pages), контент-модули (content-units) и связи (links).

Структура Web-приложения описывается в терминах страниц (pages). Они являются непосредственными элементами интерфейса, показываемыми пользователю, который заходит на сайт. Страницы состоят из контент-модулей – атомарных элементов, используемых для представления информации, описанной в модели данных.

Профили сайта (site views) и области (area) позволяют организовать страницы в иерархическую структуру. Профили сайта являются крупнейшими структурными элементами. Они описывают часть Web-приложения, разработанную для определенного класса пользователей (например, клиентов, простых посетителей, менеджеров, администраторов и др.).

Области представляют собой группы страниц, объединенных общим назначением (например, область работы с продуктами, область поддержки, область решений, и т. д.).

Навигация в приложении описывается с помощью ссылок (links).

## *CASE-пакет WebRatio*

WebRatio – это многоцелевая среда разработки, предназначенная для построения уникальных Web-приложений в различных областях. В WebRatio реализован язык WebML, и предоставляются возможности создания всех необходимых моделей языка. Для модели данных и гипертекстовой модели в пакете реализованы графические редакторы диаграмм. Пакет поддерживает автоматическую генерацию кода по моделям для платформы JEE/Tomcat и СУБД Oracle/MS SQL Server. Последние версии продукта интегрированы со средой разработки Eclipse.

На рисунке 2 представлен фрагмент примера гипертекстовой диаграммы в среде WebRatio.

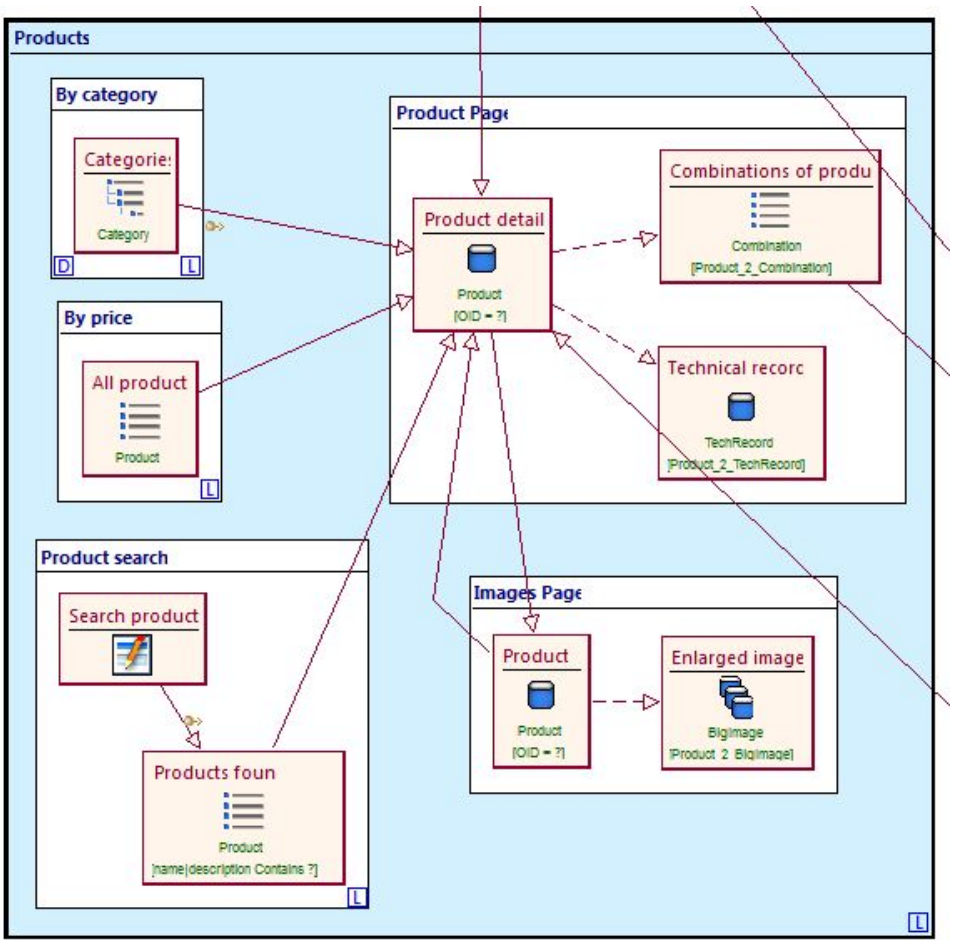


**Рисунок 2. Пример гипертекстовой модели в WebRatio.**

## *Метод DocLine*

DocLine – метод разработки документации семейств программных продуктов (Д.В.Кознов, К.Ю.Романовский, 2008). Данный метод восполняет разрыв между методами разработки семейств программных продуктов и подходами разработки технической документации.

Для исходного представления документации в DocLine предлагается оригинальный проблемно-ориентированный язык DRL (Documentation Reuse Language). DRL имеет две нотации:

- графическую – для проектирования структуры повторного использования документации;

- текстовую – чтобы описать варианты конфигурирования повторно используемых компонент и сами конкретные конфигурации для порождения конечных документов.

Для практического применения DocLine существует версия пакета, встроенная в интегрированную среду разработки приложений Eclipse. Для порождения документов в различных целевых форматах (PDF и HTML) DocLine интегрирован с популярной технологией DocBook.

## *Технология Eclipse GMF*

Eclipse – популярная платформа для построения интегрированных сред разработки. Платформа состоит из компонентов с открытым исходным кодом, используемым поставщиками инструментов для построения решений, встраиваемых в интегрированную рабочую область. Эти механизмы представляются через четко определенные интерфейсы, классы и методы в API (Eclipse Platform Overview, 2011).

Наиболее известные приложения на основе Eclipse Platform – различные «Eclipse IDE» для разработки программного обеспечения на множестве языков. В основе лежат фреймворк OSGi и SWT/JFace. Таким образом, (начиная с версии 3.0) Eclipse представляет собой не монолитную IDE, поддерживающую расширения (plug-in), а сама является набором расширений.

Graphical Modeling Framework (GMF) – технология, позволяющая по описаниям метамоделей построить полнофункциональный графический редактор, встраивающийся как расширение в среду разработки Eclipse. Конфигурации задают список элементов метамодели, которые будут использованы на диаграмме, а также их графическое представление и наборы связанных с ними элементов управления (Istria M., Chauvin M., Hunter A., 2011).

## *Проект WebMLDoc*

Основная идея данного проекта – организовать управление повторным использованием документации семейств программных продуктов на основе управления повторными активами кода. То есть, чтобы технический писатель при изменении ПО (общих активов) имел информацию о том, какие места пользовательской документации ему нужно изменить.

## Добавление вариативности в WebML

В (Голубев, 2010) описано, как в WebML и в WebRatio добавляется вариативность. Web-приложения ориентированы на взаимодействие с пользователем, поэтому при их проектировании основное внимание уделяется пользовательскому интерфейсу, который должен отражать все функции системы, а его модель является основой для дальнейшей автоматической генерации кода. Таим образом, в качестве модели, куда добавляется вариативность, выбрана гипертекстовая модель WebML.

Однако добавлять вариативность непосредственно в гипертекстовую модель неудобно ввиду ее больших размеров и загруженности данными, которые будут мешать пользователю ориентироваться при создании и разрешении точек вариативности. Таким образом, было предложено построить отдельную модель вариативности, представляющую лишь скелет гипертекстовой модели.

В (Голубев, 2010) сден акцент на страницах. Предполагается, что гипертекстовая WebML модель состоит из одного профиля сайта и из одной области, название которой совпадает с названием проекта, и этой области принадлежат все страницы приложения. Таким образом, пользователю предлагается построить свою вариативную модель, используя элементы вариативности и импортированные из WebRatio элементы (страницы гипертекстовой модели). На рисунке 3 представлен пример диаграммы такой вариативной модели.

