Коррекция обозначаемых через x полученных предварительных значений рейтинговых оценок $R''хор1$ и $R''хор2$ выполнима с помощью линейного преобразования, описываемого функцией (y) одной переменной (x) вида:

$$y=kx+b. \quad (9)$$

Учитывая, что из логических условий оценивания достигаемые максимально возможные значения y и x в случае верхнего диапазона (5) равны соответственно 4,499(9) и 4,749(9), а минимально возможные равны 4,0 и 4,0, составим систему двух линейных уравнений вида

$$\begin{cases} 4,499(9) = k1 \cdot 4,749(9) + b1, \\ 4,0 = k1 \cdot 4,0 + b1. \end{cases} \quad (10)$$

Результаты решения системы уравнений – $k1 \approx 0,665342204$ и $b1 \approx 1,338631184$. На основании этого соотношение (9) для верхнего диапазона рейтингового интервала «хорошо» принимает вид

$$y=0,665342204x+1,338631184. \quad (11)$$

Значения $k2$ и $b2$ для нижнего диапазона рейтингового интервала «хорошо» определяются аналогично.

Убедимся на конкретном примере, что с использованием линейного преобразования (11) реализуется логическое условие получения оценки «хорошо».

Пусть частная оценка практических навыков специалиста $X_П=4,49$ («хорошо»), а теории – $X_Т=5,0$ («отлично»). Средний балл его подготовленности $R'хор=R''хор1=4,745$, что, согласно (1), соответствует рейтинговому интервалу $4,5 \le$«отлично»≤ 5.

Однако по нормативным условиям получения оценки требуется, чтобы интегральная итоговая оценка $Rxop$ специалиста была не выше оценки за практику, в данном примере равной «хорошо».

Подставив полученную промежуточную рейтинговую оценку $R''хор1=4,745$ в (11), получим $Rxop=4,745 \times 0,665342204+1,338631184 \approx 4,496$, то есть итоговая рейтинговая оценка специалиста находится в требуемом верхнем диапазоне рейтингового интервала «хорошо», что соответствует заданным нормативным условиям оценивания.

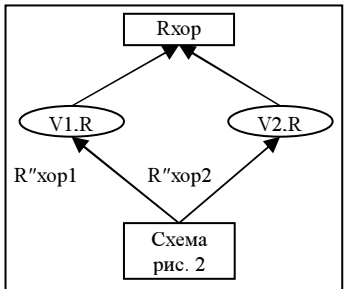


Рис. 3. Получение итоговой рейтинговой оценки

Реализация коррекции выполнима с помощью весовых коэффициентов:

$$V1,R = \begin{cases} (k1\ X3,1 + b1), \text{ если } R''хор2 > 0, \\ 0 \text{ в ином случае;} \end{cases}$$

$$V2,R = \begin{cases} 0, \text{ если } R''хор1 > 0, \\ (k2\ X3,1 + b2) \text{ в ином случае.} \end{cases}$$

Учитывая, что промежуточное значение оценки находится в одном из диапазонов рассматриваемого рейтингового интервала, математическое соотношение, описывающее метод получения итоговой рейтинговой оценки «хорошо» ($Rxop$), будет иметь вид

$$Rxop = V1,R \cdot R''хор1 + V2,R \cdot R''хор2. \quad (12)$$

Математическое соотношение (12) может быть представлено фрагментом структурной схемы, отображенной на рисунке 3.

Аналогично могут быть построены и реализованы соотношения для вычисления рейтинговых оценок R(*отл*), R(*уд*) и R(*неуд*).

Литература

1. Долгов А.И., Мартыненко А.Ф., Преснухин В.В. Научные основы построения коэффициентных методик оценки объектов // Системы управления и информационные технологии. 2008. № 4 (34). С. 61–66.

2. Государственная система обеспечения единства измерений. Шкалы измерений. М.: Стандартинформ, 2008.

