2004 - Том 7 - Выпуск 2

Проектирование информационных веб-порталов в ИСИР

Бездушный А.Н., Михеев П.Н.
ВЦ РАН

Меденников А.М.
ЦНТК РАН

В статье рассматривается методика проектирования информационных порталов, становление которой осуществляются в рамках технологии ИСИР. Методика основана на необходимости улучшения как методологического, так и технического обеспечения процесса проектирования и построения информационных систем. В статье рассматриваются исходные посылки, требования к технологии. Приводится обзор ряда распространенных средств автоматизированного построения информационных систем, дан их сравнительный анализ. Представлен принятый подход к решению задачи, сформулированы его основные методологические и архитектурные положения. Рассматриваются инструменты моделирования прикладного уровня и уровня представления. Описываются реализация инструментальной среды генерации средств представления информации, обеспечения взаимодействия с сервером приложений для платформы Microsoft .NET , включая Microsoft SharePoint Portal Server 2003 и Microsoft Content Management Server 2002.

Введение

В одном из предыдущих номеров журнала [ISIR] была опубликована статья об архитектуре системы ИСИР. Эта система была использована на практике в ряде проектов по созданию веб-порталов. Опыт реализации этих проектов показывает, что, несмотря на существенное сокращение объемов работ при использовании готовой платформы для построения приложений, существует несколько видов задач, для решения которых приходится прикладывать значительные усилия. Из них в первую очередь следует отметить две:

· смена дизайна портала

· создание стандартного интерфейса пользователя для каждого типа ресурса.

Поясним суть этих задач. Первая заключается в необходимости соблюдения общего дизайна для всех страниц портала. Причем, если в начале процесса разработки приложения сделать это относительно несложно, то после реализации даже его части замена дизайна может оказаться довольно непростой задачей. А

подобная ситуация возникает довольно часто, так как разработка самого приложения и его дизайна ведутся обычно параллельно и независимо друг от друга.

Вторая задача состоит в том, чтобы по результатам моделирования предметной области оперативно построить фрагмент приложения, обеспечивающий минимальную функциональность работы с данными: создание и удаление объектов, просмотр и изменение их свойств, поиск объекта заданного типа по значениям его атрибутов. Такая функциональность обычно реализуется по шаблону, и было бы очень удобно, если бы процесс преобразования шаблонов в исполняемый код был автоматизирован.

Отдельно следует остановиться на вопросе интеграции технологии ИСИР с другими портальными решениями. В соответствии с архитектурой ИСИР, уровень представления взаимодействует с репозиторием данных по строго определенным интерфейсам. Следовательно, любое приложение, поддерживающее этот интерфейс, может использовать репозиторий для работы с данными, обеспечивая собственное представление. Примерамитакихприложениймогутслужить Microsoft SharePoint Portal Server, Microsoft Content Management Server или ORACLE Portal. Предоставляя готовое портальное решение, эти продукты позволяют расширять его путем создания отдельных модулей (портлетов, веб-частей и т. п.), реализующих пользовательский интерфейс к внешней информационной системе. Поэтому при создании технологии проектирования и автоматической генерации интерфейса пользователя следует учитывать, что вместо полнофункционального web-портала иногда необходимо получить набор компонентов для использования в портальном решении третьей стороны.

Таким образом формулируется следующая цель: создать технологию, позволяющую автоматизировать процесс построения приложений на основе ИСИР и обладающую рядом свойств, перечисленных ниже

1) Технология должна включать в себя инструменты моделирования и соответствующие языки, позволяющие моделировать как прикладную логику работы с информацией, так и механизмы ее представления конечному пользователю. При этом мы ограничиваемся объектом моделирования, представляющим собой информационную систему, созданную на основе технологии ИСИР.

2) Инструменты моделирования должны включать такой набор средств, которой позволит создавать модели с минимальным дублированием информации. Языки спецификации моделей должны предоставлять возможность повторного использования фрагментов моделей без их прямого копирования.

3) Инструменты моделирования не дожны содержать понятий, а языки спецификации моделей – синтаксических конструкций, специфических для определенной среды выполнения приложений.

Обзор аналогичных решений

Приступая к решению поставленной задачи, необходимо было познакомиться с имеющимися идеями, подходами и средствами моделирования информационных web-систем (сайтов, порталов и т.д.), инструментальными средами их автоматизированной разработки, использующими результаты соответствующего моделирования. Было важно оценить инструментарии моделирования, соотнеся их с нашими требованиями. Рассмотрим несколько наиболее интересных решений.

