электронный научно-технический журнал
ИНЖЕНЕРНЫЙ ВЕСТНИК
Издатель ФГБОУ ВПО "МГТУ им. Н.Э. Баумана". Эл No. ФС77-51036. ISSN 2307-0595

Визуальная среда формирования пользовательского интерфейса в системе автоматизированного проектирования информационных систем

77-48211/637897
07, июль 2013
УДК: 004.4'22

авторы: Виноградова М.В., Виноградов В.И.

МГТУ им. Н.Э. Баумана, Москва, Российская Федерация

vinogradova.m@bmstu.ru

vinogradovs.fam@bmstu.ru

Введение

На данный момент существует много инструментов, автоматизирующих процесс разработки информационных систем. Они позволяют автоматически генерировать схемы баз данных [1, 2, 3] и формировать пользовательский интерфейс с помощью визуальных конструкторов [4, 5], а также избавляют программиста от написания рутинных операций за счет применения каркасов [6, 7]. Однако, большинство из этих инструментов автоматизируют только часть процесса разработки системы и требуют написания фрагментов программного кода вручную [8, 9]. Поэтому было решено разработать программный продукт, благодаря которому разработчики получат возможность создавать пользовательские интерфейсы к базе данных без программирования и который будет являться компонентом системы автоматизированного проектирования.

Данная статья посвящена описанию визуальной среды формирования пользовательского интерфейса, которая позволяет разрабатывать графический интерфейс пользователя без написания программного кода, а также определять типовые операции с данными, хранящимися в базе данных. Визуальная среда является компонентом системы автоматизированного проектирования информационных систем, разрабатываемой на кафедре «Системы обработки информации и управления» МГТУ им. Н.Э. Баумана в рамках НИР «Электронный университет» [10]. Место разрабатываемого компонента в системе проектирования представлено на рисунке 1.

Рассмотрим основные компоненты системы автоматизированного проектирования и принципы их использования.

1 Система автоматизированного проектирования

В процессе автоматизированного создания информационной системы [11] участвуют две группы специалистов: конечный пользователь и предметник. Пользователь — это специалист, который будет работать с готовой информационной системой. Он определяет функциональные требования. Предметник — это специалист, обладающий сведениями о понятиях предметной области, их реквизитах и связях.

