ЭЛЕМЕНТЫ СРЕДЫ РАЗРАБОТКИ ПРОГРАММНЫХ КОМПЛЕКСОВ НА ОСНОВЕ ОРГАНИЗАЦИИ МЕТАМОДЕЛИ ОБЪЕКТНОЙ СИСТЕМЫ

П.П. Олейник,
кандидат технических наук, доцент, системный архитектор программного обеспечения, ОАО «Астон»

E-mail: xsl@list.ru
Адрес: Ростовская обл., г. Шахты, пер. Бугроватый, 80

Рассмотрены основные элементы унифицированной среды быстрой разработки корпоративных информационных систем различных прикладных предметных областей. Описана структура уровня доступа к данным, методика проектирования приложений и реализованные проекты.

Ключевые слова: информационные системы (ИС), методы (паттерны) объектно-реляционного отображения, интегрированная среда разработки.

Введение

Неотъемлемой составляющей любого крупного предприятия или организации является информационная система. В широком смысле под информационной системой (ИС) понимают совокупность технического, программного и организационного обеспечения, а также персонала, предназначенная для того, чтобы своевременно обеспечивать надлежащих людей надлежащей информацией. Крупным подклассом ИС являются корпоративные информационные системы (КИС) под которыми понимают определенную совокупность методов и решений, используемых для создания единого информационного пространства управления и обеспечения деятельности компании [1]. Разработка и внедрение единой КИС, автоматизирующей все ключевые бизнес-процессы организации, является неотъемлемым условием функционирования предприятия любой сферы деятельности. В специфических условиях ведения бизнеса постоянно возникают задачи доработки и внесения изменений в существующий функционал. Именно поэтому многие крупные организации предпочитают собственными силами разрабатывать программное обеспечение и внедрять у себя на производстве. При этом выбор среды разработки и языка программирования во многом определяется именно возможностью быстрого внесения измене-

ний в систему. Основой КИС выступает хранилище информации, т.е. база данных (БД). Автоматизация ключевых бизнес-процессов организации заключается в создании приложений БД. В настоящее время широкое распространение получили реляционные системы управления базами данных (РСУБД) [2-3], а для проектирования и реализации приложений используют объектно-ориентированные языки программирования (ООЯП) [4-6]. Из-за различий в методах обработки информации (реляционная модель построена на основе теории множеств и оперирует группами строк, а ОО-парадигма основана на принципах инкапсуляции и проектирования независимых классов, экземпляры которых могут иметь состояние и методы обработки) возникает так называемое объектно-реляционного несоответствие (Object-Relational Impedance Mismatch) [5] для преодоления последствий которого используют методы (паттерны/шаблоны) объектно-реляционного отображения (ОРО, Object-Relational Mapping (ORM) Patterns) [5-6]. Все имеющиеся паттерны ОРО можно разделить на две большие категории: к первой относятся шаблоны, предназначенные для представления иерархии классов конкретной предметной области в виде набора таблиц реляционной БД (РБД) [5-6, 15]; работы второй категории предоставляют возможность организации метамодели объектной системы в РСУБД, которая используется для различных целей приложения [6-14, 17]. Статьи последней категории рассматривают различные аспекты реализации популярной в настоящее время технологии, названной «архитектура, управляемая моделью (Model Driven Architecture, MDA)», позволяющей описать различные аспекты приложения и предполагающей создание среды разработки приложений.

В статье представлена архитектура унифицированной среды разработки КИС, которая создавалась под руководством автора в течение нескольких последних лет и апробирована в качестве целевой платформы для реализации приложений различных предметных областей. Рассматриваемая среда быстрой разработки получила название SharpArchitect RAD Studio.

1. Критерии оптимальности среды разработки

Создание системы начнем с формирования требований, критериев оптимальности (КО), которым должна удовлетворять реализация среды разработки (КОср). Т.к. рассматриваемая среда предназначена для разработки корпоративных информационных систем, то созданные КИС также должны удовлетворять требованиям, предъявляемым к КИС [1] на основании которых сформулированы следующие КОср для SharpArchitect RAD Studio:

1. Предусмотреть возможность описания любых прикладных предметных областей. Заранее невозможно предугадать сферу применения системы, поэтому необходим общий способ описания прикладных предметных областей. Имеет смысл использовать структурные диаграммы (например, диаграммы классов) языка UML и на основе их интерпретации строить масштабируемые КИС [1, 22].