WebML

В Web Modeling Language (WebML) [WebML] процесс моделирование охватывает все важнейшие составные части информационных web-систем: описание структур данных системы (Data Model), базовый уровень прикладной логики - определение видов допустимых операций манипулирования данными (Content Management Model), спецификацию визуального интерфейса системы, обеспечивающего как просмотр информации и навигацию по ней, так интерактивное взаимодействие системы с пользователем (Hypertext Model). На уровень моделирования даже вынесено управление пользователями (User and Group Model) и управление доступом к данным (Access Model).

Концепция WebML предоставляет средства визуального моделирования. Графический язык моделирования позволяет разработчикам сформировать необходимый набор моделей в виде разнообразных спецификаций и диаграмм, сохраняемых в собственном XML-формате. А затем на их основе осуществить автоматизированное построение и сопровождение веб-системы/приложения. Эти возможности предоставляются средой визуального проектирования и реализации WebRatio (http://www.webratio.com).

Моделирование данных (Data Model), формирование **модели данных предметной области** многом аналогично ER-моделированию. Разработчик может определить набор идентифицируемых сущностей предметной области и простых бинарных связей между ними. Можно указать ограничения на степень связи. Понятие классов и подклассов (is-a связи семантического моделирования или отношения наследования ОО моделирования) не введено. Атрибуты сущностей могут быть только элементарных типов.

Для моделирования просмотра информации, навигации между ее отдельно визуализируемыми частями в рамках построения так называемой **модели гипертекста** (Hypertext Model) введены такие понятия, как страницы, их текстовые и графические элементы, навигационные связи (Navigational Links), «секции данных» (Content Units), обеспечивающие взаимодействие с элементами модели данных предметной области. К «секциям данных» относятся: отображение экземпляра сущности (DataUnit), отображение всей совокупности экземпляров сущности (IndexUnit), постраничный просмотр коллекции экземпляров сущности (ScrollUnit), интерактивный элемент модификации атрибутов экземпляра сущности (EntryUnit) и т.д. Навигационные связи описывают переходы между страницами. Связи могут быть контекстно-независимыми (обеспечивают прямой переход на другую страницу) и контекстно-зависимыми (при переходе передаются некоторые параметры, например, это может быть идентификатор сущности, которую нужно отобразить). Имеются примитивы задания и получения значений

глобальных параметров, объединения/композиции (AND и OR) страниц.

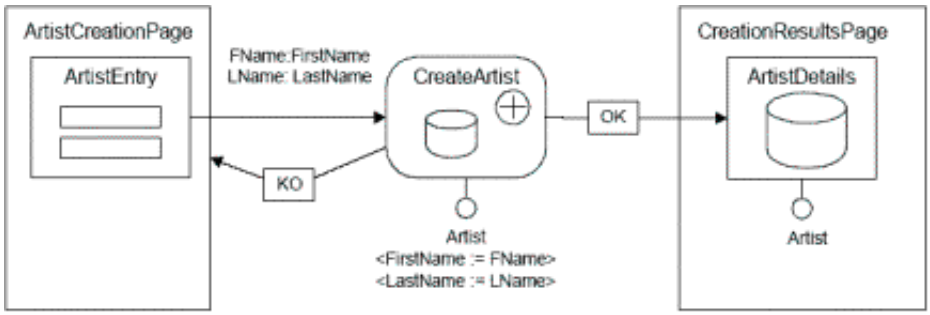

Рисунок 1. Операция создания сущности (указана входная «секция данных» - EntryUnit и выходная «секция данных» - DataUnit).

Формируя **модель операций управления содержимым и контроля доступа** (Content Management and Access Model), разработчик указывает: какие из стандартных (добавление, изменение, удаление, например, ModifyEntity - модификация атрибутов экземпляра сущности) или собственных операций можно применять к экземплярам тех или иных сущностей, кто может выполнять эти операции – какие пользователи или группы пользователей, заданные в модели пользователей и групп. Для каждой операции необходимо указать входную и выходную «секции данных», которые могут относиться как к разным страницам, так и к одной. Например (смотрите рисунок), для операции создания сущности требуется указать входную «секцию данных» (EntryUnit) и выходную «секцию данных» (DataUnit). Дело в том, что обычно проведение операции, например, модификации хранимых данных, связано с отправкой запроса серверу, выполнению соответствующего действия, вывод пользователю результата в виде следующей страницы или формы. Это еще один вид связывания страниц модели гипертекста. Например, исходная страница может быть формой редактирования хранимого объекта (EntryUnit), а результирующая страницей его просмотра (DataUnit). Важно и то, что разработчик может определять собственные операции, задавая на уровне модели только сигнатуру операции (имя функции и ее параметры). Непосредственно реализация операции моделью управления содержимым не рассматривается.