3. Долгов А.И., Мартыненко А.Ф., Преснухин В.В. Коэффициентная методика с изменяемыми весовыми коэффициентами // Искусственный интеллект. 2008. Т. 4.

УДК 007:519.816

РАСПРЕДЕЛЕННАЯ ИНФОРМАЦИОННАЯ СИСТЕМА ПОИСКА ЛИТЕРАТУРЫ

М.М. Маран, к.т.н.; Лвин Маунг Со

(Национальный исследовательский университет «Московский энергетический институт», mm@appmat.ru, lwinmgsoe@gmail.com)

Рассматриваются возможности построения и реализация распределенной информационной системы поиска литературы. Реализация выполнена согласно концепции сервис-ориентированной архитектуры на платформах Windows Communication Foundation, Microsoft SQL Server 2008 и Visual Studio 2010.

Ключевые слова: *информационная система, поиск литературы, SOA, WCF, ADO NET MVC.*

Предлагаемая информационная система обеспечивает удобный доступ через веб-интерфейс к каталогу полнотекстовых документов и мультимедийных ресурсов, а также полнотекстовый поиск и

поиск по их атрибутам [1]. Кроме того, администраторы библиотеки имеют возможность редактирования ресурсов библиотеки.

Реализация выполнена в среде Microsoft Visual Studio 2010 с использованием технологий Asp.net MVC 3, ADO.NET Entity Framework, Windows Communication Foundation (WCF). Рассмотрим эти технологии подробнее, отметив их преимущества.

Технологии реализации

Платформа ASP.NET MVC представляет собой удобную среду создания web-приложений, интегрированную с существующими функциями ASP.NET и дополненную средствами MVC (model–view–controller) (*модель–представление–контроллер*):

– *модель (model)* предоставляет данные для *представления* в ответ на запросы *контроллера*, содержит бизнес-логику приложения;

– *представление (view)* отвечает за пользовательский интерфейс, отображает данные, полученные от *модели;*

– *контроллер (controller)* обрабатывает команды пользователя, определяет *модель* для работы и связывает ее с *представлением*.

Бизнес-логика, расположенная в *модели*, включает все правила и алгоритмы, связанные с предметной областью решаемой задачи. Проще говоря, это ядро создаваемого приложения. Рассматриваемая архитектура подразумевает, что изменения в любом из компонентов оказывают минимальные воздействия на остальные части. Несколько упрощая, работу MVC-приложения можно описать следующим образом:

– команда (уведомление о нажатии кнопки, запрос адреса сайта и т.д.) передается *контроллеру*;

– *контроллер*, исходя из полученных данных, определяет и вызывает *модель*;

– *модель* на основе заложенной в ней бизнес-логики формирует набор данных;

– *контроллер* выбирает *представление* и связывает его с данными (*моделью*);

– *представление* отображает данные пользователю.

Контроллер играет роль связующего звена между *моделью* и *представлением*. При этом он стремится как можно меньше знать о подробностях их реализаций. Его задача – определить *модель* для обработки полученной команды и *представление*, которому будут отправляться итоговые данные.

Представление зависит от *модели*, так как использует получаемые от нее данные. *Модель* не зависит ни от *представления,* ни от *контроллера.* Это позволяет вести разработку *модели* независимо, а также создавать для нее несколько *представлений*.

Платформа ASP.NET MVC имеет следующие преимущества:

– облегчает управление сложными структурами путем разделения приложения на *модель*, *представление* и *контроллер*;

– не использует состояние просмотра и серверные формы, что делает эту платформу идеальной для разработчиков, которым необходим полный контроль над поведением приложения;

– использует схему основного *контроллера*, при которой запросы web-приложения обрабатываются через один *контроллер*, что позволяет создавать приложения, поддерживающие расширенную инфраструктуру маршрутизации, а также обеспечивать расширенную поддержку разработки на основе тестирования;

– хорошо подходит для web-приложений, поддерживаемых крупными коллективами разработчиков, которым необходим высокий уровень контроля над поведением приложения.

