инженерия программного обеспечения

Кознов Дмитрий Владимирович

УДК 681.3

ПРЕДМЕТНО-ОРИЕНТИРОВАННОЕ ВИЗУАЛЬНОЕ РЕШЕНИЕ ДЛЯ СБОРА И УПОРЯДОЧИВАНИЯ ИНФОРМАЦИИ ПРИ РАЗРАБОТКЕ ИНФОРМАЦИОННОЙ WEB-СИСТЕМЫ[1]

Аннотация

В настоящее время в области разработки ПО развивается предметно-ориентированное визуальное моделирование, нацеленное на разработку визуальных решений для отдельных предметных областей, проектов и задач. Однако продолжают остро стоять вопросы управления такими проектами, не хватает развитых системных подходов, позволяющих преодолеть многочисленные риски таких разработок. В связи с этим представляют интерес отдельные успешные решения. В данной работе представлено визуальное решение РУП (РУсско-финское Приграничное сотрудничество), созданное на базе Microsoft Visio и использовавшееся для проектирования контента Web-системы, предназначенной для поддержки русских путешественников в Финляндии, а финских – в России (информация о публичных и государственных услугах). Это решение дополняет инструмент онтологического инжиниринга ОРГ-Мастер, использовавшийся для создания модели контента указанной выше Web-системы: был создан визуальный язык и соответствующий графический редактор, реализована генерация из этого редактора в ОРГ-Мастер, а также поддержана циклическая разработка.

Ключевые слова: предметно-ориентированное моделирование, визуальное моделирование, модельно-ориентированная разработка, DSM, domain-specific modeling, DSM-подход, MDA, model-driven architecture, model-driven development, MDD, DSM, DSL, UML, Интернет-сервисы, е-сервисы, Web-приложения, онтологии, онтологический инжиниринг, Semantic Web.

ВВЕДЕНИЕ

Предметно-ориентированное визуальное моделирование (Domain-Specific Modeling, DSM) – это область программной инженерии, которая посвящена разработке языков и программных средств, основанных на визуальном моделировании для отдельных предметных областей, компаний, линеек программных продуктов и проектов [1]. Основная идея DSM-подхода заключается в том, чтобы сузить область применения визуальных решений и за счёт этого получить более ощутимый выигрыш от использования графических моделей – бо́льшую выразительную силу и возможность эффективной автоматической генерации по моделям конечного кода и других артефактов, дополни-

[1] Работа выполнена при поддержке грантов ENPI № 2010-021-SE396, РФФИ № 11-01-00622-а и РФФИ 12-01-00415-а.

тельные интеграционные возможности и пр. DSM-подход возник в связи с осознанием проблем использования стандартных средств визуального моделирования – прежде всего, языка UML [2, 3], а также ввиду прогресса платформ для разработки визуальных решений – EMP [4], MetaEdit+ [5], QReal [6] и др. (обзоры таких средств см. в работах [7, 8, 9]).

Однако практическое использование DSM-подхода сталкивается с рядом проблем: сложности со сбором и управлением требованиями и, в частности, с точным описанием визуального языка перед началом разработки, отсутствие явного заказчика решения, проблемы с управлением проектами, технические вопросы, связанные с циклической разработкой, поддержкой целостности моделей и т. д. [10, 11]. Также существуют сложности оценки эффективности практического применения таких решений [12]. Наконец, актуальным для исследования вопросом являются различные варианты мотивации команды, решившей использовать DSM-подход, а также те конкретные задачи, которые решаются с помощью визуального моделирования[1]. Поэтому описание практических реализаций DSM-подхода является интересной и актуальной для DSM-сообщества информацией. Кроме того, DSM-подход применяется на практике «точечно», то есть редки ситуации, когда одна та же команда, один и тот же разработчик реализовывали и использовали несколько таких решений (и, соответственно, имеют опыт в таких проектах). В большинстве случаев команды используют DSM-подход впервые и нуждаются в опыте других, поэтому информация, изложенная в данной статье, может оказаться им полезной.

В данной статье рассказывается о разработке и применении предметно-ориентированного решения РУП, созданного в рамках проекта по разработке Web-портала, предназначенного для информационной поддержки русских/финских путешественников в Финляндии/России: предоставление информации о публичных и государственных услугах, необходимых при путешествии в другую страну [17, 18]. (Эта задача была частью международного исследовательского проекта, который мы будем называть в этой статье ISS-проектом [19]). Решение предназначено для проектирования контента Web-системы с использованием средства онтологического инжиниринга ОРГ-Мастер [20]; разработка контента для этой системы оказалась задачей, сопоставимой по трудности с разработкой самого ПО, поэтому для сбора и систематизации контента было принято решение использовать онтологический инжиниринг [21, 22]. Однако выбранная технология (ОРГ-Мастер), обладая рядом преимуществ перед другими аналогами (профессиональный продукт, доступность разработчиков и аналитиков и т. д.), тем не менее, не имела наглядных средств проектирования моделей. Поэтому нами было принято решение разработать «легковесный» графический редактор с поддержкой автоматической генерации модели контента и циклической разработки (round-trip engineering) этой модели в РУП-редакторе и ОРГ-Мастере. Поставленная задача была выполнена, созданное решение было успешно использовано в ISS-проекте.

[1] Последнее интересно в силу большого разнообразия видов ПО – телекоммуникационные системы, информационные системы, Интернет-приложения, компиляторы и т. д. Следовательно, и визуальные абстракции во всех этих случаях нужны разные, для того чтобы максимально эффективно отражать специфику этого вида ПО. Например, два близких вида ПО – бортовые управляющие системы и телекоммуникационные протоколы, породили SWITCH-технологию [13], ставшую ядром целого семейства DSM-решений – например, [14, 15] и технологию RTST [16]. Фактически, SWITCH-технология является инструментом для создания DSM-решений в тех ситуациях, где эффективна автоматная парадигма, а RTST представляет собой классический пример DSM-решения, созданного для линейки программных продуктов (ПО телефонных станций). Отметим, что повторное использование DSM-решений затруднено ещё и потому, что при их реализации очень важным являются используемые средства разработки, интеграционные аспекты и пр. технические аспекты, поэтому при кажущемся идейном сходстве, например, SWITCH-технологи и RTST, области их применения различаются.