WebML довольно мощное и интересное средство моделирования, поддерживаемое средой визуального проектирования и реализации web-приложений WebRatio. К недостаткам технологии можно отнести относительно низкий уровень моделирования данных (практически реляционное) и собственные язык и формат спецификации моделей. При моделировании данных не хватает черт объектно-ориентированного моделирования таких, как наследование, сложно структурированные атрибуты. Использование стандартизованных языков моделирования, например, UML, позволило бы решить и эту задачу, а с помощью XMI обеспечить интероперабельность с другими средами проектирования. Сомнительным выглядит и вынесение на уровень моделирования управление пользователями и управление доступом к данным. Это обычно составляет часть функциональности самой информационной системы, поскольку на практике часто

обеспечивается саморегистрация пользователей, защита данных осуществляется не только на уровне визуальных форм, но и на уровне экземпляров данных, даже их частей. Модель гипертекста занимает промежуточное положение между логической и физической моделями визуального интерфейса. В ней присутствуют как возможности описания логических элементов интерфейса, так и физические, привязывающие ее к HTML. Было бы лучше разделить эти два вида моделирования, чтобы иметь возможность использовать логические модели визуального интерфейса не только для web, но и для, например, Windows приложений. А в физической модели пользоваться не только стандартными HTML примитивами, но и готовыми компонентами веб-платформы. Можно отметить и слабую поддержку повторного и/или множественного использования моделей и их частей в рамках одного или нескольких проектов реализации ИС. Это отсутствие механизма поддиаграмм, механизма включения(include) составных частей спецификаций.

WWM

WebForm-based Web application modeling Methodology (WWM) [WWM] также представляет нотацию для графического моделирования информационных web-систем (описание структур данных системы, базовый уровень прикладной логики и спецификацию визуального интерфейса системы), однако, в отличии от WebML, применяются стандартизованных языках моделирования, например, UML[UML] и UCM[UCM] (Use Case Maps – http://www.usecasemaps.org), и, кроме того, отличается количеством и видами моделей.

Для описания **структур данных** системы (**моделирование данных** предметной области) в WWM используются диаграммы классов UML. Диаграмма классов UML представляет собой граф, узлами которого являются элементы структуры модели (классы, интерфейсы, пакеты), а дугами – отношения между узлами (ассоциации, наследование, зависимости). Можно выделить следующие основные элементы диаграмм классов UML: *класс (class)* – описание общих свойств группы сходных объектов; *метакласс (metaclass)* – средство для классификации классов: экземплярами метакласса являются классы; *объект (object)* – экземпляр класса; *ассоциация (association) – бинарное* отношение между классами, являющееся способом описания взаимодействия объектов этих классов; *наследование (inheritance) – бинарное* отношение между классами, с помощью которого можно в отдельный класс вынести общие свойства нескольких классов.

После моделирования данных создаются возможные **сценарии использования системы**. Для этого используется диаграммы использования (Use Cases) языка UML. Можно выделить следующие основные элементы диаграмм использования UML. Use case - это сценарий типичного взаимодействия между пользователем и системой. Actor – роль, которую пользователь играет относительно системы. В дополнение к связям между actors и use cases, имеются два других типа связей. Они соответственно представляют uses (использует) и expends (расширяет) зависимости среди use case.

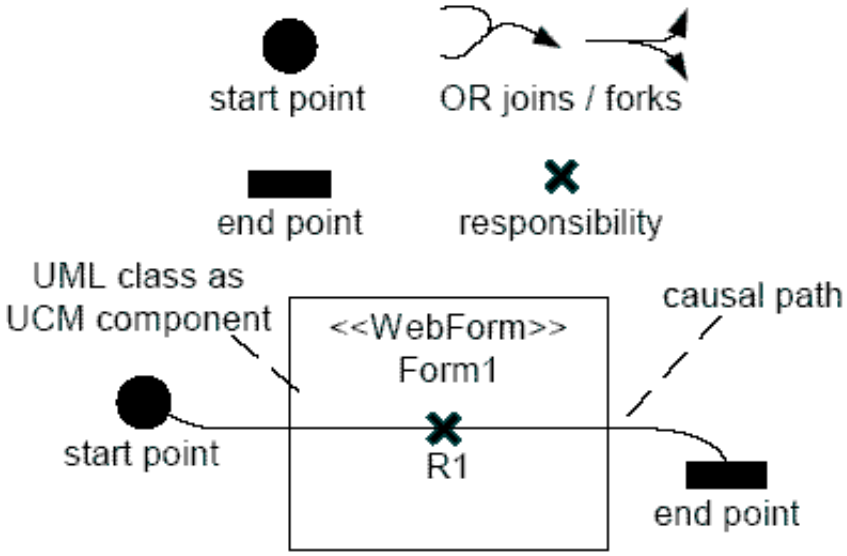

Рисунок 2. Основные понятия UseCaseMaps.