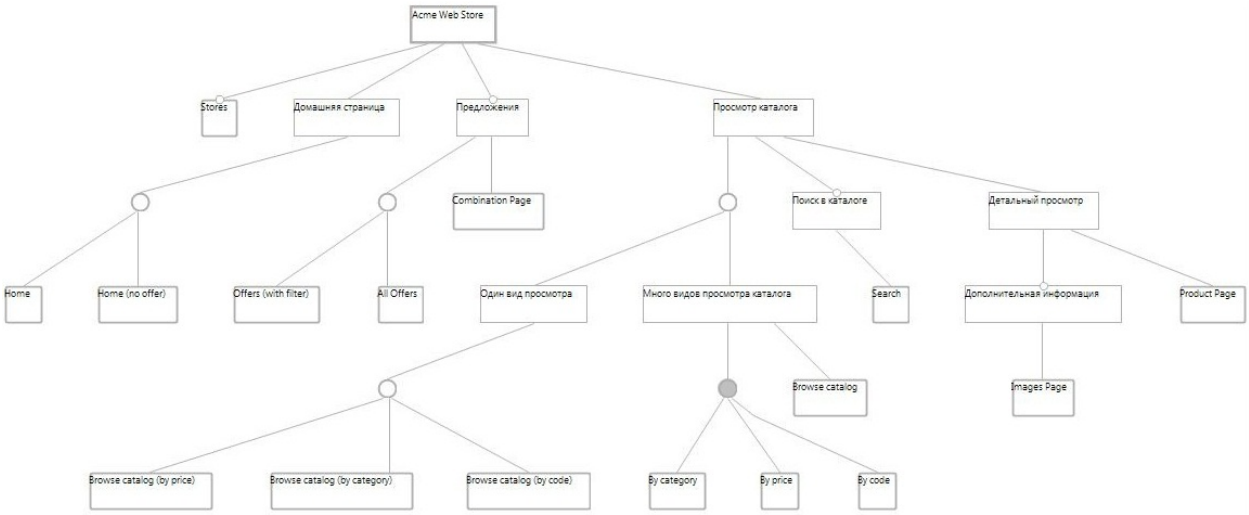


**Рисунок 3. Пример диаграммы вариативности в WebMLDoc.**

Элементы вариативности бывают трех видов:

- AND-элемент (node) – подразумевает одновременное присутствие в конечном продукте всех дочерних элементов;
- OR-элемент (OR-group) – в конечном продукте должен присутствовать хотя бы один из дочерних элементов;
- XOR-элемент (XOR-group) – в конечном продукте должен присутствовать только один из дочерних элементов.

А также некоторые элементы могут быть помечены как опциональные. Их пользователь может включать или не включать в конечный продукт по собственному усмотрению. Нетрудно видеть, что в (Голубев, 2010) реализованы основыне конструкции диаграмм возможностей.

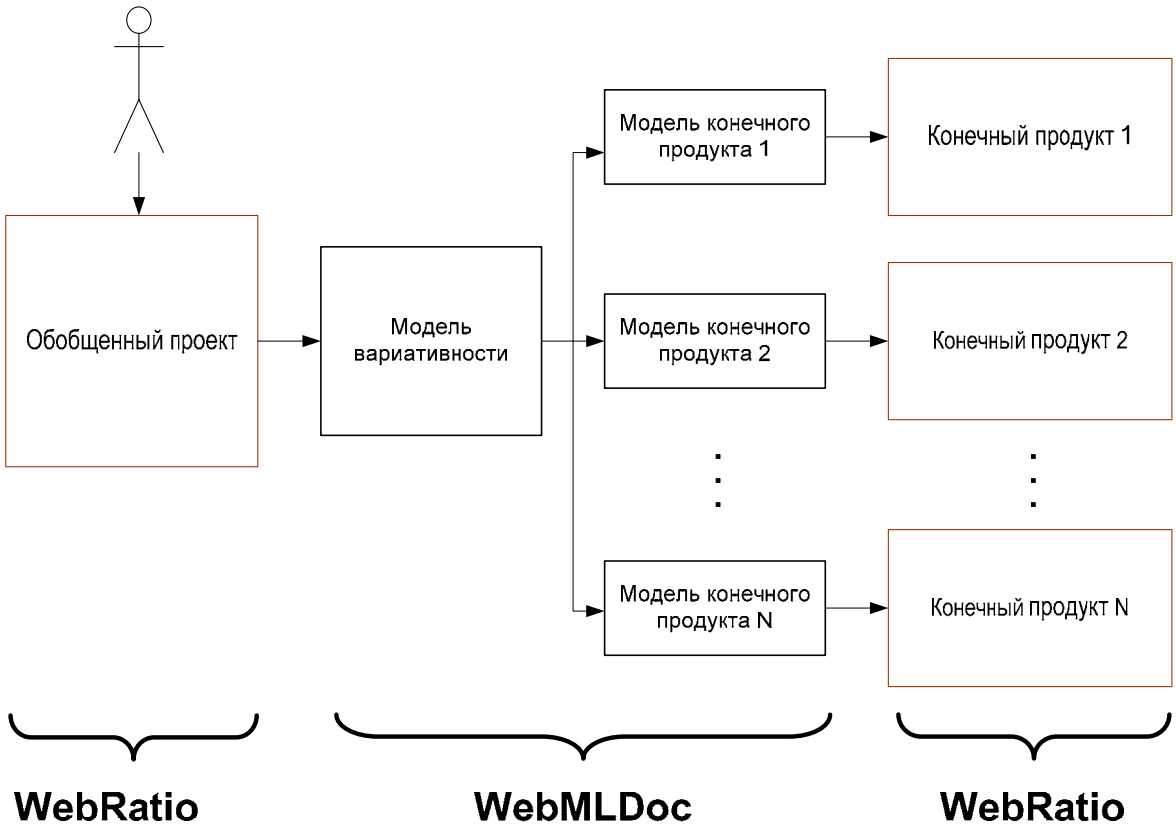


**Рисунок 4. Схема построения семейства Web-приложений.**

Таким образом, построение семейства Web-приложений следующим образом.

1. Построение WebML-описания обобщенного продукта с полным набором функциональности семейства.

2. Построение модели вариативности по полученной гипертекстовой модели языка WebML.

3. Разрешение вариативности и получение разрешенной модели варитивности для конечного продукта.

4. Генерация WebML-модели конечного продукта на основе разрешенной диаграммы вариативности и WebML-модели обобщенного проекта.

### *Связь WebRatio с DocLine*

В работе (Дорохов, 2010) представлено решение проблемы привязки документации, создаваемой в DocLine, к элементам гипертекстовой модели WebRatio. Ставилась задача по изменениям пользовательского интерфейса Web-приложения, создаваемого в среде WebRatio, определить список разделов пользовательской документации в DocLine, требующих корректировки или проверки.

# Глава 2. Доработка поддержки вариативности Web-приложений

Подход, предложенный в (Голубев, 2010) удобен в том случае, когда Web-приложение имеет простую структуру (состоит из нескольких страниц). Если оно объединяет в себе несколько профилей сайта и областей страниц, работать с таким приложением в существующем редакторе оказывается невозможно. В этой главе будет описано, как предлагается усовершенствовать подход Голубева для работы с такими приложениями.