1. КОНТЕКСТ

1.1. ISS-ПРОЕКТ

Проект Improving Social Services (ISS-проект) [19] проходил в 2011–2013 годах в рамках международной программы ENPI Finnish Russian Cross-Border Cooperation [23] и был ориентирован на поддержку сотрудничества между финскими/русскими гражданами и государственными/публичными российскими/финскими службами в целях упрощения доступа к различной информации, необходимой путешественникам, когда они планируют и осуществляют свои путешествия на территорию соседней страны.

Таким образом, основной целевой аудиторией проекта являлись так называемые *свободные путешественники* – люди, которые путешествуют по соседней стране без поддержки туристических агентств и других организаций. У таких людей в чужой стране могут случаться различные проблемы: ДТП, проблемы со здоровьем, потеря/кража документов, проблемы при самостоятельном бронировании/покупке ж/д и авиабилетов, использование различного вида транспорта в чужой стране и т. д. И разрешать эти проблемы им, в отличие от организованных туристов, приходится самостоятельно. С другой стороны, свободные путешественники не имеют опыта проживания в чужой стране, как, например, иммигранты, и поэтому сталкиваются с большим количеством новых для себя ситуаций.

В проекте приняли участие Санкт-Петербургский государственный университет, Санкт-Петербургский политехнический университет, технологический университет города Лаппеенранта, комитет по информатизации Санкт-Петербурга, информационно-аналитический центр Санкт-Петербурга и другие организации. В рамках проекта была разработана пилотная Web-система [17] и форум путешественников [24], проведено три международных студенческих школы [25], созданы рекомендации для российских органов власти, направленные на улучшение доступа к государственным и публичным услугам для иностранцев.

1.2. РАЗРАБОТКА WEB-ПОРТАЛА ДЛЯ РУССКИХ И ФИНСКИХ ПУТЕШЕСТВЕННИКОВ

Разработка Web-систем на основе онтологий является популярной темой, которая в последнее время привлекает к себе всё больше внимания. Существуют подходы, которые используют онтологии для концептуализации Web-порталов [26], другие идут дальше и предлагают различные варианты автоматической генерации сайтов и порталов по онтологиям [27][1]. В последнем случае, на наш взгляд, принципиальным является поиск и формализация различных метафор визуализации информации, с тем чтобы сайты, созданные на основе онтологий, были не просто наборам html-страниц, соединённых гиперссылками, а содержали более сложные элементы пользовательского интерфейса – иначе пользоваться такими сайтами очень неудобно. Такие элементы интерфейса можно генерировать автоматически по схеме онтологии так же, как генерируют пользовательский интерфейс по схеме базы данных [30, 31].

В ISS-проекте мы не стали использовать системы автоматической генерации сайтов, поскольку перед нами стояли разнообразные задачи, среди которых автоматический переход от моделей к программному коду не был центральной. Вот эти задачи: исследование запросов потенциальных пользователей нашей системы [32, 33], способы поиска и извлечения информации, а также поиск подходящих для нашей предметной области метафор Web-визуализации [34, 35] и, наконец, задача проектирования контента, решению которой посвящены результаты, изложенные в данной статье. Схема процесса разработки Web-системы представлена на рис. 1, а сама Web-система подробно описана в [18].

[1] Ряд интересных case-studies в области автоматизированной разработки сайтов по онтологиям можно найти в [28].

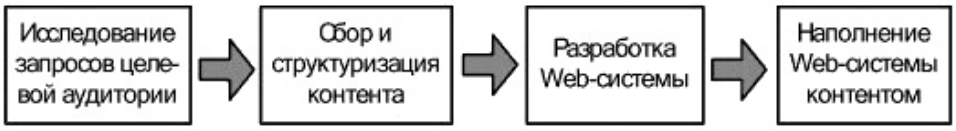


Рис. 1. Схема процесса разработки Web-системы для поддержки русских и финских путешественников

1.3. ТЕХНОЛОГИЯ ОРГ-МАСТЕР

Выбор технологии ОРГ-Мастер [20] как средства проектирования контента был обусловлен тем, что эта технология является рабочим инструментом, а не академической разработкой (большое количество средств онтологического инжиниринга можно найти в [36, 37], однако при использовании таких средств на практике часто не хватает многих элементарных, «не научных» функций; средства архитектурного моделирования предприятий [38] являются более зрелыми, и ОРГ-Мастер входит в их число). Кроме того, было важно, что разработчики программных средств и авторы методологии ОРГ-Мастер оказались доступны, дружелюбны и готовы к сотрудничеству, что также очень важно, так как программные средства такого класса обычно сложны для использования. Наконец, ввиду близости и доступности авторов, удалось легко решить лицензионные вопросы.

ОРГ-Мастер является эффективным средством для моделирования архитектур бизнес-предприятий[1]. За более чем двадцать лет существования и развития технологии с её помощью было выполнено около 500 проектов в России, странах Балтии и ближайшем зарубежье, в частности, для таких организаций, как правительство Москвы, конституционный суд РФ, Газпром и т. д. Технология состоит из методологии бизнес-моделирования, а также набора программных средств. Последние устроены таким образом, что легко делятся на два уровня: средства базового онтологического инжиниринга и средства бизнес-моделирования, причём вторые надстраиваются над первыми. В ISS-проекте оказались востребованы именно средства базового онтологического инжиниринга.

Ниже представлены определения основным конструкциям языка моделирования ОРГ-Мастера; данная информация важна для этой статьи, так как представленное в статье РУП-решение ориентировано на интеграцию с ОРГ-Мастером и, прежде всего, с его языком моделирования.

Классификатор – это контейнер, содержащий дискретные, упорядоченные элементы информации (*позиции*), которые организованы в иерархию и описывают некоторый связный и достаточно однородный фрагмент предметной области.

Классификатор может иметь *атрибуты*, значения которым можно задавать для каждой его позиции. Каждый атрибут имеет тип – либо из списка стандартных (документ, комментарий, целое число и т. д.), либо какой-либо *свободный атрибут*. Последний является повторно используемым перечислимым типом, задаваемым отдельно – его можно использовать в разных атрибутах, для разных классификаторов. В классификатор может входить множество позиций одинакового типа. Кроме того, одна позиция может иметь несколько типов, например, позиция «Директор» в классификаторе «Организационная структура» может иметь тип «Элемент организационной структуры» и «Сотрудник». Наконец, в одном классификаторе могут храниться позиции разных типов.