Далее для каждого сценария использования создается **модель Web-форм** (представляется в виде диаграммы Use Case Map (UCM). Не будем подробно описывать, что представляет собой этот стандарт. В кратце (смотрите рисунок), Use Case Map описывает путь (соответствует потоку управления) по графу компонентов (components) системы (system), позволяя в каждом компоненте выполнять определенные действия (responsibilities). Применительно к моделированию информационных web-систем компонентами (components) являются отдельные страницы, а действиями (responsibilities) “секции данных” (например, просмотр информации, изменение, удаление), переходы на другие страницы (компоненты), например, на страницы осуществляющие добавление/изменение информации. Существует возможность создавать свои собственные действия, обеспечивающие как просмотр и модификацию данных, так и реализующие любую другую логику. Фактически, подобные действия описывает “элементы управления” аналогичные элементам управления Delphi и Visual Studio.

К основным достоинствам WWM относится использование распространенных стандартов (UML и UCM), которые поддерживаются многими средами визуального проектирования, например, UCMNav[UCMNav] (http://www.usecasemaps.org/tools/ucmnav/index.shtml) для UCM. Стоит отметить, что хотя использование стандартов является плюсом данной технологии, они налагают и некоторые ограничения на модели. Например, UML не поддерживает наследование свойств. К недостаткам технологии можно отнести слабую поддержку повторного использования моделей, например, невозможность описать структуру страницы один раз и использовать ее в дальнейшем во всех страницах, а также отсутствие механизма поддиаграмм и механизма включения(include) составных частей спецификаций.

XWMF

XWMF [XWMF] основана на Resource Description Framework (RDF) и описывает способы применения RDF/RDFS для описания структур данных, определения видов

допустимых операций манипулирования данными и спецификации визуального интерфейса информационных web-систем. XWMF поддерживает следующие виды моделей: модель структур данных системы (Modeling Schema), модель навигации (Navigational Schema), модель разметки страниц (Layout Schema) и схема управления доступом (Access Control Schema). Каждая из этих моделей представлена в формате RDF в соответствии с заданной RDF-схемой. Существуют программные средства на Tcl и Prolog поддерживающие XWMF (http://nestroy.wi-inf.uni-essen.de/xwmf/#XWMF_Tools_Prolog) и предназначенные для автоматической генерации кода на основе подобных описаний.

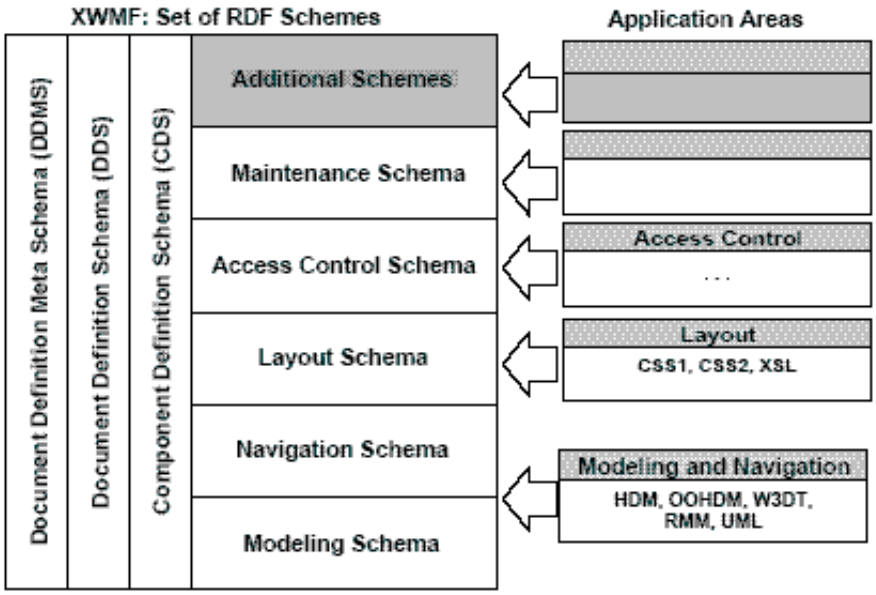

Рисунок 3. Виды моделей XWMF.

Модель данных (Modeling Schema) фактически представляет собой RDF-схему с некоторыми расширениями. RDF Schema [RDF] предоставляет базовую систему типов для моделей RDF. Эта система типов включает некоторые предопределенные термины, такие как Class, subPropertyOf и subClassOf. Для схем, ориентированных на конкретные приложения, используются дополнительные термины. В XWMF также используются некоторые дополнительные термины, приближая выразительность модели данных в XWMF к диаграммам классов UML. Например, ограничения на степень связи, абстрактные классы, тип отношения и т.д. Кроме того, описания структуры данных системы в XWMF можно создавать иерархии наследования свойств. **Модель навигации** (Navigational Schema) определяет связи между страницами, при этом началом и концом каждой связи являются отдельные элементы страниц. Элементы страниц описываются с помощью **модели разметки страниц** (Layout Schema), которая фактически представляет собой описание HTML-разметки в формате RDF (например, туда входят элементы page, header, title, body, value и т.д.). Существует возможность указать где отображать атрибуты элементов модели предметной области на странице. **Модель управления доступом** (Access Control Schema) описывает пользователей, которые могут получать доступ к соответствующей странице.