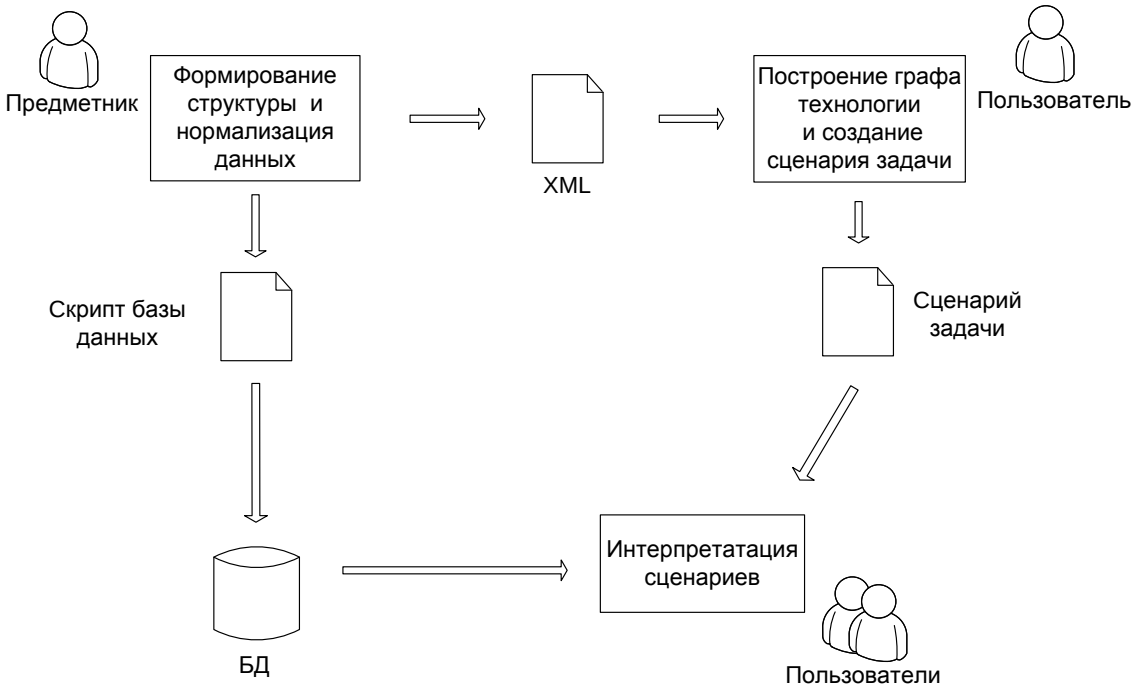

Рисунок 1. Место разрабатываемого компонента в системе

Процесс автоматизированной разработки информационной системы состоит из двух этапов:

1) Предметник задает исходные данные для построения структуры базы данных с помощью визуального конструктора. Он определяет множество понятий, их ключи, реквизиты, связи между понятиями, роли и псевдонимы для идентификации пути до конкретного понятия. Созданная структура записывается в файл в XML — формате [12]. На основе структуры, описанной предметником, определяются значения вычисляемых реквизитов, строится и нормализуется структура базы данных, автоматически генерируется сценарий создания схемы БД на языке SQL.

2) Пользователь задает логику и интерфейс прикладной задачи с помощью визуальной среды формирования пользовательского интерфейса. Исходными данными является описание базы данных, получаемое из файла в формате XML. Используя визуальную среду, пользователь указывает понятия, которые будут использоваться в задаче, тип задачи: просмотр, удаление, изменение или добавления новых записей, и определяет внешний вид интерфейса. На основе этих данных формируется граф технологии, который описывает алгоритм выполнения задачи в терминах модели вычислений, управляемых

потоками данных. Граф технологии записывается как сценарий задачи будущей ИС в виде текстового файла специального формата.

В результате автоматизированной разработки будет получен скрипт создания базы данных на языке SQL и набор сценариев задач. Скрипт на языке SQL используется для создания базы данных, работающей под управлением стандартной СУБД [13]. Для выполнения сценариев необходима программа — интерпретатор. Интерпретатор обрабатывает файлы сценариев задач, интерпретирует их и выполняет записанные в них команды. В процессе выполнения сценария происходит взаимодействие с пользователем, обращение к базе данных и реализация алгоритма задачи. Итоговый и промежуточные результаты выполнения сценария задачи отображаются в виде экранных форм.

Особенностью данной системы автоматизированного проектирования является использование модели вычислений, управляемых потоками данных [14]. Эта модель предполагает, что алгоритм программы определен как преобразование множества входных данных в выходные данные (результат) с помощью операторов, которые реализуют алгоритмические зависимости между выходным реквизитом и множеством входных реквизитов. Для описания одинаковых алгоритмических зависимостей будут использоваться одинаковые операторы. Конкретная реализация оператора не имеет значения на этапе проектирования задачи, поскольку при переходе от описания задачи к программным компонентам оператор может быть заменен любой процедурой, выполняющей алгоритмическую зависимость оператора. Для того чтобы оператор точно соответствовал алгоритмической зависимости, результатом его выполнения должен быть только один реквизит, и на вход оператора должны поступать только те реквизиты, которые необходимы для его выполнения. Последовательность выполнения операторов определяется готовностью их операндов.

Для описания алгоритма программы, управляемого потоками данных, используем модель представления алгоритма графом технологии. Рассмотрим принципы его построения.

2 Описание алгоритма задачи графом технологии

Граф технологии позволяет описать действия над данными и последовательность их выполнения. Граф технологии – это двудольный ориентированный граф. Он задается парой:

$$G^T = \langle W\, q \rangle$$

, где W — множество вершин, q — множество дуг графа.

Вершинами графа являются функции и данные: $W = V \cup F^T$, где $V = \{v_i\}$ — множество данных (переменных), $F^T = \{f_i^T\}$ — множество функций (операторов). Дуги определяют использование данных функциями. Дуги соединяют вершины, соответствующие данным, с вершинами, соответствующими функциям. Будем предполагать, что на вход функции поступает множество данных, а с выхода снимается

один и только один элемент данных. Граф технологии позволяет описывать зависимости между данными вида: $v_i = f^T(\{v_j\})$.