Домен – это конструкция для разбиения больших моделей на компоненты.

[1] Так называемое архитектурное моделирование предприятия (Enterprise Architecture Management, EA) – дисциплина для активной и целостной реакции компании на разрушительные воздействия посредством описания (идентификации) и анализа изменений в направлении к желаемому бизнес-видению и результатам. EA приносит пользу через предоставление бизнес- и IT-руководителям зрелых рекомендаций для урегулирования практик и проектов, с тем чтобы достигнуть целевых бизнес-показателей. EA используется для управления процессом принятия решений посредством эволюции будущего состояния архитектуры компании [40].

Проекция – это отношение между классификаторами, в рамках которого могут устанавливаться *связи* между позициями этих классификаторов.

1.4. MICROSOFT VISIO

Мы выбрали в качестве DSM-платформы для разработки РУП-решения пакет Microsoft Visio [28], потому что последний позволяет легко создавать многофункциональные графические средства – получаемые редакторы оказываются надстройкой над Visio и позволяют использовать всю его функциональность (многостраничность, печать, множество небольших, но полезных функций пользовательского интерфейса – уменьшение/увеличение, передний/задний план и т. д.). Кроме того, Visio имеет полноценный SDK, обеспечивающий программный доступ к свойствам фигур, страниц, палитры и другим объектам диаграмм, а также он позволяет создавать программные обработчики для многочисленных событий – перемещения и изменения фигур, открытие/закрытие диаграмм и т. д. Код редакторов можно создавать на встроенном в Visio языке VBA (Visual Basic for Applications – урезанная версия языка программирования Visual Basic), а также в среде Microsoft Visual Studio (при создании проекта нужно выбрать соответствующие опции – тогда подключится и станет доступным Visio SDK). В данном проекте был выбран VBA, поскольку разрабатываемый редактор был небольшим. Однако модуль интеграции с ОРГ-Мастером был создан на языке C# по причинам больших удобств при разработке (среда разработки Microsoft Visual Studio).

Ещё одной причиной выбора Visio как платформы для разработки РУП-решения было то, что мы исходно начали создавать модели в Visio, ещё не собираясь разрабатывать специального программного решения.

2. МОТИВАЦИЯ РАЗРАБОТКИ РУП-РЕШЕНИЯ

Выбрав в качестве средства проектирования контента нашей Web-системы ОРГ-Мастер, наша рабочая группа столкнулась с проблемой – моделировать оказалось неожиданно трудно. Дело в том, что ОРГ-Мастер является профессиональным средством, рассчитанным на опытных аналитиков. И для того чтобы успешно его использовать, требуется значительное время на обучение, причём речь идёт не просто об освоении программного средства, а об овладении уникальной (и местами загадочной) техникой моделирования, реализованной в продукте. Кроме того, нам требовалось ещё выработать свой собственный способ применения ОРГ-Мастера именно для нашей предметной области – она существенно отличалась от тех проектов, где ОРГ-Мастер успешно использовался.

Первые наши опыты работы с ОРГ-Мастером проходили с большими трудностями.

- Мы тратили очень много времени, чтобы понять, кто из нас и что вставил в модель и почему он сделал именно так.
- Было также неясно, как именно разбивать информацию на разные классификаторы, как использовать атрибуты и проекции.
- Мы не понимали, сколько моделей ОРГ-Мастера у нас должно получиться в итоге – по одной на каждую подобласть или одна общая.
- Наконец, была большая неясность со средствами типизации однородной информации – далеко не сразу мы поняли, как следует пользоваться типами ОРГ-Мастера.

Для преодоления трудностей с проектированием модели мы начали использовать Visio. Вначале мы делали это стихийно, стремясь предварительно классифицировать найденную информацию и одновременно стараясь, чтобы наши абстракции моделирования имели ясные проекции в язык моделирования ОРГ-Мастера. Разбирая наши первые эскизы с опытными аналитиками, мы проясняли многочисленные неясности и восполняли наши пробелы в практическом использовании ОРГ-Мастера. Но одновременно наши модели разрастались, и оказалось, что вносить в них изменения становится все труднее – нам понадобились составные фигуры, имеющие несколько областей для текстового ввода, размеры этих фигур прихо-

дилось часто менять «вручную» (информация вставлялась, удалялась, исправлялась), точки соединения фигур с линиями терялись и т. д. Вместе с тем у нас кристаллизовался сам язык моделирования, который в конце концов превратился в язык составления схем моделей ОРГ-Мастера. Два этих обстоятельства – трудности в создании спецификации в обычном Visio и сформировавшийся визуальный язык – подтолкнули нас к разработке собственного графического редактора.

3. ПРЕДМЕТНО-ОРИЕНТИРОВАННЫЙ ЯЗЫК РУП

Созданный нами язык состоит из трёх следующих частей: доменная модель, модель типов, модель предметной области. Абстрактный синтаксис языка РУП представлен на рис. 2[1].

На этом рисунке все три модели разделены и содержатся в отдельных прямоугольниках, однако очевидно, что это разграничение носит довольно условный характер – например, класс «Классификатор» вошёл в две разные модели (в доменную и в модель подобласти). Кроме того, между моделью типов и моделью подобласти имеются тесные связи, выраженные с помощью ассоциаций. Однако разделение на модели удобно, так как упорядочивает метамодель и повышает понимаемость абстрактного синтаксиса (на практике абстрактный синтаксис представляет трудность для понимания специалистами, например, экспертами предметной области, для которой создаётся DSM-решение). Кроме того, это разделение реализовано также в редакторе – палитра с нотацией разделена на три закладки.

Остановимся детальнее на каждой модели языка РУП.