К недостаткам технологии можно отнести собственные словари спецификации моделей для моделирования данных. Использование стандартизованных языков

моделирования, например, UML для моделирования данных и XMI для обеспечения интероперабельность с другими средами проектирования позволило бы сделать подобные описания более понятными, как и для машин, так и для людей. Аналогично WebML, сомнительным выглядит и вынесение на уровень моделирования управление пользователями и управление доступом к данным. Модель гипертекста излишне детализирована, по сути, она представляет HTML-страницы в формате RDF.

Концептуальное описание

Целью представленного решения является поддержка процесса разработки сложных информационных систем, основанных на технологии ИСИР. Следуя предложенной методологии моделирования, архитектор системы последовательно создает ряд моделей, описывающих разрабатываемую систему, по которым затем с помощью инструментальных средств автоматически получаются фрагменты кода системы.

Методология моделирования

Предлагаемая методология моделирования информационной системы во многом основана на методологиях, предлагаемых в других технологиях автоматизированной разработки web-приложений ([Hera], [RMM]). Архитектор разрабатываемой системы выполняет ряд шагов, на каждом их которых создается спецификация модели, описывающая определенный аспект системы. Согласно многоуровневой архитектуре ИСИР [ISIR] перечень необходимых моделей выглядит так:

- Модель предметной области
- Модель прикладной логики
- Модель взаимодействия с пользователем

Модель предметной области (или *концептуальная модель*) в рамках технологии ИСИР создается на основе объектно-ориентированной модели данных. Источники информации, имеющие такую модель своих данных и поддерживающие стандартные протоколы доступа к ним, называются репозиториями. Подробнее об этом см. [ISIR]

Модель прикладной логики предназначена для описания операций над данными. В отличии от модели предметной области, определяющей семантику данных в репозитории, модель прикладной логики определяет действия, которые могут быть выполнены над этими данными. Важным частным случаем является логическая группировка данных. Следует отметить, что возможные действия над данными не исчерпываются операциями, представленными в модели. Эти операции моделируют набор стандартной функциональности, которая востребована различными клиентскими приложениями. В дальнейшем этот список будет расширен.

Модель взаимодействия с пользователем может быть разделена на две: *модель визуального представления информации* и *модель персонификации*. Первая

описывает, как именно должна быть представлена информация для «среднестатистического» конечного пользователя. Вторая – какие изменения в это представление нужно внести для конкретного пользователя. Как видно из обзора, модель персонификации имеет большое значение в современной методологии построения информационных систем, поэтому она требует отдельного рассмотрения и будет представлена в следующей статье.

Общая методология

В дополнение к существующей модели данных архитектору предоставляются инструменты моделирования прикладного уровня [ISIR] (модель прикладной логики) и уровня представления [ISIR] (модель интерактивного взаимодействия с пользователем). Выбираются из существующих или создаются соответующие языки спецификации моделей. Создаются инструментальные средства, обеспечивающие трансляцию моделей в выполняемый код. Трансляция управляется набором параметров (профилем), определяющим целевую платформу и некоторые аспекты отображения элементов моделей на элементы этой платформы. Трансляция выполняется таким образом, чтобы получаемый в результате этого процесса код был строго изолирован в соответствии с архитектурой многоуровневой системы [ISIR]. На каждом уровне код инкапсулируется в объекты, предоставляющие соответствующую функциональность через согласованные интерфейсы. *Объекты уровня доступа к данным* (в ИСИР называемый связующим уровнем) обеспечивают универсальный объектный интерфейс доступа к гетерогенным источникам данных. *Объекты прикладного уровня*, в частности, реализуют часто используемую логику работы с данными, востребованную различными клиентами, которым также предоставляются интерфейсы. Использующие их *объекты уровня представления* обеспечивают компоновку визуальной среды пользователя, представление в ней данных, а также преобразование и передачу нижележащему уровню информации о действиях пользователя. Физическое взаимодействие между объектами разных уровней может осуществляться различными способами: от локального вызова метода объекта до взаимодействия через, например, web-сервисы в случае, когда код разных уровней выполняется на разных платформах.