Пример графа технологии, который задает зависимости v4=f1(v1,v2) и v5=f2(v4,v3), приведен на рисунке 2, где v1–v5 — вершины данных, f1–f2 — вершины функций, дуги — использование данных в функциях.

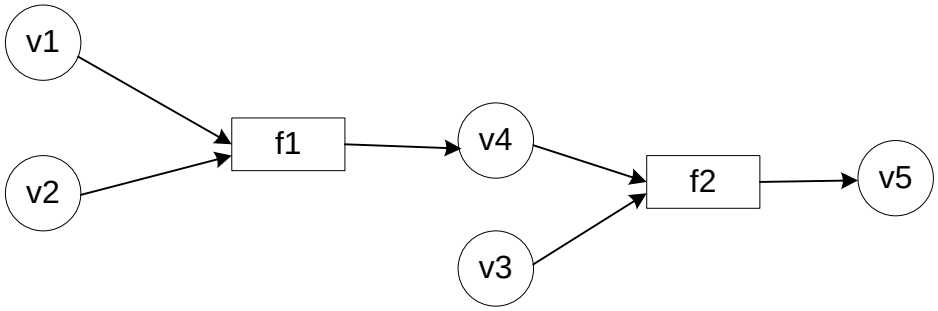

Рисунок 2. Пример графа технологии

Каждая вершина данных связана с конкретной переменной и однозначно определяется именем этой переменной. Вершина функции определяется выполняемой алгоритмической зависимостью, набором входных вершин и выходной вершиной.

На основе графа технологии, определенного для данных некоторой задачи, может быть построен алгоритм решения этой задачи. Перед определением алгоритма определим понятия смежность, достижимость и путь. Перечисленные понятия определяются на графе технологии только для вершин данных.

Вершина данных v4 является смежной для вершин v1 и v2, если на графе технологии присутствует зависимость, в которой операндами являются v1 и v2, а результатом – v4, например, зависимость v4=f1(v1,v2).

Вершина данных v5 является достижимой из вершин данных v1, v2 и v3, если существует транзитивная зависимость между вершинами v1, v2 и v3 и вершиной v5. На графе технологии достижимость обозначается цепочкой смежности, в которой в качестве операндов последующих операторов используются результаты операторов, выполненных ранее. Например, зависимости v5=f2(v3,v4) и v4=f1(v1,v2) реализуют достижимость v5 из v1, v2 и v3.

Путь из вершин данных v1, v2 до вершины v5 определяется как последовательность операторов, выполненных для реализации достижимости, и промежуточных вершин данных, полученных в результате выполнения операторов.

Для построения алгоритма необходимо выделить множество вершин $V^{in}=\{v_i^{in}\}$, соответствующих входным данным, вершину v^{out}, соответствующую результату задачи, и найти путь от вершин V^{in} до вершины v^{out}. Путь проходит через множество промежуточных вершин данных $V^{io}=\{v_j^{io}\}$ и вершин операторов $F^{io}=\{f_k^{io}\}$.

Последовательность достижения вершин операторов F^{io} определяет последовательность их выполнения при решении задачи. Алгоритму соответствует связный подграф графа технологии, реализующий путь от множества входных вершин до выходной вершины. Алгоритм решения задачи задается в виде:

$$A^T = \{G^T, V^{in}, v^{out}, q^A\}$$

, где q^A — множество дуг графа технологии, лежащих на пути от V^{in} до v^{out} .

Исходными данными для построения графа технологии являются: набор типовых функций $F^T = \{f_i^T\}$ и множество реквизитов $V = \{v_i\}$, заданных предметником.

Граф технологии строится в зависимости от типа задачи: просмотр данных, добавление понятия, удаление понятия, изменение реквизитов понятия. Для программной реализации задачи необходимо преобразовать ее граф технологии в текстовую спецификацию, которая будет анализироваться и выполняться программой-интерпретатором. Рассмотрим теоретические принципы формирования спецификации сценария задачи.