Платформа ADO.NET Entity Framework (EF) – объектно-ориентированная технология доступа к данным, являющаяся объектно-реляционным отображением (object-relational mapping, ORM) решений для .NET Framework от Microsoft. Она предоставляет возможность взаимодействовать с объектами как посредством LINQ в виде LINQ to Entities, так и с использованием Entity SQL. Для облегчения построения web-решений используются и ADO.NET Data Services (Astoria), и связка из Windows Communication Foundation и Windows Presentation Foundation, позволяющая строить многоуровневые приложения, реализуя один из шаблонов проектирования – MVC, MVP или MVVM [2]. Платформа ADO.NET Entity Framework дает возможность разработчикам создавать приложения для доступа к данным, работающие с концептуальной моделью приложения, а не напрямую с реляционной схемой хранения. В результате можно добиться уменьшения объема кода и снижения затрат на сопровождение приложений, ориентированных на обработку данных.

Таким образом, основной задачей ORM является установка соответствия между объектами, используемыми в приложении, и таблицами, хранящимися в реляционных БД.

Создание SQL-запросов. При запросе приложением объектов ORM-библиотека самостоятельно создает SQL-код запросов и передает его в СУБД. При необходимости разработчик может вмешаться в данный процесс с целью тонкой оптимизации производительности. При этом возможно возникновение проблемы несоответствия типов. В качестве решения ORM при записи значения может использовать свойство объекта.

Абстракция используемой БД. В процессе работы с ORM-библиотекой приложение оперирует привычными для него объектами. Но при этом для хранения информации могут использоваться раз-

личные реляционные СУБД, в частности, SQL Server, SQL Server Express, SQL Server Compact, mySQL и т.д. Это вносит дополнительный уровень гибкости в архитектуру приложения.

Приложения Entity Framework имеют следующие преимущества:

– могут работать с концептуальной моделью в терминах предметной области, в том числе с наследуемыми типами, сложными элементами и связями;

– освобождаются от жестких зависимостей от конкретного ядра СУБД или схемы хранения;

– сопоставления между концептуальной моделью и схемой, специфичной для конкретного хранилища, могут меняться без изменения кода приложения;

– разработчики имеют возможность работать с согласованной моделью объектов приложения, которая может быть сопоставлена с разными схемами хранения, возможно, реализованными в различных СУБД;

– несколько концептуальных моделей могут быть сопоставлены с единой схемой хранения;

– поддержка запросов LINQ обеспечивает проверку синтаксиса во время компиляции для запросов к концептуальной модели.

Технология Windows Communication Foundation (WCF) [3] предоставляет единую инфраструктуру разработки, повышающую производительность и снижающую затраты на создание безопасных, надежных и транзакционных web-служб нового поколения.

WCF – это технология для построения сервис-ориентированной архитектуры приложений (SOA – service-oriented architecture), что позволяет абстрагироваться от конкретной технологии, на которой этот сервис реализован, и пользоваться им из других приложений, написанных на любых других платформе, языке, технологии, – главное, чтобы реализация клиента отвечала определенным правилам. Кроме того, логика самого сервиса и его реализация полностью отделены от коммуникационной составляющей и существует возможность декларативно изменять способ взаимодействия с сервисом путем изменения конфигурационного файла. Можно изменить протокол взаимодействия, адрес, настроить максимальное количество подключений, ограничить размер пакетов, тайм-аут подключения к сервису и выполнения операции и многое другое.