Моделирование прикладной логики

Инструмент моделирования прикладных операций предназначен для создания моделей, описывающих прикладную логику работы с данными. Операция – это преобразование множества данных системы в некоторую внутреннюю структуру и набор интерфейсов, обеспечивающих различные представления этой структуры. Каждое понятие инструмента моделирования определяет тип операций, имеющих одинаковые алгоритм преобразования и набор интерфейсов. Как показывает практика, в информационных системах, разрабатываемых на технологии ИСИР, наиболее востребованы следующие типы операций:

- Операция, предоставляющая доступ к подмножеству свойств одиночного экземпляра данных. Она предназначены для группировки логически связанных свойств и используется, например, в объектах уровня представления, отображающих свойства ресурса.

- Операция, извлекающая из множества данных коллекцию однотипных ресурсов в соответствии с заданными ограничениями и предоставляющая интерфейсы для доступа к частям этой коллекции. Например, в информационных системах широко используются выборка части коллекции по первой букве некоторого свойства при организации алфавитных индексов и по диапазону порядковых номеров элементов коллекции при организации, например, страничного просмотра.
- Операция, предоставляющая специальные интерфейсы доступа к данным, составляющим мультииерархические структуры. В качестве примера можно привести интерфейс, выполняющий фильтрацию иерархии в соответствии с заданными критериями и интерфейс, выдающий список элементов иерархии, образующих путь от корня структуры до заданного элемента.
- Операция атрибутного поиска, извлекающая коллекцию элементов данных согласно заданным условиям на их атрибуты и имеющая интерфейсы для разных вариантов предоставления результатов поиска, например, «краткая информация» и «полная информация».
- Операция, выполняющая запрос к множеству данных на некотором формальном языке запросов и предоставляющая доступ к результату запроса через стандартный интерфейс. Эта операция обеспечивает выполнение сложной логики, для которой не реализованы специальные операции.

Модель прикладной логики формируется на основе концептуальной модели. Для каждого класса последней архитектор включает в модель прикладного логики операции тех типов, которые он считает применимыми к этому классу, указывает связи между операциями и необходимыми элементами концептуальной модели (классами, атрибутами), специфицирует дополнительные свойства, например, текст запроса к репозиторию.

Моделирование визуального интерфейса

Инструмент моделирования представления служит для создания моделей визуальных интерфейсов приложений. Требования к свойствам этого инструмента определяются поставленными во введении задачами. Помимо указанных задач, на уровне представления возникает **з**адача автоматизации поддержки навигационного контекста.

Рассмотрим модель взаимодействия пользователя с информационной системой. Взаимодействие осуществляется посредством «информационных страниц». Каждая такая страница состоит из набора информационных элементов, образующих иерархическую структуру: одни элементы могут быть составной частью других элементов. Некоторые элементы являются «основными», т. е. отображают интересующую пользователя информацию и предоставляют ему перечень допустимых в текущем состоянии действий, а часть – вспомогательными: предоставляющие выбор дополнительных действий, выводящих вспомогательную информацию или выполняющих оформительскую функцию. Содержание информационной страницы определяется набором параметров и их значений. Информационные элементы могут использовать

значения определенных параметров для генерации своего содержания. Помимо информационного содержания, пользователю предоставляется также возможность выбора действия, предоставляемая навигационными элементами, являющимися разновидностью информационных. Действие заключается в переходе к следующей информационной странице., для чего навигационный элемент должен определить целевую страницу в терминах параметров и их значений. Сложность любой реализации этой модели заключается в двух моментах. Первый: при определении значений параметров следующей страницы соответствующий элемент определяет только существенные для него параметры, однако этот список может не определять полный перечень параметров, необходимых для корректного отображения элементов новой страницы. Вопросы агрегирования потребностей различных информационных элементов в контекстной информации должны решаться таким образом, чтобы при разработке информационных элементов не было необходимости внедрять в него логику поддержки других элементов. Второй: механизмы передачи контекстной информации довольно существенно различаются на разных платформах. Поэтому необходимо дать разработчику возможность специфицировать эту информацию в платформенно-независимом виде и обеспечить отображение этих спецификаций на технологии целевых платформ.

Теперь рассмотрим понятия инструмента моделирования представления. Их удобно описывать объектной моделью, так что при добавлении новых понятий можно использовать концепцию наследования для уточнения семантики существующих. Мы вводим следующие основные понятия:

· ExternalProperty

· ContextParam

· RenderedObject

· SimpleRO

· NavigationRO

Понятие ExternalProperty используется для добавления в модель вспомогательных элементов, представляющих значения свойств других элементов модели, с целью уменьшения дублирования информации в моделях.

Понятие ContextParam представляет множество элементов модели, описывающие контекстные параметры. В модели могут быть установлены связи, определяющие область действия параметров.

Понятие RenderedObject является базовым для всех элементов моделей, соответствующим визуальным элементам. Мы вводим два понятия, наследующие это свойство.

- SimpleRO – визуальные компоненты простой структуры. Приложения этого понятия моделируют объекты, отображающие статический HTML-текст, или

же производящие его на основе заданных статических XML- и XSL-документов. Отличительной особенностью объектов этого типа является их статичность и, следовательно, возможность кэширования выдаваемой ими информации для оптимизации производительности.