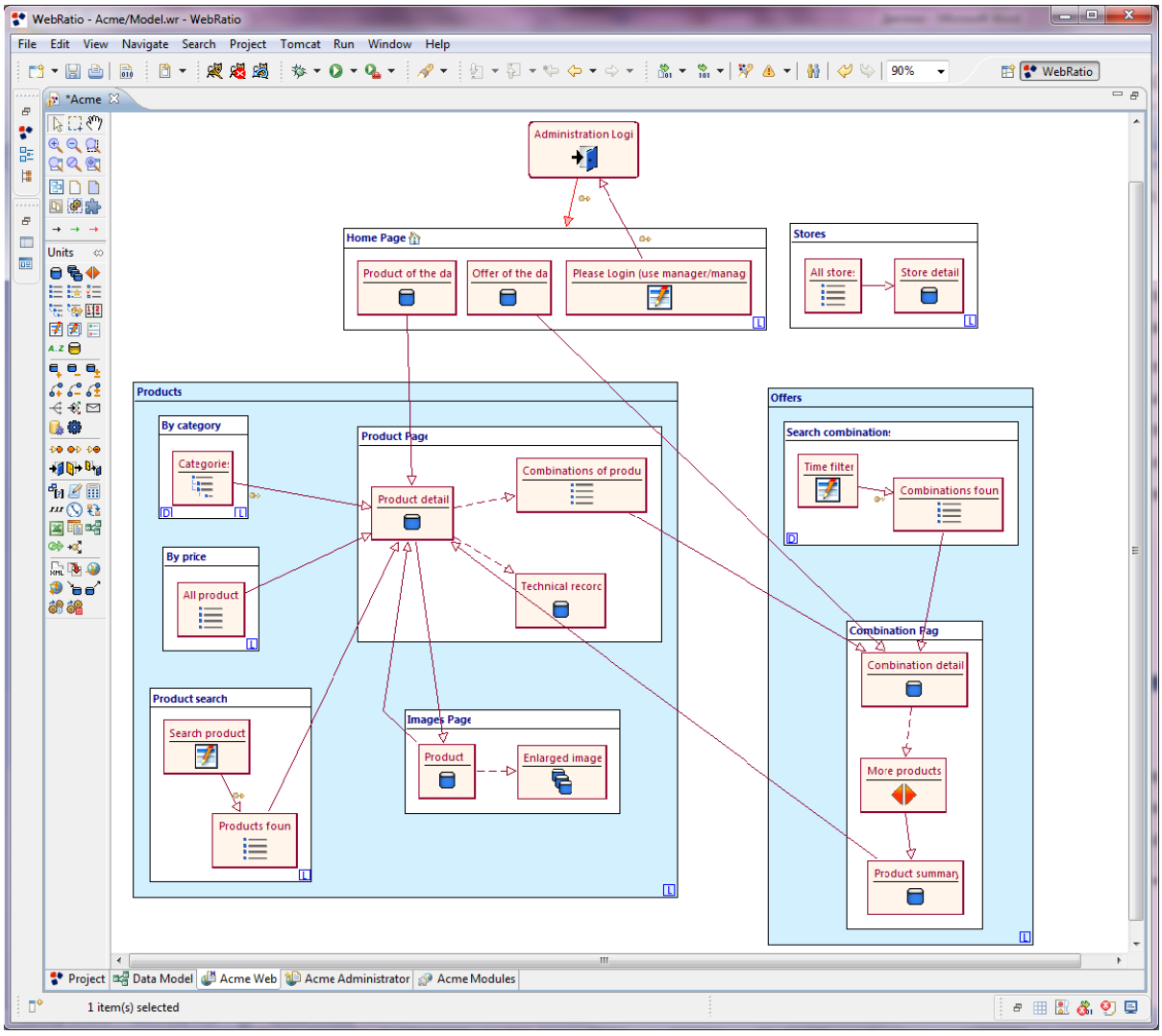


**Рисунок 5. Диаграмма профиля сайта в WebRatio.**

Рассмотрим гипертекстовую модель Web-приложения, описанного на языке WebML: она представляет собой набор диаграмм, соответствующих различным профилям (site view).

Каждый профиль содержит в себе набор страниц, некоторые из них объединены в группы – тематические области (area). Пример одного из профилей гипертекстовой модель в WebRatio представлен на рисунке 5.

При построении модели вариативности можно игнорировать профили и области, рассматривая только страницы приложения. Это позволит использовать ранее разработанный подход, но это сделает практически невозможной работу пользователя с диаграммой вариативности.

Таким образом, необходимо доработать этот подход таким образом, чтобы диаграмма вариативности отражала верхнеуровневую иерархическую структуру гипертектовой модели.

## *Расширение модели вариативности*

В первую очередь, в диаграмму вариативности должны быть добавлены элементы, соответствующие профилям и областям Web-приложения: SiteView и Area.

На рисунке 6 представлен пример иерархической структуры элементов гипертекстовой модели.

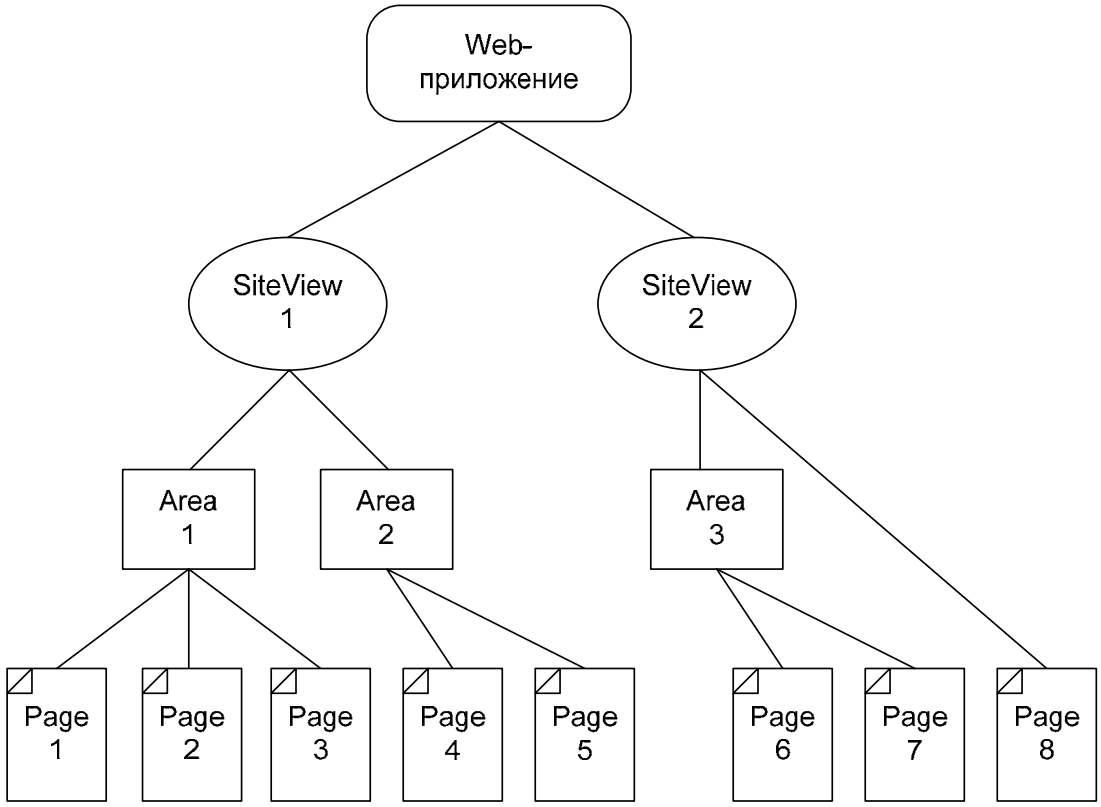


**Рисунок 6. Иерархия элементов гипертектовой модели.**

Чтобы сохранить эту иерархию в вариативной модели, предлагается автоматически переносить ее в модель и не допускать ее изменения пользователем. Действия пользователя должны быть ограничены манипулированием с элементами вариативности.

## *Построение каркаса диаграммы вариативности*

Предлагается строить вариативную диаграмму на основе автоматически генерируемого каркаса, представляющем собой иерархическое дерево элементов гипертекстовой модели, как показано на рисунке 7. Построение модели вариативности должно происходить вручную и выражается в добавлении пользователем элементов вариативности в это дерево.