В основе технологии WCF лежит принцип связи с помощью обмена сообщениями, и любые объекты, моделируемые в виде сообщений (например, HTTP-запрос или сообщение очереди сообщений, MSMQ), можно представить единым образом в модели программирования. Это обеспечивает универсальный интерфейс API для разных транспортных механизмов. Сообщения можно отправлять через интрасети или через Интернет общими транспортами, такими как HTTP и TCP. С помощью встроенных точек расширения WCF добавляются дополнительные транспортные механизмы. Служба WCF поддерживает несколько шаблонов обмена сообщениями, включая запрос–ответ, одностороннюю и дуплексную связь. Разные транспорты поддерживают разные шаблоны обмена сообщениями и таким образом влияют на типы поддерживаемых взаимодействий. Интерфейсы API и среда выполнения WCF также помогают отправлять сообщения безопасно и быстро.

При создании приложений для .NET разработчики пользуются средой Visual Studio. В WCF и в Visual Studio есть инструменты для реализации служб. В WCF встроена модель размещения, позволяющая размещать службы в IIS или в Managed Services на платформе Windows. WCF поддерживает развитую модель многопоточности и ограничения пропускной способности (throttling), которая позволяет управлять созданием экземпляров с минимальными усилиями. Вне зависимости от того, обрабатываются ли поступающие запросы одновременно в одном или в нескольких потоках, модель программирования остается одинаковой, так что разработчик может не вдаваться в детали (которые, однако, остаются ему доступными). WCF поддерживает различные способы обмена сообщениями, например: запрос–ответ, односторонний и дуплексный поток. Поддерживаются также пиринговые сети, в которых клиенты могут обнаруживать друг друга и обмениваться данными при отсутствии централизованного механизма управления. Словом, технология WCF важна потому, что современные приложения немыслимы без служб, а именно это и составляет назначение и смысл WCF.

Структура программы eLibrary

eLibrary – распределенная информационная система, состоящая из взаимодействующих в локальной сети подсистем (рис. 1). Каждая подсистема функционирует как собственная информационная система, у подсистем имеются своя СУБД и свои приложения. Подсистемами в рассматривае-

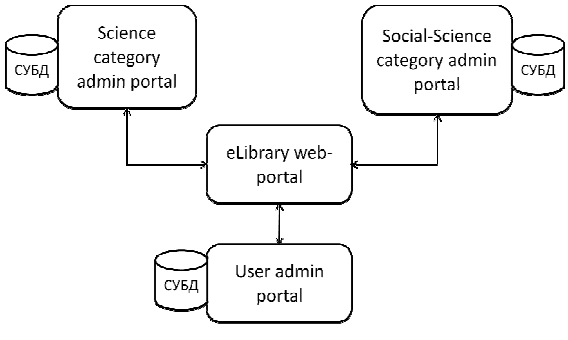


Рис. 1. Архитектура системы eLibrary

мом случае являются Science category admin portal, Social-Science category admin portal, User admin portal, eLibrary web-portal.

Рассмотрим кратко структуру и назначение подсистем.

Подсистема Science category admin portal (см. рис. 2) создана для выполнения действия с документами по категории «Научный» в библиотеке. В ней функционирует СУБД Microsoft SQL Server 2008. В подсистеме имеется одно web-предложение, с помощью которого администраторы могут работать с данными. Для взаимодействия с другими подсистемами в ней также размещен реализованный на WCF web-сервис. Этот сервис поддерживает способ обмена данными и выполнения операций в формате сообщений SOAP.

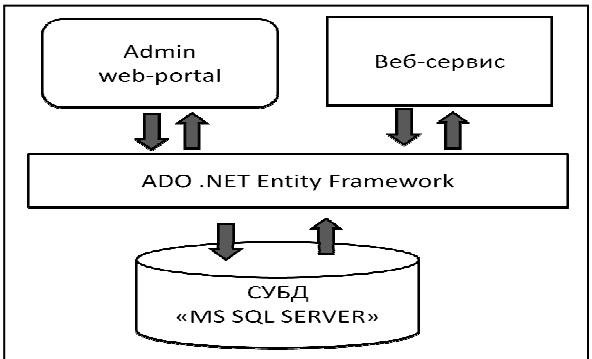


Рис. 2. Архитектура подсистемы Science category admin portal

Как показано на рисунке 2, для взаимодействия с данными СУБД Microsoft SQL Server 2008 использована платформа ADO.NET Entity Framework. Для ее функционирования разработано представление реляционных таблиц в виде классов. Web-приложение создано на основе технологии ASP.NET MVC Framework на языке C# в среде Microsoft Visual Studio 2010.