· NavigationRO – объекты, составляющие навигационное пространство приложения. Применения этого понятия описывают объекты, которые могут образовывать основу информационной страницы, представляемой пользователю.

Каждая сущность системной модели типа RenderedObject может быть отображена на класс в среде приложений, для которого поддерживается операция выдачи содержимого пользователю. Можно также реализовывать отображения на специализированные классы.

Реализация

В настоящее время реализуется прототип инструментальной среды, задачей которого является автоматизированная генерация исполняемого кода уровня представления на платформе Microsoft .NET. Прототип позволет по одной модели создавать как приложение на ASP.NET, так и компоненты для MicrosoftSharePointPortalServer 2003. Реализация и частично исследование поддержаны MicrosoftResearch.

Реализация на платформе ASP.NET

Архитектура приложения, получаемого с помощью инструментальной среды, имеет следующий вид. Автоматически генерируется ASP.NET страница (класс, наследуемый от System.Web.UI.Page), в методе Render которого содержится код, обеспечивающий визуализацию страницы. Для доступа к данным вызываются методы прокси-объекта, который реализует операцию модели прикладной логики и генерируется автоматически. Этот объект использует библиотеки поддержки ИСИР для формирования и посылки запросов на сервер приложения.

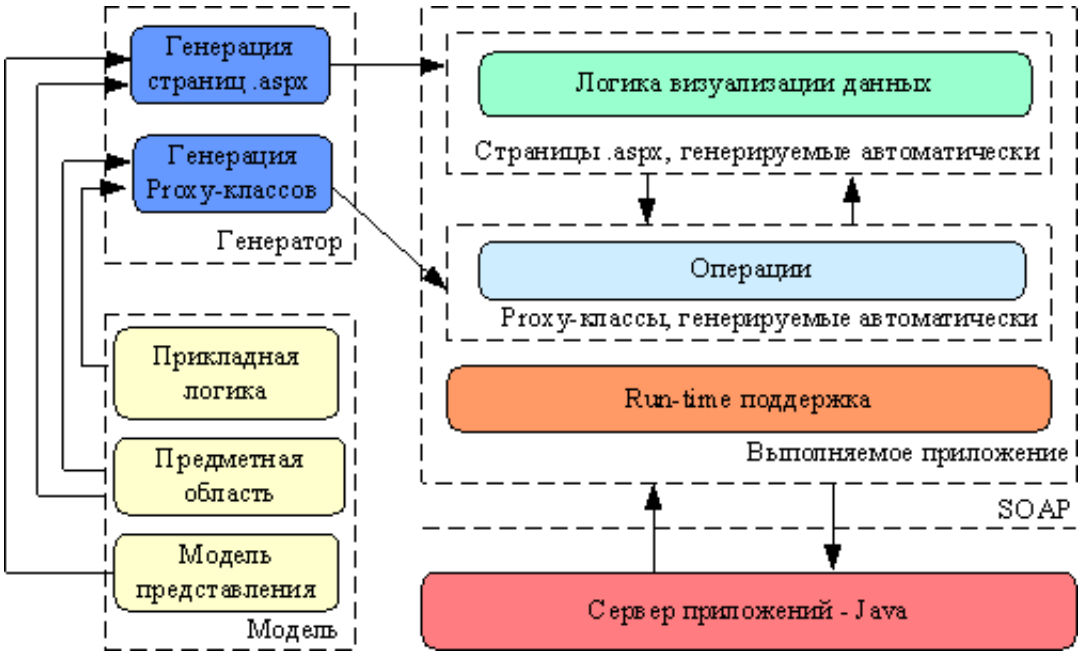

Рисунок 4. Реализация на платформе ASP.NET.

Реализация на платформе SharePointPortalServer 2003

Реализация для SharePointPortalServer отличается лишь тем, что содержимое метода Render переносится в метод RenderWebPart класса, который наследуется от класса Microsoft.SharePoint.WebPart. Этот класс, вместе с прокси-классами доступа к репозиторию, компонуется в библиотеку (assembly).