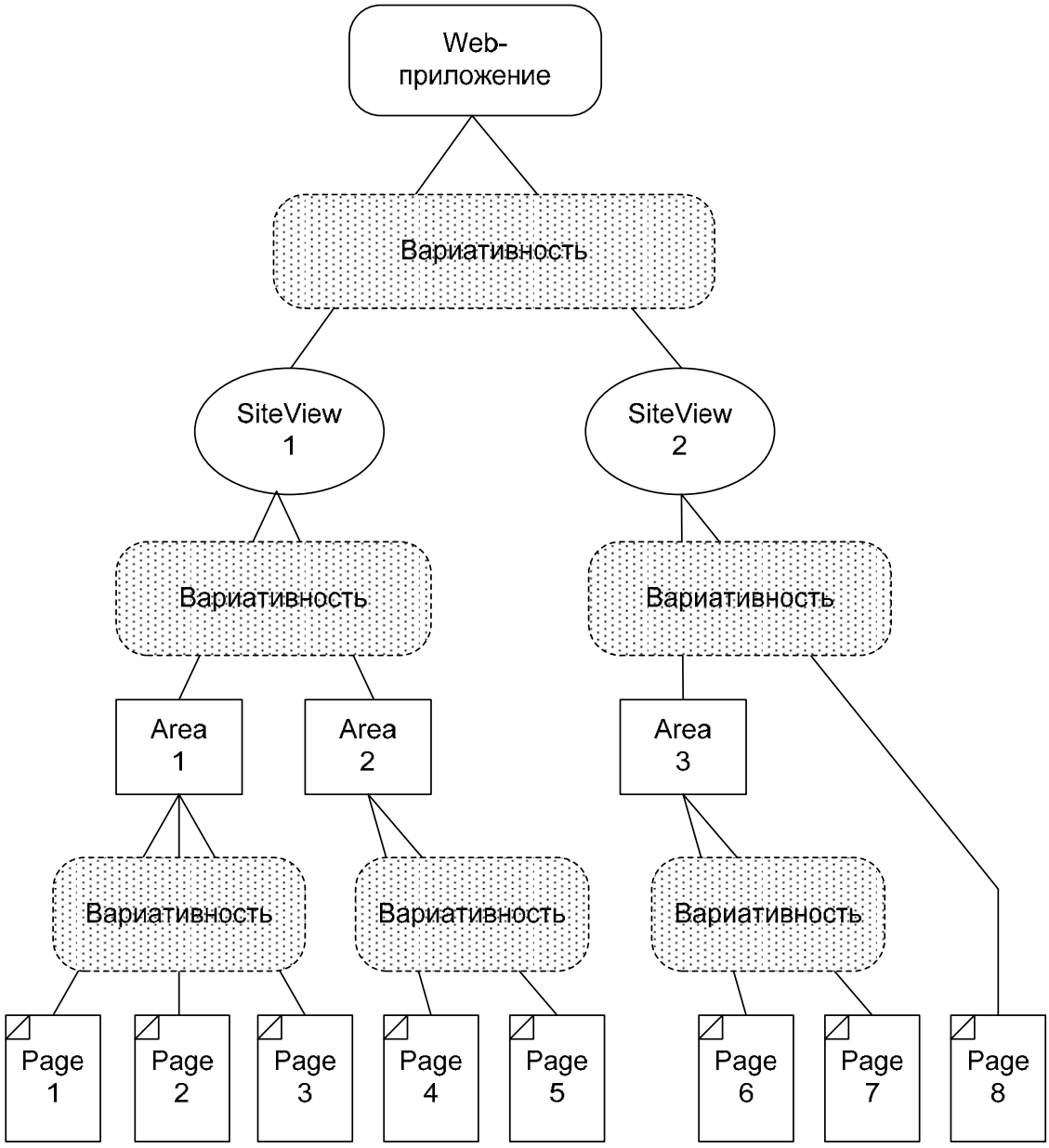


**Рисунок 7. Схема диаграммы вариативности.**

На рисунке 7 заштрихованными областями отмечены места, куда могут быть добавлены элементы вариативности (это может быть как один элемент, так и многоуровневая конструкция элементов). Получается вполне наглядная структура, в которую пользователь постепенно сможет встраивать вариативность.

Так как далеко не каждый модуль в итоге должен содержать элементы вариативности, было решено изначально строить каркас диаграммы в виде, соответствующем рисунку 6, то есть с прямыми связями от корня к профилям, от профилей – к областям и страницам, и

от областей – к страницам. После этого пользователь сам сможет выбрать, в каких узлах ему надо добавить вариативность, а в каких все остается как есть.

### *Включение элементов вариативности в диаграмму*

На диаграмме могут присутствовать следующие элементы вариативности:

- AND-элемент (node) – подразумевает одновременное присутствие в конечном продукте всех дочерних элементов;
- OR-элемент (OR-group) – в конечном продукте должен присутствовать хотя бы один из дочерних элементов;
- XOR-элемент (XOR-group) – в конечном продукте должен присутствовать только один из дочерних элементов.

Некоторые элементы диаграммы могут быть помечены как опциональные – их присутствие в конечном продукте определяется пользователем.

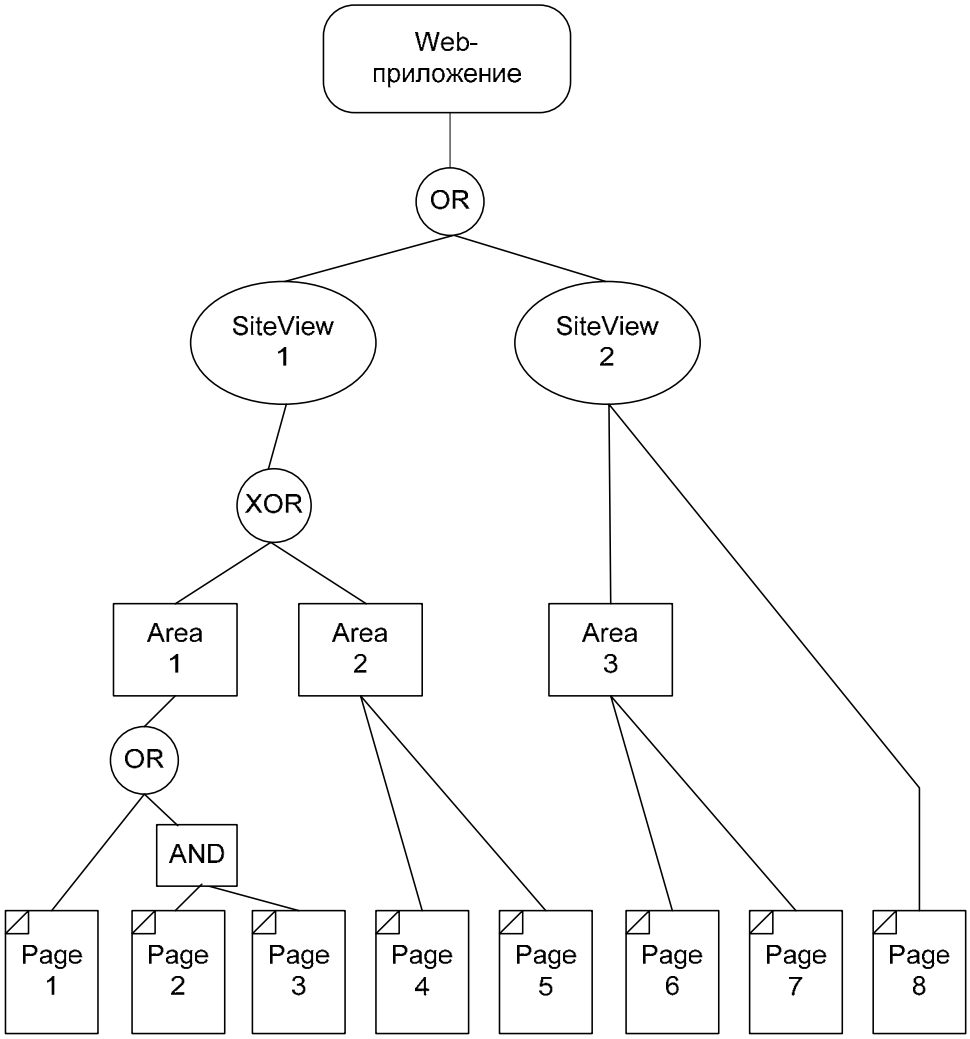


**Рисунок 8. Пример модели вариативности.**

Добавление элементов вариативности предлагается осуществлять для каждого узла в отдельности. Пользователю предоставляется список прямых потомков узла, из которого он сможет выбрать элементы для объединения элементом вариативности. После этого соответствующий элемент будет добавлен в качества потомка узла, а выбранные элементы станут потомками этого нового элемента вариативности. Это удобно как с точки зрения пользователя – избавляет его от необходимости выстраивать все это на диаграмме вручную, так и с точки зрения целостности диаграммы – в каждый момент времени дерево вариативности остается корректным.

Например, на рисунке 9 изображено, как происходит добавление вариативного элемента к узлу диаграммы. Пользователь выделил элемент SiteView и выбрал несколько его потомков (в данном случае всех, но это не обязательно) для объединения в XOR-группу.

Теперь при создании продукта, пользователь должен выбрать только одну из областей: Area1 или Area2.