3 Преобразование графа технологии задачи в спецификацию

Спецификация — это аналог исходного кода программы, который формируется на основе графа технологии задачи и состоит из множества строк-правил, хранящихся в текстовом файле. Для того, чтобы спецификация задачи корректно отображала алгоритм задачи, заданный на графе технологии, используется аспектная грамматика G^A. Грамматика задается четверкой:

$$G^A = \langle V^A, N^A, S^A, p^A \rangle$$

, где $V^A = H \cup D^C$ — множество терминальных символов, соответствующих множеству операторов H и множеству констант D^C ,

$N^A = V \cup A$ — множество нетерминальных символов, соответствующих множествам данных V и атрибутов A,

S^A — аксиома грамматики, соответствует результату выполнения одной задачи,

p^A — набор правил вывода грамматики:

$S^A \to \xi_i h_i \mid \xi_j h_j$

$N_i^A \to \xi_i h_i \mid \xi_j h_j$

где $h_i, h_j \in H$ — символы операторов, $\xi_i, \xi_j \in \{V \cup A \cup R \cup D^C\}$ — цепочка символов.

Терминальным символам грамматики соответствуют множество операторов и множество константных данных. Нетерминальным символам соответствуют переменные, которые вычисляются в ходе выполнения алгоритма. Наличие нескольких правил вывода

для нетерминального символа означают альтернативные способы вычисления переменной.

Используя описанные правила можно записать граф технологии в текстовом формате, каждая строка которого имеет вид:

<переменная>=<символ>.<символ>.<оператор>

,где <символ> — переменная или константа.

Значение константы указывается в кавычках. Синтаксический анализатор построчно считывает правила из файла, извлекает имя переменной, разбирает правило на лексемы. Последняя из лексем является названием оператора. Прочие лексемы разделяются на имена переменных и значения констант. Разобранное правило передается программе-интерпретатору для семантического разбора и исполнения.

Базовый набор типовых функций для построения графа технологии состоит из функций обращения к базе данных, функций отображения данных, функций работы с переменными и множеством записей данных, функций работы с элементами пользовательского интерфейса. В качестве примера основных функций приведем следующие:

- Ф_Уник — получить уникальные значения;
- Ф_Выбор — выбор значения пользователем;
- Ф_Список — собрать список из переменных;
- Ф_Отобр — отобразить шаблон;
- Ф_Обн — обновление переменной;
- Ф_Сигнал — активный оператор — управляющее действие пользователя.

Для каждой функции определена ее сигнатура, задающая перечень, последовательность, тип и семантику ее операндов и результата.

4 Описание визуальной среды формирования пользовательского интерфейса

Для реализации компонента системы автоматизированного проектирования, который строит граф технологии задачи пользователя и соответствующую ему спецификацию, была разработана визуальная среда. Данная среда позволяет сформировать алгоритм и графический интерфейс пользователя для типовых задач работы с базой данных без написания программного кода.

Для каждой типовой задачи []: ввод, изменение, удаление и просмотр данных,— определен шаблон графа технологии. Граф технологии конкретной задачи получается после подстановки в шаблон ссылок на элементы базы данных и поля ввода. Пример шаблона графа технологии для задачи добавления данных представлен на рисунке 3.

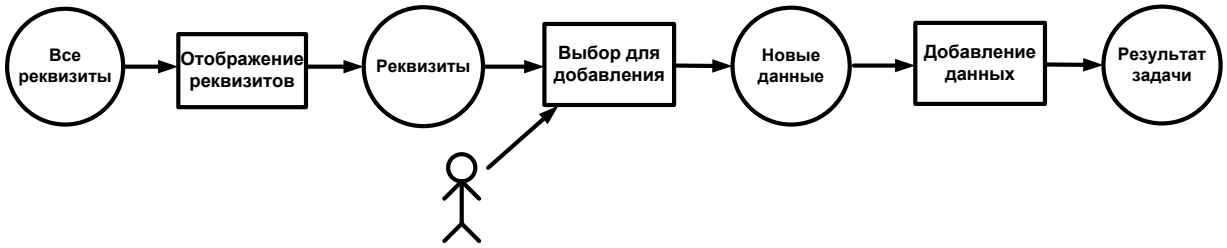

Рисунок 3. Граф технологии задачи добавления данных

Графу технологии задачи добавления данных соответствует следующий фрагмент спецификации:

добСписДанные = группа_нов_id."1".Ф_Список

добСписТабл = "Section"."1".Ф_Список

аксиома_задачи_добавления = rec_id."Student".добСписДанные.добСписТабл.ф_добавитьбд

Спid = резДобавления."Student".Ф_ВсеИзБД

В результате визуального проектирования формируется текст спецификации задачи. Спецификация формируется на основе типа задачи, структуры базы данных и последовательности ввода данных. Спецификация задачи состоит из правил описания интерфейса пользователя, правил перехода между переменными, правил, задающих связь между переменными и элементами интерфейса.

Пример фрагмента спецификации для определения свойств экранной формы:

<свойство i> = “Название свойства”. <значение> . Ф_ОпрСвойство

<список свойств1>=<свойство1>.<свойство2>.<количество>. Ф_Список

<форма> = <список свойств>. Ф_СоздФорму

<компонент i>=<тип>.Ф_СоздатьКомпонент

<компонент i1>= <компонент i>.<список свойств2>. Ф_ОбнКомпонент

Пример фрагмента спецификации для установления связи переменных с элементами ввода:

<переменная1>= "Click". <компонент i>.Ф_Сигнал

<переменная2> = Fl.<компонент i> .Ф_Выбор

<переменная2>=<компонент>.Ф_Ввод

Визуальная среды состоит из четырех модулей:

Модуль чтения XML-данных осуществляет анализ XML файла, содержащего описание структуры БД.

Модуль определения атрибутов и их свойств позволяет пользователю выбрать тип задачи: просмотр, удаление, изменение, добавления данных, а также главное понятие — имя таблицы, данные которой будут использоваться в задаче. На основании множества таблиц, созданного модулем чтения XML-данных, с учетом выбора пользователем основной таблицы, формируется древовидная структура, корнем которой является

выбранная пользователем таблица, узлами — таблицы, связанные с ней с помощью внешних ключей, листьями — атрибуты таблиц.

Модуль определения визуальных параметров формы и компонентов позволяет настроить основные характеристики формы задачи, такие как ширина, высота, положение, цвет, заголовок и определить свойства компонентов.

Модуль формирования файла спецификации отвечает за создание и сохранение в файле текста спецификации.

Последовательность действий при работе с визуальной средой представлена на рисунке 4. На вход поступает информация, определенная предметником и сохраненная в формате XML. Пользователь выбирает тип задачи: добавление, удаление, изменение или просмотр записей. В зависимости от типа задачи пользователь определяет необходимые атрибуты: выбирает основное понятие, задает количество, порядок и вид фильтров для выборки информации из базы данных, перечень отображаемых реквизитов и порядок их группировки. На основе заданных данных автоматически создается шаблон пользовательского интерфейса задачи и открывается в режиме визуального конструктора. Пользователь визуально определяет параметры интерфейса, такие как цвет, размер, заголовок и другие характеристики экранной формы, расположение и свойства ее компонентов. После настройки экранной формы описание графа технологии задачи записывается в виде спецификации в текстовый файл.