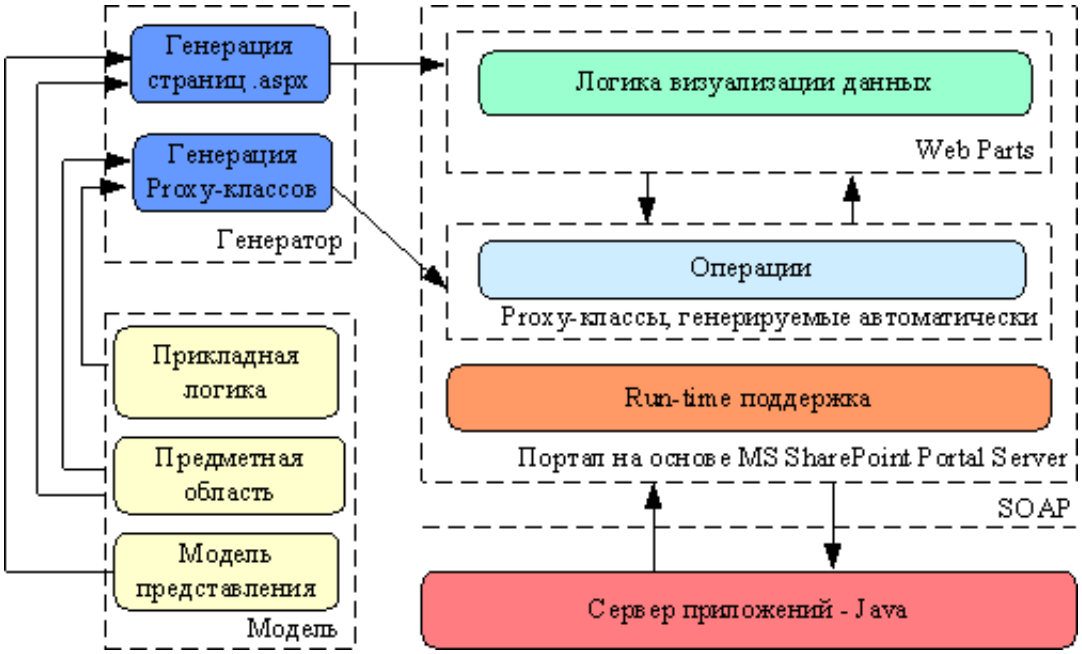

Рисунок 5. Реализация на платформе SharePointPortalServer 2003.

Литература

[ISIR] Бездушный А.А., Бездушный А.Н., Нестеренко А.К., Серебряков В.А., Сысоев Т.М., Архитектура и технологии RDFS-среды разработки цифровых библиотек и Web-порталов. // Электронные библиотеки, 2003, Том 6, Выпуск 4. http://www.elbib.ru/index.phtml?page=elbib/rus/journal/2003/part4/BNSBS

[WebML] Stefano Seri, Piero Fraternali, Aldo Bongio: "Web Modeling Language (WebML): a Modeling Language for Designing Web Sites". WWW9 Conference, Amsterdam, May 2000.

[WWM] Kaewkasi, C. and W. Rivepiboon: “WWM: A Practical Methodology for Web Application Modeling”. In The 26th Annual International Computer Software and Applications Conference. 2002: IEEE Computer Society.

[UML] OMG, UML 2.0 Specifications, http://www.uml.org/

[UCM] Use Case Maps, http://www.usecasemaps.org/index.shtml

[UCMNav] Use Case Maps Navigator,
http://www.usecasemaps.org/tools/ucmnav/index.shtml

[XWMF] Reinhold Klapsing, Gustaf Neumann: “An eXtensible Web Modeling Framework”. Proceedings of the 8th World Wide Web Conference, Poster Session Toronto, May 11-14, 1999.

[RDF] Resource Description Framework (RDF) Model and Syntax Specification. World Wide Web Consortium. http://www.w3.org/RDF

[DC] DC: Dublin Core Metadata Initiative. http://dublincore.org/

[WAE] Jim Conallen “Building Web Applications with UML”, Addison-Wesley, 2002.

[UWE] Nora Koch, Hubert Baumeister, Rolf Hennicker, and Luis Mandel: “Extending UML for Modeling Navigation and Presentation in Web Applications”. In Geri Winters and Jason Winters, editors, In Modeling Web Applications in the UML Workshop, UML2000, York, England, October 2000.

[XPF] Andreas Kraus and Nora Koch: “Generation of Web Applications from UML Models using an XML Publishing Framework”. In 6th World Conference on Integrated Design and Process Technology (IDPT), June 2002.

Об авторах

Бездушный Анатолий Николаевич - кандидат физико-математических наук, с.н.с. Вычислительного Центра имени А. А. Дородницына РАН. Сфера деятельности: системное программирование, параллельные вычисления, сети, базы данных,распределенные системы, информационно-поисковые технологии, цифровые библиотеки, Web-порталы.
Тел. (095)1355471(*4216)
E-mail: bezdushn@ccas.ru

Михеев Петр Николаевич - аспирант вычислительного центра РАН. Сфера деятельности: программирование, базы данных, цифровые библиотеки.
E-mail: miheev@ccas.ru

Меденников Антон Михайлович - м.н.с. Центра научно-технических коммуникаций РАН. Сфера деятельности: базы данных, цифровые библиотеки, управление доступом к данным.
E-mail: meden@ccas.ru