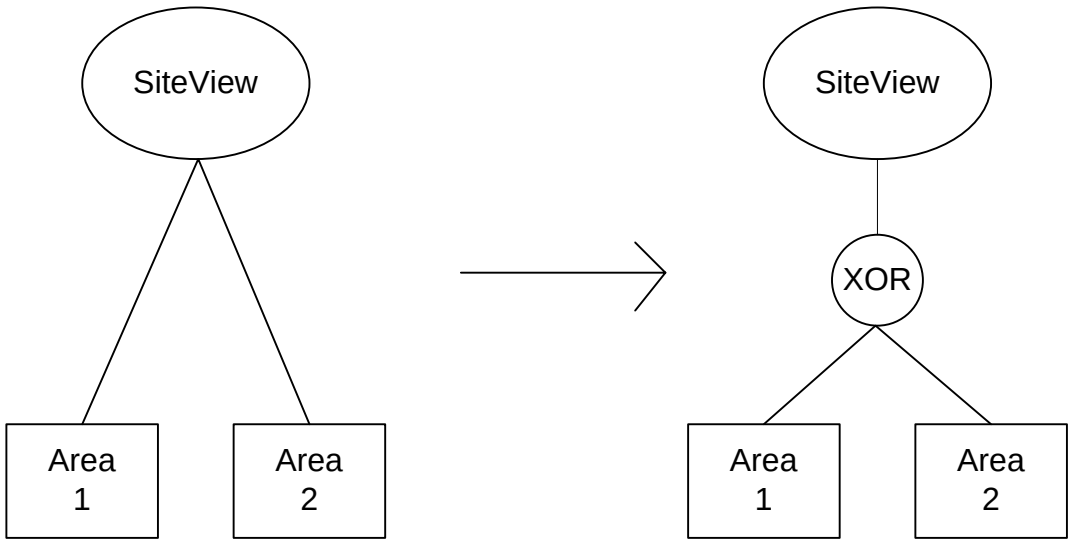


**Рисунок 9. Пример добавления элемента вариативности к узлу.**

# Глава 3. Новые возможности редактора WebMLDoc

Система WebMLDoc представляет собой визуальный редактор для работы с WebML-диаграммами и документацией в формате DocLine. WebMLDoc позволяет пользователю добавлять вариативность в собственный проект и получать на выходе WebML-диаграммы готовых продуктов и DocLine-документацию к ним. Из этого набора, используя существующие программные средства, пользователь получит готовое приложение и документацию.

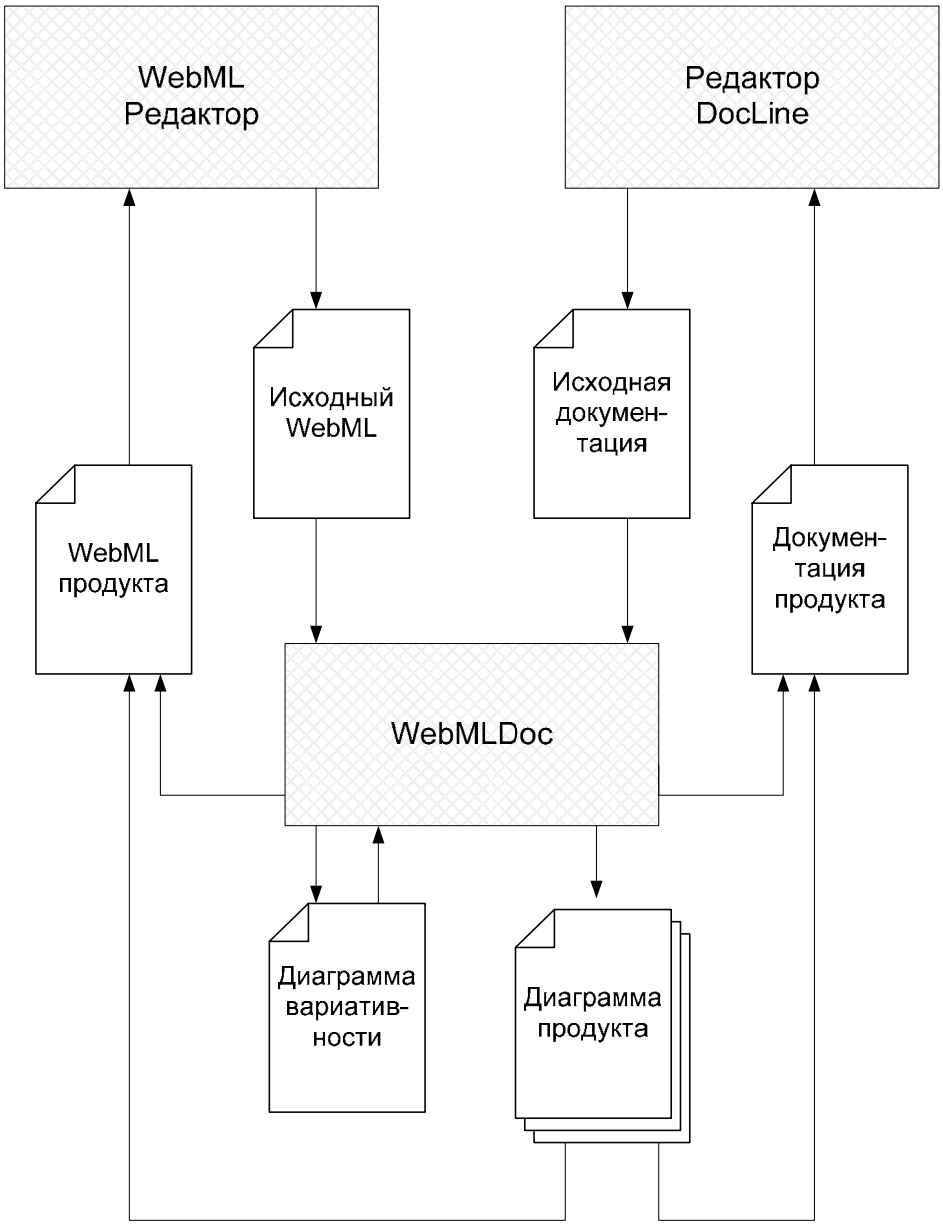


**Рисунок 10. Схема взаимодействия редакторов.**

### *Создание диаграммы вариативности*

При создании нового проекта пользователь указывает расположение файлов с WebML-моделью и DocLine-документацией в файловой системе. На основе этих данных происходит генерация заготовки диаграммы вариативности, в которую пользователь должен добавить необходимые элементы вариативности.

Остальные изменения (такие как добавление новых невариативных элементов, их удаление или изменение) должны вноситься на уровне WebML-модели и после синхронизации будут перенесены на диаграмму.

### *Редактирование диаграммы вариативности*

Взаимодействие пользователя с системой осуществляется путем вызова контекстных меню для тех элементов диаграммы, которые требуют расширения. Таким способом можно добавить к элементу диаграммы элемент вариативности, который объединит часть потомков, привязать к элементу диаграммы элемент документации или удалить потерявший актуальность элемент вариативности.

### *Синхронизация WebRatio и WebMLDoc*

При изменении WebML-модели соответствующие изменения должны быть внесены и в диаграмму вариативности. По требованию пользователя редактора WebMLDoc происходит считывание последней WebML-модели и исправление диаграммы вариативности в соответствии с нею. После этого пользователь может просмотреть отчет о том, какие элементы были удалены, изменены или добавлены и какие элементы документации ему следует исправить в связи с этим.

### *Создание продукта*

После добавления необходимых элементов вариативности на диаграмму пользователь может перейти к созданию конечного продукта семейства. Выбрав из контекстного меню файла диаграммы вариант «Add product» пользователь получит список существующих элементов. Выбор элементов происходит в строгом с соответствие с диаграммой вариативности. Например, система не позволит пользователю отметить два элемента,

являющиеся потомками одного XOR-узла. После того, как пользователь выделит те элементы, которые должны присутствовать в конечном продукте, и подтвердит свой выбор, будет сформирована диаграмма продукта, файл с документацией продукта и WebML-диаграмма продукта.

### *Перегенерация продукта*

В случае изменения общей WebML-модели, если не произошло удаления задействованных в продукте элементов, можно произвести перегенерацию WebML-модели продукта на основе уже существующей диаграммы вариативности. Если такие изменения произошли, то генерация пройдет некорректно.

### *Пошаговое описание работы с редактором WebMLDoc*

Работу пользователя с системой можно поэтапно описать следующим образом.

1. Создание в редакторе WebRatio проекта Web-приложения, содержащего полный набор функциональности планируемого семейства Web-приложений.

2. Создание набора пользовательской документации исходного Web-приложениия средствами DocLine.

3. Создание проекта в WebMLDoc на основе гипертекстовой модели исходного WebRatio-продукта.

4. При создании проекта в редакторе WebMLDoc происходит автоматическая генерация дерева элементов исходного web-приложения с фиксированной иерархией.

5. Пользователю предлагается дополнить дерево элементов точками вариативности и группирующими элементами.

6. Так же пользователю предлагается связать элементы дерева с соответствующими им элементами документации, чтобы в конечном итоге получить для каждого продукта документацию, четко соответствующую набору его функциональности.