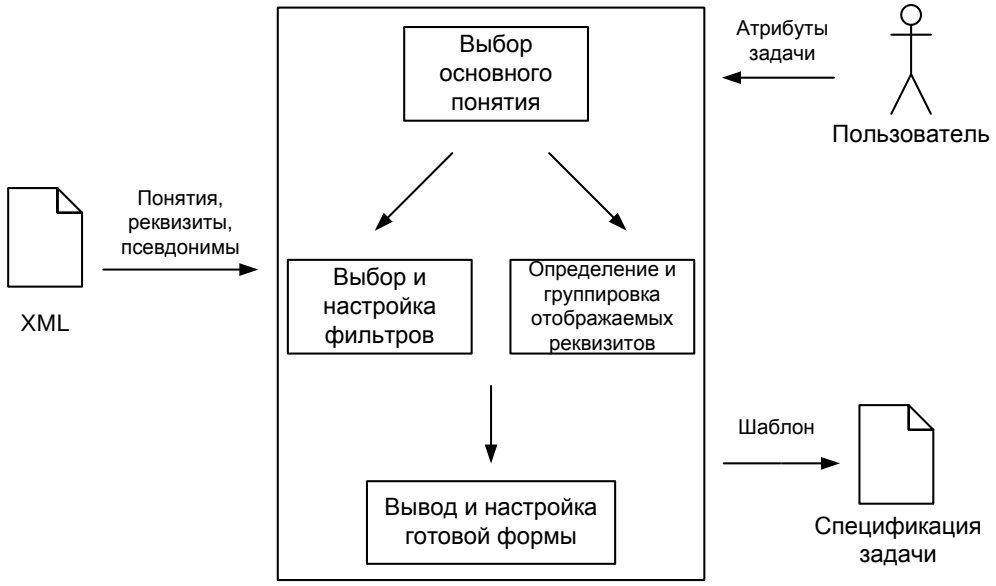

Рисунок 4. Последовательность действий при работе с визуальной средой

Созданная система построения сценариев задач интерфейса базы данных обладает дружественным интерфейсом. Пример экранной формы визуальной среды представлен на рисунке 5.