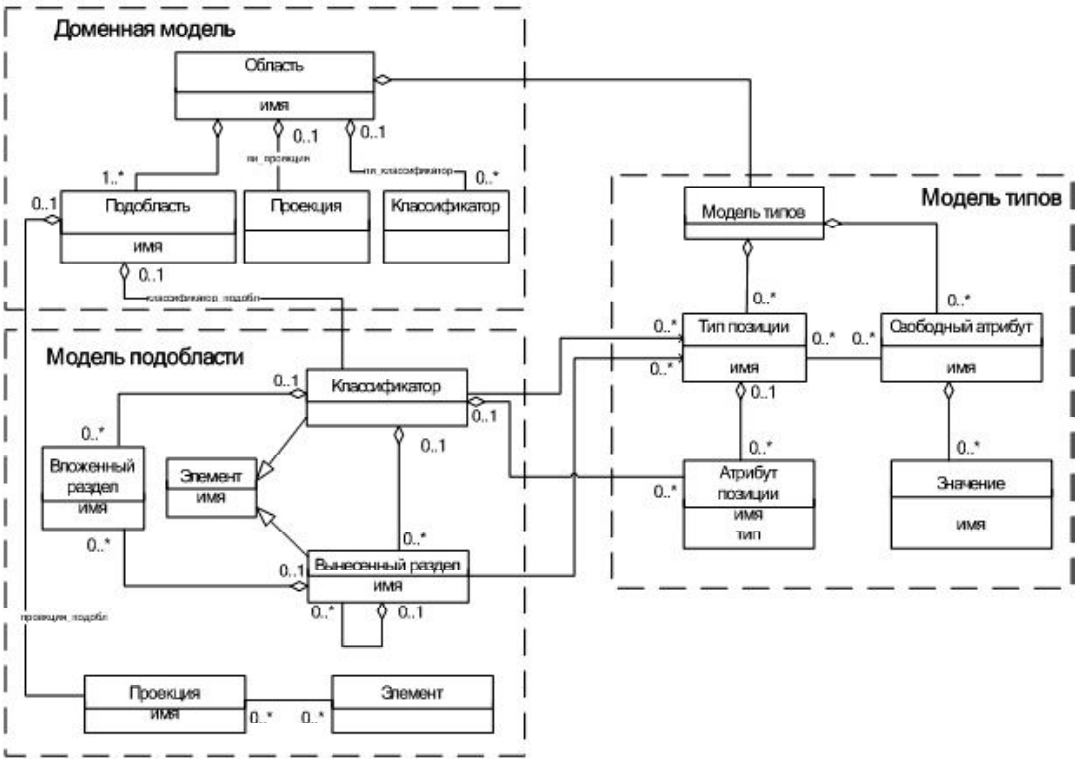


Рис. 2. Абстрактный синтаксис языка РУП

[1] При разработке этой метамодели мы следовали с небольшими отклонениями стандарту MOF [41], хотя невысокая сложность нашей метамодели позволила нам фактически использовать простейшее подмножество диаграмм классов UML. Более строгое следование какому-либо определённому стандарту или формализму (например, Ecore [4] или GOPPRR [1, 7]) при разработке языка оправдано, если метамодель получается большой или если предполагается её автоматическая обработка, то есть генерация на её основе графического редактора.

3.1. ДОМЕННАЯ МОДЕЛЬ

Данная модель является средством для задания схемы предметной области в терминах крупных компонент – подобластей, которые соответствуют доменам в ОРГ-Мастере. На рис. 3 представлен пример фрагмента схемы нашей предметной области «Русско-финское приграничное взаимодействие», выполненный с помощью РУП-языка и включающий в себя следующие подобласти: «В Финляндию на машине», «Туризм и шопинг в Финляндии», «Пересечение границы с Финляндией», «Образование в Финляндии».

Корнем иерархии метамодели РУП является класс «Область». Этот класс имеет единственный атрибут – имя. Так, имя нашей области – «Русско-финское приграничное взаимодействие». Одна область может содержать много подобластей (минимально – хотя бы одну), модель типов, а также классификаторы и проекции. Если область, минуя подобласти, содержит классификаторы и проекции (эта возможность в метамодели РУП осуществляется посредством ассоциаций «пи_классификатор» и «пи_проекция»), то последние могут использоваться в разных подобластях.

3.2. МОДЕЛЬ ТИПОВ

Даная модель является подмножеством диаграмм классов UML и используется для спецификации типов позиций классификаторов. На рис. 4 представлен пример типов –

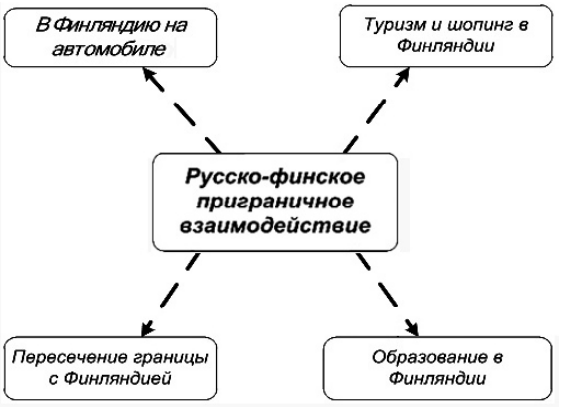


Рис. 3. Пример доменной модели

тип «Маршрут», используемый при описании системы транспорта Петербурга и городов юго-восточной Финляндии. Различные виды маршрутов наследуют от этого обобщённого маршрута.

Мы не стали включать описание ассоциаций и наследования в метамодель РУП, так как это делается стандартно (см., например, метамодель языка UML), а с другой стороны, такое описание сильно загромоздило бы нашу метамодель, сделав её существенно менее понятной.

3.3. МОДЕЛЬ ПОДОБЛАСТИ

В данной модели в терминах классификаторов и проекций описываются подобласти, заданные в доменной модели. На рис. 5 приведён пример схемы подобласти. В центре диаграммы находится подобласть, пря-

Рис. 4. Пример модели типов

моугольники с серыми заголовками – это классификаторы, входящие в данную подобласть, прямоугольники с прозрачными заголовками (в этом примере их всего два: «Возможности» и «Информация для иностранцев») – это сложно устроенные разделы классификаторов, которые вынесены в отдельные сущности на схеме. На этом рисунке показано также, что классификаторы «Возможности повышения квалификации» и «Образовательные учреждения» соединены проекцией – то есть каждой возможности соответствует один или несколько типов образовательных учреждений, которые могут эту возможность реализовать.