7. После построения такого дерева, можно переходить к созданию конечных продуктов семейства. Вызывается мастер, предлагающий разрешить точки вариативности, предлагая пользователю выделить те элементы функциональности, которые должны присутствовать в конечном продукте.

8. После завершения работы мастера создаются следующие элементы:

    - новая диаграмма, представляющая собой иерархическое дерево функциональности конечного продукта;

    - WebRatio-проект, соответствующий новому продукту;

    - DRL-документация, представляющий собой пользовательскую документацию продукта.

9. Далее средствами WebRatio из проекта автоматически генерируется Web-приложение и средствами DocLine – документация.

# Глава 4. Реализация системы

## *Общая схема работы системы WebMLDoc*

На схеме ниже изображена схема работы системы WebMLDoc в целом. Выделенная часть реализована в рамках данной работы, остальное – в работе (Лебедева, 2011).

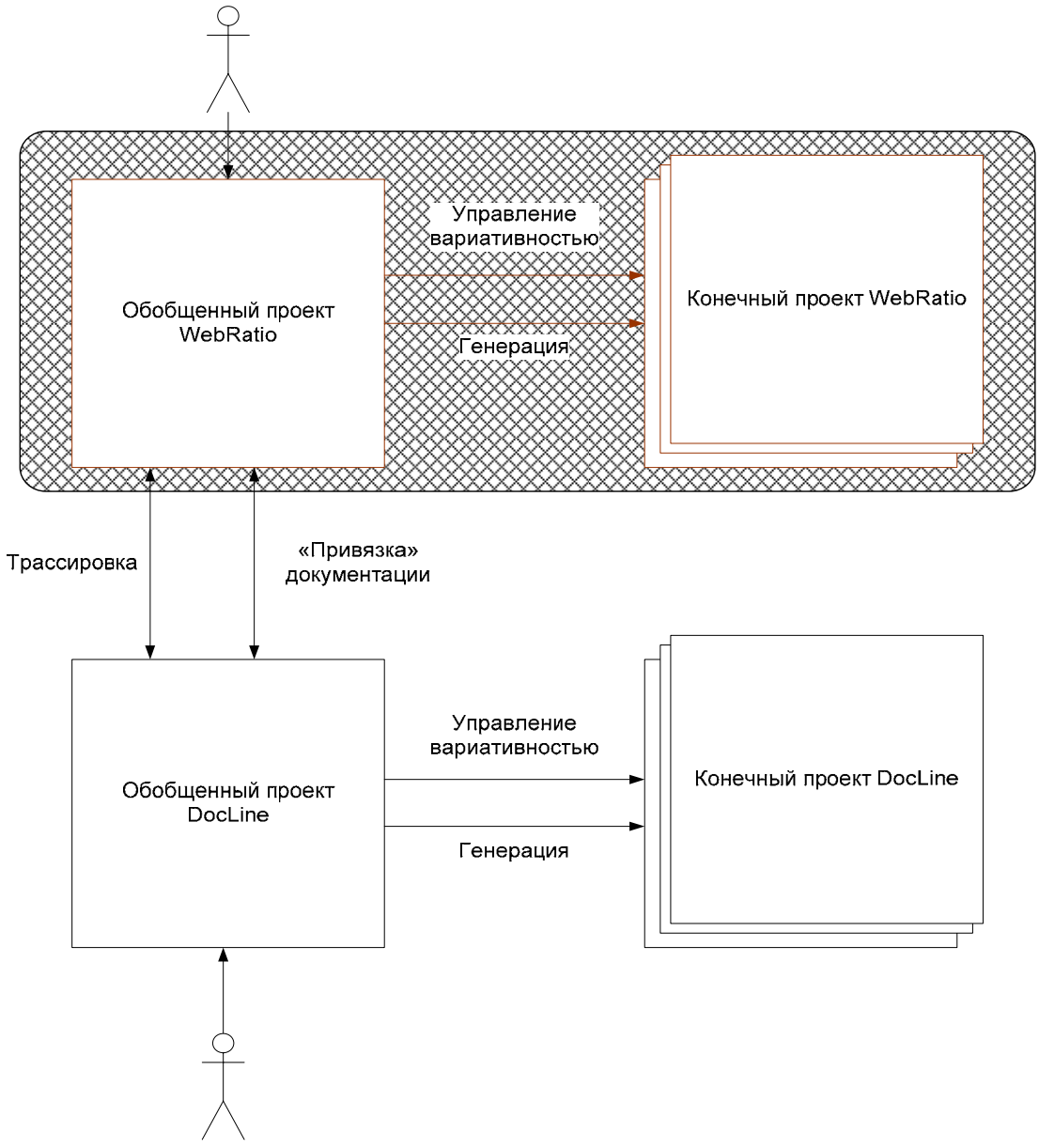


**Рисунок 11. Общая схема работы WebMLDoc.**

## *Подробности реализации*

### Создание редактора диаграмм

Редактор WebMLDoc реализован на платформе Eclipse с использованием технологии Eclipse GMF. Эта технология позволяет на основе набора моделей автоматически генерировать графические редакторы. Дополнительная функциональность добавляется путем добавления собственных функций в точки расширения (extension points).

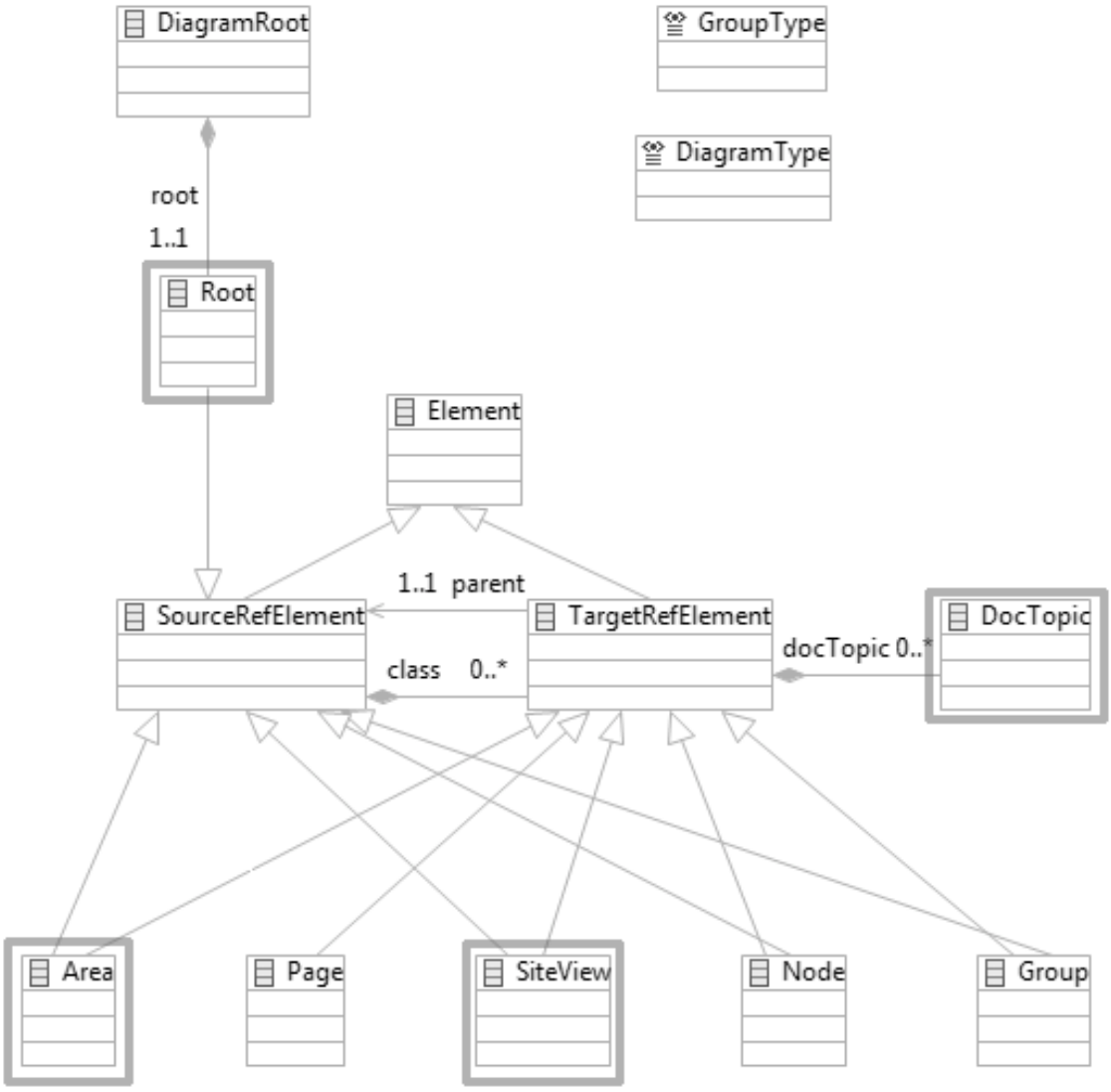


**Рисунок 12. Пример доменной модели.**

Для построения редактора с помощью GMF требуется создать доменную модель (domain model). В данной работе за основу была взята модель из работы (Голубев, 2010). Она была дополнена несколькими новыми элементами, которые выделены на рисунке 12 серыми рамками.