Подсистема Social-Science category admin portal. Архитектура и функции данной подсистемы такие же, как у подсистемы Science category admin portal. В БД этой подсистемы хранятся документы по общественным наукам.

Подсистема User admin portal предназначена для управления данными об администраторах и пользователях библиотеки. Ее архитектура аналогична описанной выше, различаются структура данных и их представлений в виде классов.

Подсистема eLibrary web-portal (рис. 3) является клиентом для пользователей библиотеки. В ней только одно web-приложение, позволяющее пользователям найти все документы в библиотеке. В этой подсистеме нет собственной БД, нужные данные можно получить из других подсистем с помощью их web-сервисов. Для этого в системе создан специальный слой, который управляет всеми работами с данными.

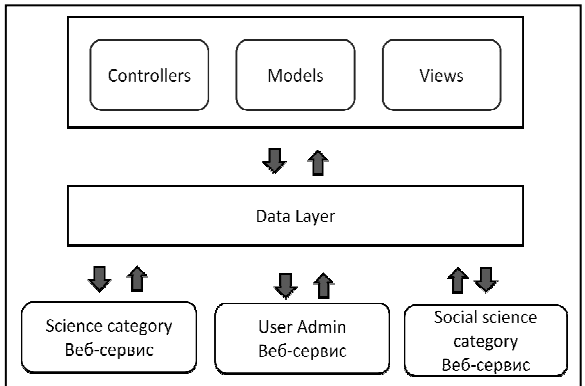


Рис. 3. Архитектура подсистемы eLibrary web-portal

При поступлении запроса пользователя Data Layer посылает запросы в web-сервисы, а при получении ответов из web-сервисов передает их в Controller, затем пользователю (рис. 4). С помощью этого сайта пользователь может найти материалы по названию, по издателю, по авторам и по ключевым словам. Кроме этого, полученные результаты могут фильтроваться, например, по книгам или по статьям. Для просмотра документов пользователю необязательно авторизоваться в системе, а для получения материалов авторизация обязательна.

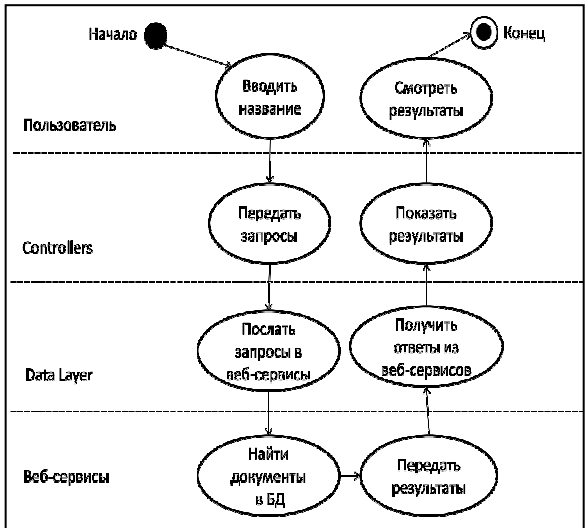


Рис. 4. Диаграмма поиска документов по названию

Предложенная информационная система реализована на широко используемых платформах и дает возможность пользователям вести поиск литературы по различным критериям.

Литература

1. Таненбаум Э., Ван Стеен М. Распределенные системы. Принципы и парадигмы. СПб: Питер, 2003. 845 с.

2. Josuttis N.M. SOA in Practice: The Art of Distributed System Design (Theory in Practice). O'Reilly Media. 2007. 759 p.

3. WCF 4 для профессионалов / Сибарро П. [и др.]. М.: Диалектика, 2011. 464 с.