Вложенный раздел – это позиция классификатора, которая существенно характеризует содержание классификатора и потому попадает на РУП-модель в виде свойства классификатора. Если же позиция является очень значимой и на РУП-диаграмме целесообразно показать её детальную структуру, то используется конструкция «Вынесенный раздел», которая изображается отдельным прямоугольным символом. Вложенный раздел также характеризует классификатор и важен с точки зрения проектирования контента, но его внутренняя структура не существенна, не важна, и никаких связей с другими сущностями у него нет. Поэтому он изображается в тестовом виде внутри классификатора[1]. Так, на рис. 5 у классификатора «Получение квалификации» позиции «Дошкольное образование», «Общее образование» и пр. являются внутренними разделами, а позиции «Возможности» и «Информация для иностранцев» – вынесенными разделами.

4. ГРАФИЧЕСКИЙ РЕДАКТОР

Рассмотрим более детально разработанный РУП-редактор. Его главное окно представлено на рис. 6 *а*. Список реализованной функциональности представлен ниже.

- Поддержка множества диаграмм.
- Раздельная поддержка трёх нотаций.
- Поддержка диалогов для ввода/редактирования.

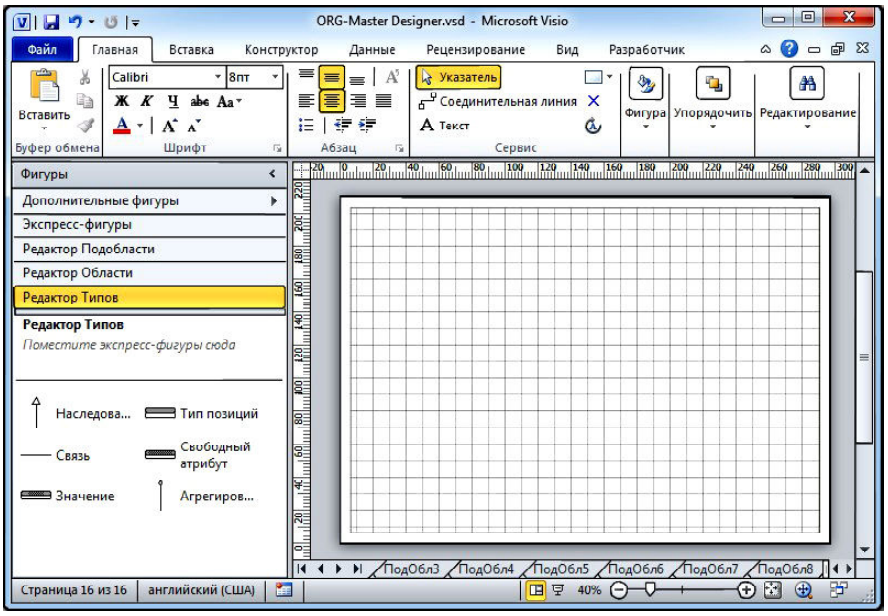
Рис. 5. Пример модели подобласти

[1] Отметим, что необходимость при верхнеуровневом проектировании видеть структуру разделов, в общем-то, и приводит к тому, что такие разделы изображаются в виде отдельных графических символов – в противном случае пришлось бы изображать иерархическое дерево (пусть даже минимальной глубины два) в текстовом виде внутри символа классификатора, что очень неудобно как в реализации, так и в использовании.

- Поддержка сложных фигур.
- Поддержка целостности модели.
- Средства пакетирования решения.

Остановимся подробно на поддержке целостности модели. Эта задача была особенно актуальна в связи с наличием в модели трёх видов диаграмм: во-первых, было необходимо поддержать разделение нотаций (об этом было рассказано выше); во-вторых, на диаграммах разных видов информация частично дублировалась, и в связи с этим было важно, чтобы были средства, препятствующие её рассинхронизации. Так при связывании типа позиции с классификатором мы реализовали выбор из существующих, заданных в модели типов: новый тип в этом диалоге задать нельзя, можно только выбрать из существующих, то есть типов, ранее созданных на диаграммах. Пример соответствующего диалога представлен на рис. 6 *б*. Кро-

а)



б)

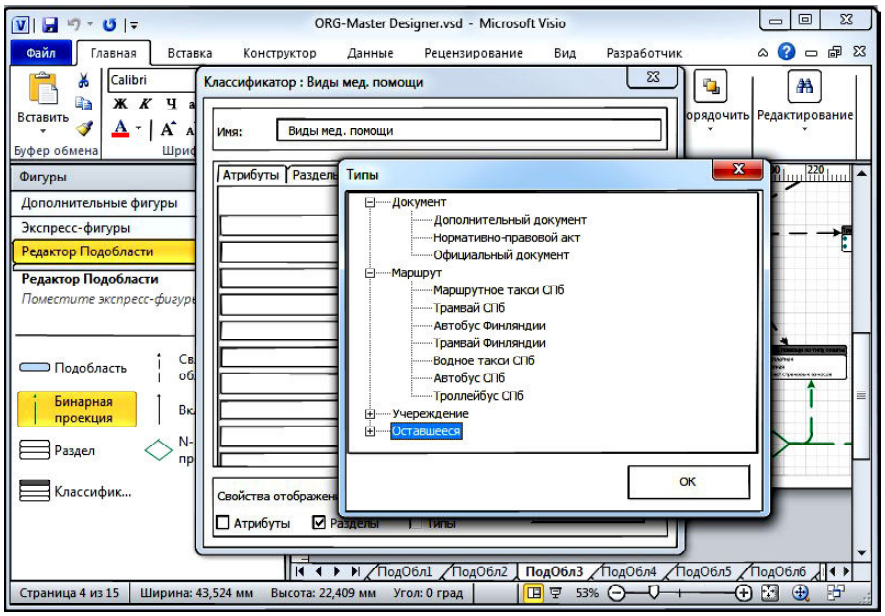


Рис. 6: *а)* главное окно РУП-редактора;
б) пример диалога выбора типов позиций классификатора.

ме того, наш редактор контролирует, что проекцией нельзя связать корректность связей – например, он не позволяет связать подобласть и классификатор с помощью проекции. Контролируется также наличие не более одной области/подобласти на диаграмме.

5. СРЕДСТВА ГЕНЕРАЦИИ И СИНХРОНИЗАЦИИ

В РУП-решении мы реализовали генерацию модели ОРГ-Мастера по РУП-модели, а также синхронизацию из РУП-редактора в ОРГ-Мастер. Обратной генерации (из спецификации Visio по модели ОРГ-Мастера) реализовано не было, ввиду экономии ресурсов разработки. Так, в частности, пришлось бы решать задачу автоматической раскладки различных графов.