На диаграмме представлены все виды элементов, которые нужны для задания вариативности для Web-приложения, написанного на WebML:

- элементы гипертекстовой модели языка WebML – Page, Area, Site View;
- элементы вариативности: AND-элемент (node), OR-элемент (OR-group), XOR-элемент (XOR-group);
- корень вариативного дерева – Root.

На основе доменной модели средствами GMF была автоматически сформирована модель генерации кода и по ней сгенерирован код для описания доменной модели и внесения в нее изменений.

Для получения графического редактора диаграмм на основе доменной модели с помощью средств GMF было сформировано и настроено три модели:

- GMFGraph – модель графической нотации для классов доменной модели;
- GMFTool – модель палитры инструментов для изменения диаграммы;
- GMFMap – модель связывания моделей GMFGraph и GMFTool.

На основе модели GMFMap была сформирована GMFGen-модель и на ее основе сформирован код пакета diagram для графического редактора.

После генерации код был модифицирован вручную, чтобы соответствовать поставленной задаче:

- добавлен импорт WebRatio-проектов и формирование на их основе заготовки диаграммы вариативности;

- добавлены средства для корректного добавления в заготовку различных точек вариативности;
- добавлена возможность синхронизации с проектом WebRatio и изменения диаграммы вариативности в соответствии с его текущим состоянием;
- добавлена генерация проектов WebRatio для конечных продуктов.

## Импорт WebRatio-проекта

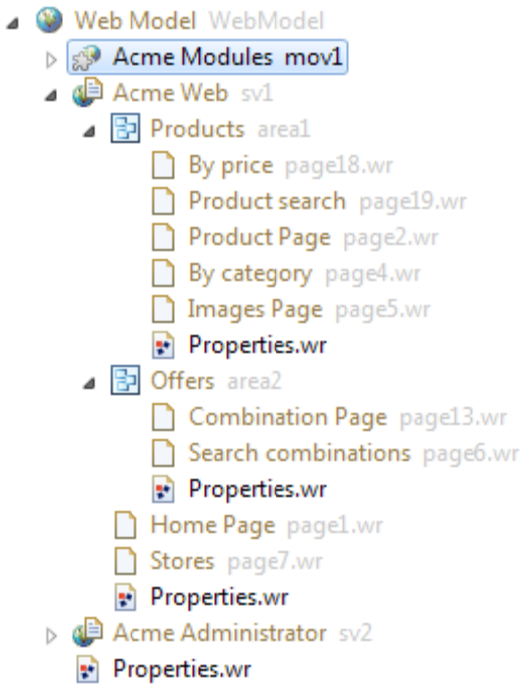


**Рисунок 13. Структура проекта WebRatio.**

Так как гипертекстовая диаграмма WebML-модели в приложении WebRatio хранится в виде иерархической структуры папок, импорт в проекте (Голубев, 2010) был организован путем считывания данных из файлов properties.wr лежащих в папках siteView и area, и pageXX.wr, описывающих лежащие в папках страницы.


**Рисунок 14. Фрагмент заготовки диаграммы вариативности.**

В (Голубев, 2010) импорт был организован путем считывания всех страниц из первой области первого профиля сайта. В новом проекте мы считываем полную структуру папок и полученное дерево элементов записываем в файл. С этим файлом мы ассоциируем файл соответствующей диаграммы. Таким образом, пользователь получает заготовку для добавления элементов вариативности.

## Добавление вариативности на диаграмму

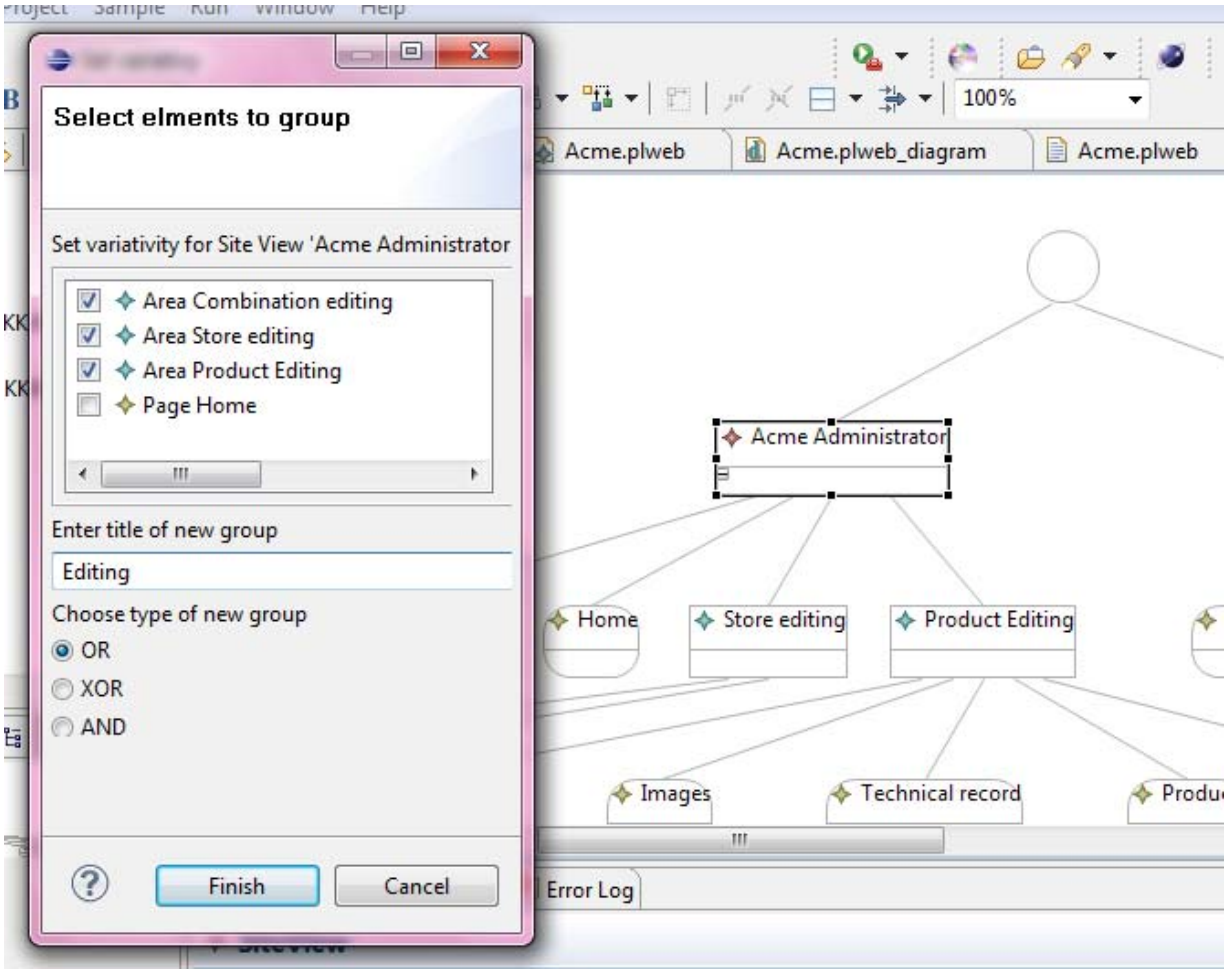


**Рисунок 15. Добавление нового элемента вариативности.**

В (Голубев, 2010) добавление вариативности осуществлялось путем создания элементов диаграммы с помощью палитры элементов. Эти действия производились пользователем вручную. Во-первых, это неудобно для пользователя – помимо самого элемента нужно добавлять еще и связи, во-вторых, при работе с трехярусной диаграммой велика вероятность, что пользователь ошибется и неправильно соединит элементы. Поэтому было предложено осуществлять добавление элементов, вызывая контекстное меню узла и выбирая из предложенного списка элементы для объединения под новым элементом вариативности.