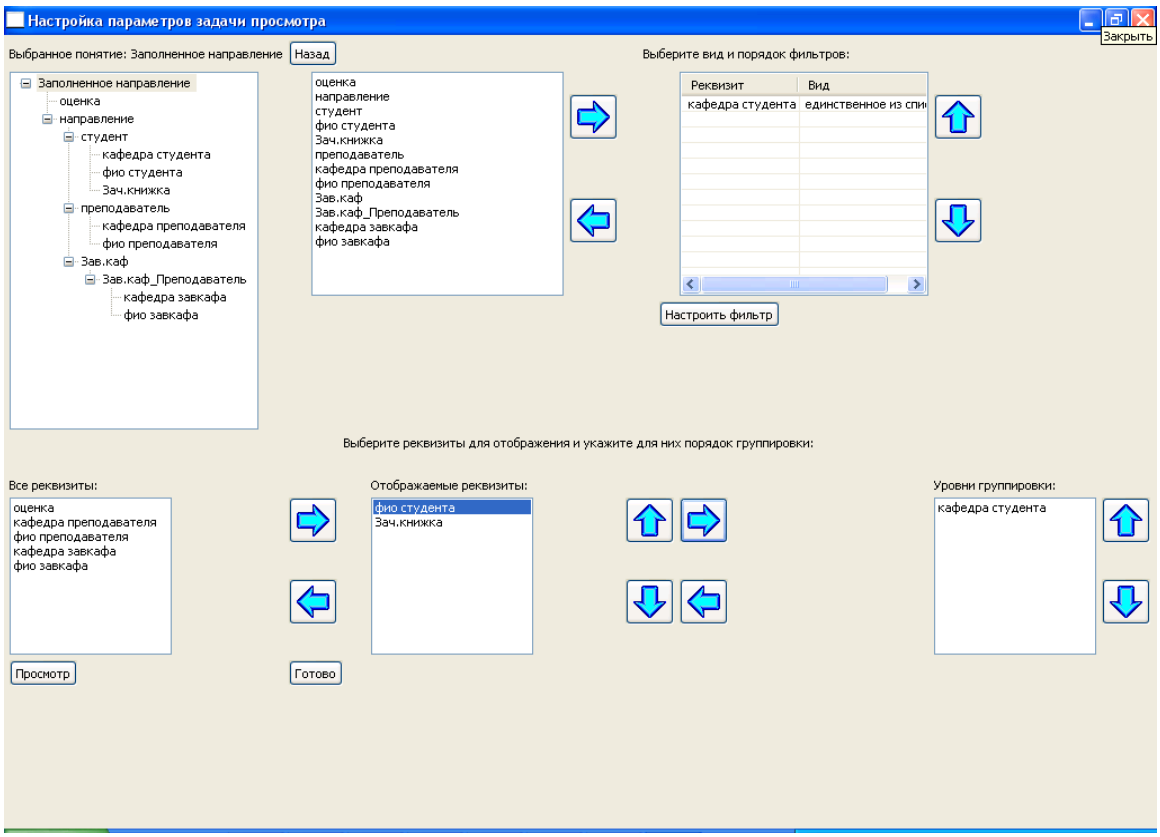

Рисунок 5. Форма указания атрибутов задачи

Заключение

Результатом проведенной работы является создание визуальной среды формирования пользовательского интерфейса, при помощи которой можно автоматизировать процесс разработки прикладных программ. Визуальная среда обеспечивает возможность создания сценария программы без программирования, простой и удобный интерфейс, низкую стоимость по сравнению с аналогами. Использование разработанного компонента системы автоматизированного проектирования в совокупности с другими ее компонентами позволит сократить время и затраты на разработку и модификацию информационных систем, а также повысить их качество за счет автоматизации многих этапов проектирования и программирования.

Литература

1. Кузнецов С. Д. Базы данных. Модели и языки. М.: Бином-Пресс, 2008 г. 720 с.
2. Уолтерс У. и др. SQL Server 2008: ускоренный курс для профессионалов: пер. с англ. М.: ООО «И.Д. Вильямс», 2009. 768с.
3. Кирстен В. и др. Постреляционная СУБД Cache 5. Объектно-ориентированная разработка приложений. М.: Бином, 2005 г. 416 с.
4. Большаков С. А. Концепция построения сайта преподавателя университета // Наука и образование. МГТУ им. Н.Э. Баумана. Электрон. журн. 2011. № 11

Режим доступа: http://technomag.bmstu.ru/doc/281085.html 77-30569/281085 (дата обращения 20.10.2013).

5. Орлов С.А., Цилькер Б.Я. Технологии разработки программного обеспечения. СПб.: Питер, 2012 г. 608 с.
6. Голощапов А.Л. Microsoft Visual Studio 2010. СПб: БХВ-Петербург, 2011 г. 544с.
7. Мацяшек Л.А., Лионг Б.Л. Практическая программная инженерия. М.: Бином, 2009 г. 956 с.
8. Тоноян С. А., Балдин А. В., Елисеев Д. В. Методика модернизации стандартных модулей типовой конфигурации на базе технологической платформы «1С:Предприятие 8» с минимальными доработками // Наука и образование. МГТУ им. Н.Э. Баумана. Электрон. журн. 2012. № 08. Режим доступа: http://technomag.bmstu.ru/doc/450231.html (дата обращения 20.10.2013).
9. Гусятников В.Н., Безруков А.И. Стандартизация и разработка программных систем. М.: Финансы и статистика, 2010 г. 288 с.
10. Информационная управляющая система МГТУ им. Н. Э. Баумана"Электронный университет". Концепция и реализация / Агеева Т. И., Балдин А. В., Барышников В. А. [и др.] ; ред. Федоров И. Б., Черненький В. М. М. : Изд-во МГТУ им. Н. Э. Баумана, 2009. 374 с.
11. Методика проектирования информационных систем с возможностью наращивания и быстрой модернизации на примере кафедры ВУЗа // Вестник информационных технологий в образовании: Сб. учебно-методических и научных работ. Выпуск 1. М.:МГТУ им. Н.Э. Баумана, 2005. С 100-117.
12. Гапанюк Ю.Е., Ревунков Г.И. Введение в XML-технологии. Учебное пособие. М.:МГТУ им. Н.Э. Баумана, 2010 г. Режим доступа http://sfm2007.narod.ru (Дата обращения 20.10.2013).
13. Виноградова М.В., Игушев ЭГ. Конструктор баз данных на основе сущностей и их реквизитов с возможностью нормализации // Наука и образование. МГТУ им. Н.Э. Баумана. Электрон. журн. 2012. № 01. Режим доступа: http://technomag.edu.ru/doc/242645.html (дата обращения 20.10.2013).
14. Виноградова М.В. Технология проектирования мультиаспектных информационно-образовательных систем // Международная конференция по вопросам обучения с применением технологий E-Learning ONLINE EDUCA MOSCOW 2007: Сб. тез. докл. Москва, 2007. С.97-99.