Поддержка циклической разработки (round trip engineering) является важной чертой модельно-ориентированных решений, не имея этой поддержки, такие решения оказываются трудными в эксплуатации или даже бесполезными.

Следуя [29], будем понимать под *циклической разработкой* работу с электронной информацией посредством двух программных продуктов (например, РУП-редактора и ОРГ-Мастера), каждый их которых работает со своей моделью (таким образом, общая информация составляется из двух этих моделей, но они имеют пересечения!). При этом если одна модель меняется пользователем, то соответствующие изменения автоматически попадают в другую модель, то есть происходит синхронизация. Важной характеристикой синхронизации является то, что она меняет модель во втором продукте лишь локально, в тех местах, где изменилась соответствующая информация в первой модели (change propagation). А остальные части второй модели остаются без изменений. Если просто перегенерировать вторую модель по первой, то будет потеряна информация, которая содержится только во второй модели и отсутствует в первой.

При работе с моделью РУП-редактор сохраняет данные как в vsd-файле, так и в модели ОРГ-Мастера. Это делается при стандартном сохранении Visio (например, при нажатии клавиш Ctrl-S). При этом менять модель из ОРГ-Мастера в тот момент, когда с нею происходит работа из РУП-редактора, нельзя. Таким образом, мы легко получаем одностороннюю синхронизацию из РУП в ОРГ-Мастер.

6. ПРОЦЕСС РАЗРАБОТКИ И ИСПОЛЬЗОВАНИЯ

При разработке нашего решения мы ориентировались на модель процесса из [10]. Мы приступили к созданию редактора, после того как зафиксировали метамодель языка РУП. Сама разработка метамодели сопровождалась сбором информации для контента Web-системы и созданием соответствующих спецификаций в Visio, а также изучением технологии ОРГ-Мастер. Последнюю мы изучали, так сказать, в «полевых условиях», прямо в рамках нашего проекта, ввиду нехватки времени. Важно отметить, что мы не начинали разработку редактора и метамодели до тех пор, пока не почувствовали удовлетворения от наших моделей и получившегося языка. Сформировавшийся язык мы зафиксировали метамоделью. Создавать метамодель, и тем более, редактор в ситуации, когда не ясны основные абстракции моделирования, очень неэффективно: переделки занимают существенное время и в целом могут замедлить процесс разработки.

Очень важно правильно определить момент, когда следует перейти от экспериментов с языком к созданию метамодели: здесь важно не торопиться, но и не затягивать – язык всё равно, скорее всего, будет ещё меняться, но его каркас должен быть сформирован, и именно его фиксирует метамодель. И эта метамодель реализуется в редакторе, а после того, как в пробной версии редактора создаётся несколько примеров, язык уточняется. Подчеркнём, что в этом итеративном процессе итерируется (то есть уточняется) малая часть языка, а большую предлагается

разработать и зафиксировать в первой версии метамодели.[1]

В процессе создания языка мы активно разрабатывали целевые модели, и последние не были примерами – мы пробовали удачно изобразить, представить собираемую для контента нашей Web-системы информацию. Этот способ хорошо себя зарекомендовал, но потребовал большого количества рутинной работы: достаточно быстро объем спецификаций зафиксировался по объёму и информационному охвату и после этого изменялся незначительно, а при изменениях в языке приходилось вносить многочисленные изменения во все созданные с его помощью спецификации. В связи с этим пользы РУП-решения было меньше, чем могло бы быть, так как много работы оказалось уже сделано к моменту его появления. Тем не менее, остался большой объем небольших локальных изменений, и с этим было бы трудно справиться без специального редактора. При этом оказался полезным механизм циклической разработки, так как именно при незначительных изменениях, когда уже была сгенерирована по РУП-модели модель в ОРГ-Мастере и последняя стала активно заполняться, механизм циклической разработки и соответствующая дисциплина не позволили модельным артефактам ОРГ-Мастера и РУП оказаться в состоянии рассогласованности. В таких ситуациях обычно жертвуют верхнеуровневыми моделями, которые приобретают демонстрационный статус.

Разработка решения заняла около 4-х человеко-месяцев. Ею руководил аналитик, который был ответственен за предметную область и одновременно являлся главным пользователем редактора, разбирался в визуальном моделировании (создавал метамодель и генерационные проекции) и разработке ПО. Последнее важно, так как аналитик был способен сформулировать ясные и выполнимые требования к решению. Кроме того, в работе нам помогал главный архитектор программных средств ОРГ-Мастера, консультируя нас в вопросах интеграции. Без помощи аналитика и архитектора нам не удалось бы в столь сжатые сроки получить полноценное решение. Отметим, что сочетание разных экспертиз и ролей по отношению к создаваемому решению в лице аналитика ускорило разработку решения. Обычно на практике эти экспертизы оказываются рассредоточенными по разным людям, что замедляет разработку и приводит к большему количеству проб и ошибок: как правило, DSM-решение обладает существенной новизной, что влечёт за собой соответствующие риски – такого продукта либо вообще никто не делал, либо этот проект является новым для коллектива[2]. В данном случае этих рисков удалось избежать.

Разработка РУП-решения велась итеративно – всего было произведено семь итераций, два раза пришлось полностью перерисовать все диаграммы, общее количество диаграмм составило 15 штук формата А4.

Важно отметить, что мы тщательно следили за тем, чтобы не заниматься реализацией избыточной функциональности. Мы безжалостно отметали пожелания «хорошо бы….» или «так правильно….. », если за ними не стоял реальный запрос именно со стороны нашего проекта. Так, мы отказались от реализации следующей функциональности, которая обычно входит в графические средства.

- Мы не стали реализовывать двустороннюю циклическую разработку, хотя воз-

[1] Мы не готовы дать формальных критериев для определения момента, когда следует фиксировать язык метамоделью и начинать реализовывать редактор. Тем более, что когда используется генеративная DSM-платформа, автоматически создающая по метамодели каркас редактора, ситуация отличается от той, которая была в нашем проекте – быстро получить редактор и попробовать созданные абстракции на примерах важно и очень удобно. Тем не менее, и в этом случае период создания зрелой первой версии должен присутствовать; преждевременная формализация вообще вредна, даже безотносительно к немедленной разработке редактора, так как сковывает незрелую мысль и препятствует необходимым обновлениям. В целом, для каждого проекта на практике можно определить тот момент, когда уже достаточно собрано информации о предметной области и о языке и можно переходить к формализации.