После подтверждения выбора, в диаграмму автоматически будут внесены нужные изменения. Например, после подтверждения выбора из рисунка 14 диаграмма изменится следующим образом.

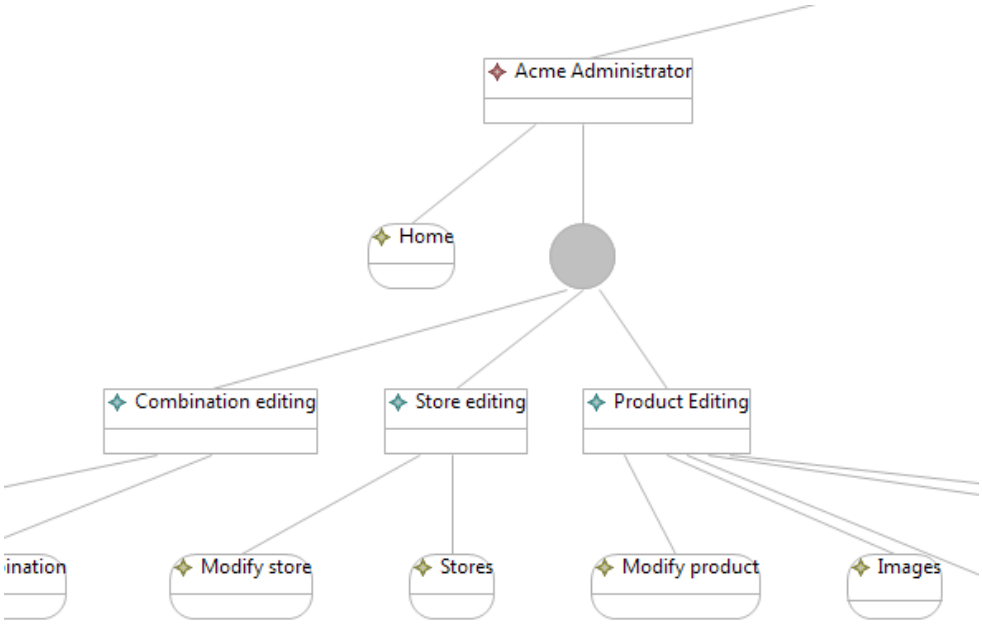


**Рисунок 16. Фрагмент диаграммы вариативности.**

## Внесение изменений в диаграмму после изменения WebML-модели

В связи с добавлением новых элементов функция обновления диаграммы была полностью изменена. Теперь оно происходит путем считывания и сравнения деревьев вариативной модели и дерева гипертекстовой модели WebRatio-проекта. Из дерева вариативной модели удаляются элементы, ранее удаленные из WebRatio-проекта и в него же добавляются новые. После этого происходит перезапись xml-файла и обновление соответствующей ему диаграммы. Все произошедшие изменения записываются в отельный файл.

## Создание конечного продукта и генерация WebRatio-проекта

В эту функцию были внесены изменения, связанные с расширением метамодели данных. Была добавлена поддержка новых видов элементов диаграммы.

## *Трудности реализации*

Ниже перечислены основные трудности, возникшие при выполнении работы.

- Сложности изучения технологии GMF в связи с отсутствием единой и ясной документации. В связи с этим процесс создания моделей для генерации кода редактора оказался очень затруднительным.
- После создания корректной модели код не всегда генерируется правильно.
- Сложность изменения модели после генерации и «ручных» модификаций кода;
- Сложность программного взаимодействия с элементами диаграммы.

# Заключение

В ходе выполнения данной работы были достигнуты следующие результаты.

- Изучен язык WebML, его гипертекстовая модель и пакет WebRatio, а также технология Eclipse GMF.
- Доработан подход (Голубев, 2010) для поддержки любых приложений (с несколькими профилями сайтов и тематическими областями страниц).
- Редактор WebMLDoc усовершенствован следующим образом:
    - расширена метамодель диаграммы вариативности таким образом, чтобы можно было работать с любым приложением, созданным на основе WebML-модели;
    - модифицированы следующие функции WebMLDoc:
        - импорт гипертекстовой модели обобщенного проекта WebRatio;
        - синхронизация модели вариативности в редакторе WebMLDoc и обобщенного проекта WebRatio;
        - разрешение модели вариативности и экспорт конечного проекта в WebRatio;
    - реализовано автоматическое построение заготовки вариативной модели;
    - реализованы средства для корректного построения модели вариативности в редакторе WebMLDoc на базе обобщенного проекта WebRatio.

Можно выделить следующие направления развития разработанного подхода.

- Возможность редактирования модели конечного продукта в редакторе WebMLDoc. Это оправдано, если после создания этой модели понадобилось внести в нее незначительные изменения  наподобие добавления или удаления какого-то общего актива.

- Отслеживание соответствия общей модели и модели конечного продукта, если изменились какие-либо общие активы.
- Учет связей и контент-модулей гипертекстовой модели в модели вариативности WebMLDoc.
- Добавление вариативности в другие модели языка WebML.

# Список литературы


Clements P., Northrop L. (2002). *Software Product Lines: Practices and Patterns*. Boston: Addison-Wesley.

*Eclipse Platform Overview.* (2011). Получено из eclipse.org: http://www.eclipse.org/platform/overview.php

Istria M., Chauvin M., Hunter A. (2011). *Graphical Modeling Project (GMP).* Получено из eclipse.org: http://wiki.eclipse.org/GMP

Kang, K., Cohen, S., Hess, J., Novak, W., Peterson, S. (1990). *Feature-Oriented Domain Analysis (FODA) Feasibility Study*. Pittsburgh: CMU/SEI-90-TR-021.

Parnas, D. L. (1976). On the Design and Development of Program Families. *IEEE Transactions on Software Engineering*.

Stefano Ceri, Piero Fraternali, Aldo Bongio, Marco Brambilla, Sara Comai, Maristella Matera. (2003). *Designing Data-Intensive Web Applications.* Amsterdam: Morgan Kaufmann.

Голубев, Н. А. (2010). *Расширение гипертекстовой модели языка WebML средствами поддержки вариативности.* СПбГУ, математико-механический факультет, кафедра Информатики, Санкт-Петербург.

Дорохов, В. (2010). *Автоматизированная поддержка пользовательской документации Web-приложений, разрабатываемых в среде WebRatio.* СПбГУ, математико-механический факультет, кафедра Системного программирования, Санкт-Петербург.

К. Ю. Романовский, Д. В. Кознов. (2007). Язык DRL для проектирования и разработки документации семейств программных продуктов. *ВЕСТНИК Санкт-Петербургского университета Сер. 10 Вып. 4*.

Д.В.Кознов, К.Ю.Романовский. (2008). DocLine: метод разработки документации семейства программных продуктов. Программирование, 2008, № 4 С. 1-13.

Лебедева, М. В. (2011). *Разработка документации в задаче одновременной разработки продукта и его документации на основе семейств.* СПбГУ, математико-механический факультет, кафедра информатики, Санкт-Петербург.

# Глоссарий

***Обобщенный проект WebRatio (General WebRatio Project)*** – исходный проект в среде разработки WebRatio, содержащий общие активы для разрабатываемого семейства Web-приложений. На основе этого проекта строится модель вариативности в WebMLDoc.

***Конечный проект WebRatio (Customized WebRatio Project)*** – проект WebRatio, полученный в результате «разрешения» модели вариативности в WebMLDoc. Модель вариативности строится по обобщенному проекту WebRatio. Результат разрешения модели вариативности экспортируется в среду WebRatio из WebMLDoc, что и является конечным проектом WebRatio.

***Генерация (перегенерация)*** – автоматическое создание «с нуля» одного актива по некоторому исходному; при повторной генерации (перегенерации) прежний сгенерированный актив удалятся и генерируется новый.

***Синхронизация*** – автоматическое внесение точечных изменений в один актив по точечным изменениям в другом активе. Принципиально отличается от перегенерации.

***«Привязка» (mapping)*** – создание связей между элементами двух активов.

***Трассировка*** – автоматическое прослеживание изменений в одном активе по изменениям в другом. В отличие от синхронизации, изменения автоматически не вносятся во второй актив, а только определяется список его элементов, которые нужно изменить. Сами изменения могут вноситься «вручную».