[2] Этот вид рисков можно добавить к соответствующему списку рисков и трудностей при разработке DSM-решений, представленному и подробно обсуждаемому в работе [10].

можность менять схему модели прямо из ОРГ-Мастера с отражением появившихся классификаторов и проекций в РУП-редакторе была бы, без сомнения, удобна.

- Мы не реализовали функцию загрузки элемента на диаграмму. Анализ отношения количества повторно используемых элементов в нашей модели (нормированное затратами на поддержку «ручной» синхронизации) к затратам на реализацию этой функциональности, а также имевшиеся у нас ресурсы (люди, время) на разработку РУП-решения привели нас к выводу, что эффективнее оставить эту функциональность нереализованной и выполнять соответствующую работу «вручную».
- В нашем редакторе не реализована полная поддержка целостности, например, контроль того, что во всей модели существует только одна область, а также того, что в модели нет повторяющихся подобластей. Всего таких сущностей в нашей модели меньше десяти, так что поддержка целостности «вручную» здесь не представила затруднений.
- Мы не стали тратить усилий на разработку инсталляционного пакета и ряда других возможностей, облегчающих развертку решения, поскольку пользователей у него было не много.

В итоге РУП-решение было разработано вовремя и активно использовалось при работе с моделью контента, позволяя нам решить наши главные проблемы: легко вносить изменения и создавать красивые диаграммы (для использования в проектных презентациях и отчётах).

ЗАКЛЮЧЕНИЕ

Мы не преувеличиваем значимость созданных нами средств – их использование вне рамок нашего проекта ограничено. Но это полностью соответствует философии DSM-подхода – лёгкость разработки нового языка и поддерживающих его программных средств, точность в удовлетворении нужд целевого проекта и отсутствие универсальности и ориентации на использование в рамках другого контекста. Во многом, в силу последнего обстоятельства, разработка решения не требует больших ресурсов и становится возможной. При этом при всей кажущейся универсальности такие решения всё же соединены (и часто незримой связью) с контекстом, их породившим. Тем не менее, предложенные нами средства могут быть использованы как прототип инструментов для проектирования моделей ОРГ-Мастера – когда такие инструменты начнут разрабатываться, то наш опыт может оказать полезным. Проделанная нами работа может также служить примером быстрой разработки визуальных средств под конкретные нужды, чтобы другие люди, столкнувшись с подобными задачами, могли адекватно оценить свои перспективы и необходимые для их осуществления трудозатраты.

Хотелось бы выразить признательность Илье Алексееву и Евгению Калугину, принявшим активное участие в разработке РУП-решения, а также Татьяне Лебедковой, Елене Шакировой и Надежде Квас, которые создавали с помощью РУП-решения и ОРГ-Мастера контент для Web-системы [17, 19].

Литература

1. *Kelly S., Tolvanen J.-P.* Domain-Specific Modeling: Enabling Full Code Generation. John Wiley & Sons. 2008.

2. *Thomas D.A.* MDA: Revenge of the Modelers or UML Utopia? IEEE Software 21(3), 2004. P. 15–17.

3. *Demeyer S., Ducasse S., Tichelaar S.* Why unified is not universal: UML shortcomings for coping with round-trip engineering / Proceeding UML'99 Proceedings of the 2nd international conference on The unified modeling language: beyond the standard. P. 630–644.

4. *Сорокин А.В., Кознов Д.В.* Обзор Eclipse Modeling Project // Системное программирование, 2010. Т. 5, № 1. С. 6–32.

5. MetaEdit+, MetaEdit+, http://www.metacase.com/ (дата обращения 28.10.13).

6. *Кузенкова А.С., Дерипаска А.О., Таран К.С., Подкопаев А.В., Литвинов Ю.В., Брыксин Т.А.* Средства быстрой разработки предметно-ориентированных решений в MetaCASE-системе QReal //

Научно-технические ведомости Санкт-Петербургского государственного политехнического университета. Информатика. Телекоммуникации. 2011. Т. 4, № 128. С. 142–145.

7. *Павлинов А.А., Кознов Д.В., Перегудов А.Ф., Бугайченко Д.Ю., Казакова А.С., Чернятчик Р.И., Иванов А.Н.* О средствах разработки проблемно-ориентированных визуальных языков // Системное программирование. 2006. Т. 2, № 1. С. 116–141.

8. *Сухов А.О.* Инструментальные средства создания визуальных предметно-ориентированных языков моделирования. Фундаментальные исследования, 2013. № 4-4. С. 848–852.

9. *Поляков В.А., Брыксин Т.А.* Подходы к заданию семантики интерпретации диаграмм в рамках DSM-подхода. Системное программирование. Том 7, вып. 1: Сб. статей / Под ред. А.Н.Терехова, Д.Ю. Булычева. СПб.: Изд-во СПбГУ, 2012. С. 187–217.

10. *Koznov D.* Process Model of DSM Solution Development and Evolution for Small and Medium-Sized Software Companies / Proceedings 15th IEEE Internationa EDOC Conference Workshops. 2nd International Workshop on Models and Model-driven Methods for Service Engineering (3M4SE 2011). 29 August – 3 September, 2011. Helsinki. P. 85–92.

11. *Кознов Д.В.* Разработка и сопровождение DSM-решений на основе MSF. Системное программирование. Том 3, вып. 1: Сб. статей / Под ред. А.Н.Терехова, Д.Ю.Булычева. СПб.: Изд-во СПбГУ, 2008. С. 80–96.

12. *Hermans F, Pinzger M, van Deursen A.* Domain-Specific Languages in Practice: A User Study on the Success Factors. MODELS 2009, LNCS 5795. P. 423–437.

13. *Гуров В.С., Мазин М.А., Нарвский А.С., Шалыто А.А.* UML. SWITCH-технология. Eclipce // Информационно-управляющие системы, 2004. № 6. С. 12–17.

14. *Альтерман И.З., Шалыто А.А.* Промышленные АСУ и контроллеры. 2005. № 10. С. 49–52.

15. *Казаков М.А., Корнеев Г.А., Шалыто А.А.* Разработка логики визуализаторов алгоритмов на основе конечных автоматов // Дистанционное и виртуальное обучение, 2003. № 6. С. 27–58.

16. *Терехов А.Н.* RTST-технология программирования встроенных систем реального времени / Записки семинара Кафедры системного программирования «CASE-средства RTST++». 1998. С. 3.

17. ISS Web-system, http://109.124.104.91:8072/News/RelatedNews/1 (дата обращения 28.10.13).

18. *Азарсков А., Нурулин Ю., Самочадин А.* Система визуализации государственных и общественных социальных сервисов: функциональные возможности. БХВ, 2013.

19. Improving Social Services, South-East Finland-Russia ENPI CBC 2007–2013, Project 2010–021–SE396, 2011–2013, https://sites.google.com/site/improvingsocialservices/ (дата обращения 28.10.13).

20. *Григорьев Л.Ю., Кудрявцев Д.В.* Организационное проектирование на основе онтологий: методология и система ОРГ-Мастер // Научно-технические ведомости Санкт-Петербургского государственного политехнического университета. Информатика. Телекоммуникации. 2012. Т. 1, № 140. С. 21–28.

21. *Гаврилова Т.А.* Об одном подходе к онтологическому инжинирингу // Новости искусственного интеллекта, 2005. № 3. С. 25–31.

22. *Гаврилова Т.А.* Логико-лингвистическое управление как введение в управление знаниями // Новости искусственного интеллекта. 2002. № 6. С.28–33.

23. South-East Finland-Russia ENPI CBC, http://www.southeastfinrusnpi.fi/ (дата обращения 28.10.13).

24. Cross-Border Travel: Problems free traveling across the Finnish-Russian border, www.crossbordertravel.eu (дата обращения 28.10.13).

25. ISS Student Schools, http://www.math.spbu.ru/user/dkoznov/iss.htm (дата обращения 28.10.13).

26. *Гаврилова Т.А., Гелеверя Т.Е., Горовой В.А.* Онтологии как средство концептуализации web-порталов // Искусственный интеллект, 2002. № 3. С. 80–86.

27. *Domingue J., Fensel D., Hendler J.A.* Handbook of Semantic Web Technologies. Springer. 2011.

28. http://www.w3.org/2001/sw/sweo/public/UseCases/ (дата обращения 28.10.13).

29. Microsoft Visio, http://office.microsoft.com/ru-ru/visio/ (дата обращения 28.10.13).

30. *Sendall S., Küster J.* Taming Model Round-Trip Engineering. Proceedings of Workshop on Best Practices for Model-Driven Software Development (2004). P. 1–10.

31. *Иванов А.Н., Стригун С.С.* Технологическое решение REAL-IT: автоматизированная разработка пользовательского интерфейса информационных систем. Системное программирование. 2005. Т. 1, № 0. С. 124–147.

32. *Иванов А.Н.* Технологическое решение REAL-IT: создание информационных систем на основе визуального моделирования. Системное программирование. 2005. Т. 1, № 0. С. 89–100.

33. *Самочадин А. В., Кознов Д. В., Морозов А. В., Аллахвердиев Э. Р., Шакирова Е. Р., Хамаева Э. А., Лебедкова Т.А.* Метод выявления и ранжирования востребованности информации средствами Интернета // Системное программирование, 2012. Т. 7, № 1. С. 106–136.

34. *Морозов А.В., Кознов Д.В., Самочадин А.В.* Статистическое исследование проблем российских туристов при путешествиях в Финляндию // Системное программирование, 2012. Т. 7, № 1. С. 137–160.

35. *Кознов Д. В., Азарсков А. В., Самочадин А. В., Шевцова Ю. А., Романовский К. Ю.* Модельно-ориентированный метод спецификации государственных услуг // Вестник Санкт-Петербургского Университета. Сер. 10, вып. 4. 2012. С. 103–117.

36. *Самочадин А.В., Нурулин Ю.Р.* Информационная поддержка публичных услуг. СПб.: БХВ-Петербург, 2013.

37. *Кудрявцев Д.В.* Системы управления знаниями и применение онтологий. Учебное пособие; Санкт-Петербургский государственный политехнический университет. Санкт-Петербург, 2010.

38. http://www.w3.org/2001/sw/wiki/Tools (дата обращения 28.10.13).

39. *Berneaud M., Buckl S., Diaz-Fuentes A., Matthes F., Monahov I., et al.* Trends for Enterprise Architecture Management Tools Survey. IBM report, May 16, 2012, http://www-05.ibm.com/at/events/21_10_roundtable/pdf/tools.pdf (дата обращения 28.10.13).

40. Gartner, IT Glossary, http://www.gartner.com/it-glossary/enterprise-architecture-ea/ (дата обращения 28.10.13).

41. OMG Meta Object Facility (MOF) Core Specification. Version 2.4.1 OMG, 2013.

VISUAL MODELING SOLUTION IN DOMAIN ANALYSIS OF INFORMATION WEB-SERVICE

Abstract

Domain specific modeling addresses to development of efficient visual modeling solutions for particular domains and even for single software projects. But there is a lack of systematic approaches in this area. Therefore, research community is interesting in information about successful domain specific solutions. In the paper, the visual modeling solution for development of information e-service is presented. The main focus of the solution is content modeling of an information e-service. The content is quite complicated and rather big, and its structure directly influences into user interface and features of the software. We used enterprise architecture management tool ORG-Master to collect and structure information for the content. But it appeared the tool does not support visual modeling of the content structure (it was quite natural for it because it is oriented for other tasks). To close this gap we developed small visual language, graphical editor on the base of Microsoft Visio, generator to ORG-Master and synchronizator for round-trip engineering.

The paper describes the solution lifecycle: problems, which have led to idea of using domain specific modeling, selecting functionality to meet time and financial constraints, and other practical questions, which we resolved to use domain specific modeling successfully.

Keywords: domain-specific modeling, enterprise architecture management, ontology engineering, Microsoft Visio, e-service, information e-service.

Кознов Дмитрий Владимирович, кандидат физико-математических наук, доцент кафедры системного программирования математико-механического факультета СПбГУ, dkoznov@yandex.ru.