2. Реализовать поддержку различных архитектур приложений. В настоящее время существует множество различных архитектур от классической двухзвенной «файл-сервер» до многозвенной архитектуры с поддержкой «распределенных объектов» [4, 6]. Выбор оптимальной структуры во многом зависит от масштаба реализуемой КИС и аппаратных возможностей, имеющихся у предприятия. Выполнение данного КО позволит своевременно и с минимальными затратами масштабировать приложения при возрастающих потребностях организации.

3. Использовать современный язык программирования и современной СУБД. Популярной парадигмой сегодня является объектно-ориентированное программирование, поэтому использование соответствующего языка программирования является наиболее целесообразным [4]. Для хранения информации в настоящее время чаще всего используют реляционные системы управления базами данных (РСУБД). Применение современных технологий позволяет увеличить срок эксплуатации системы и сократить затраты на ее владение [2].

4. Реализовать инфраструктуру безопасности. Разграничение прав доступа к данным и отображение минимального набора информации, достаточного лишь для выполнения должностных обязанностей являются ключевыми особенностями разработки современных КИС [1]. Имеет смысл предусмотреть возможность двух способов проверки подлинности пользователей: 1) внутренней - основанной на учетных записей, имеющихся в КИС (чаще всего используется для Веб-приложений) и 2) встроенной – основанной на механизмах, предоставляемых операционной системой.

5. Обеспечить поддержку механизмов расширения функциональных возможностей среды раз-

работки. Современные языки программирования предоставляют обширную метаинформацию [4], что позволяет реализовать «плагинную» архитектуру, поддерживающую установку модулей расширений, представляющих собой dll-библиотеки [17]. Этот подход разрешает безгранично увеличивать функционал системы и её гибкость.

После того как представлены ключевые КОср можно перейти к описанию внутренних особенностей реализации.

2. Структура метамодели объектной системы

В широком смысле слова метамодель – это модель, позволяющая описать другие модели. В нашем понимании метамоделью является объектная модель, которая описывает модель прикладной предметной области. В основе рассматриваемой среды разработки положена единая метамодель системы, реализованная в виде иерархии классов языка C#. Отметим, что сама метамодель хранится в виде таблиц РБД [7-14, 16-17].

При реализации были сформулированы следующие критерии оптимальности метамодели (КОмм) [16]:

1. Разработать единую иерархию, позволяющую представить ключевые элементы объектно-ориентированной парадигмы, такие как классы, атрибуты, наследование (одиночное и множественное), ассоциации и т.п.

2. Предусмотреть возможность предоставления базовых (системных) классов, в которых реализован наиболее общий функционал

3. Разработать иерархию атомарных литеральных типов, представляющих наиболее распространенные типы данных современных объектно-ориентированных языков программирования.

Рассмотрим требования более подробно. Реализация требования КОмм1 позволит унифицировано обрабатывать все элементы с помощью различных структур данных, например с помощью итераторов при формировании цикла. При этом пользователю предоставляется возможность применять хорошо зарекомендовавшие себя объектно-ориентированные подходы, как например наследование и организация ассоциаций.

КОмм2 требует наличия системных (встроенных) классов. В общем случае пользователи не могут модифицировать поведение этих классов, но имеют возможность наследоваться от них. При этом наличие множественного наследования предоставляет широкие возможности моделирования задач прикладной предметной области.

Современные ООЯП имеют в своем арсенале богатый набор встроенных атомарных литеральных типов. Поэтому необходимо предоставить подобный функционал в разрабатываемой системе, что соответствует требованию КОмм3 [6, 18].

Идея представления метамодели объектной системы не нова. В работах, посвященных стандарту ODMG 3.0 [19-20], и в работах, описывающих SQL:2003 [21], имеются определённые решения. Но первый стандарт ориентирован лишь на удобство обработки метамодели с помощью встроенных коллекций и итераторов, а второй – на представление сложных типов данных в РСУБД. Поэтому было принято решение самостоятельно разработать иерархию классов, полностью удовлетворяющую поставленной задаче. Результат представленный в данной работе основывается на работах [7-8, 11-13, 16].

На *рис. 1*, в виде диаграммы классов UML [4, 22], представлена разработанная иерархия. Рассмотрим ее соответствие выделенным критериям оптимальности. Иерархия соответствует требованиям КОмм1, т.к. содержит все ключевые элементы ОО-парадигмы: классы, атрибуты, ассоциации и предоставляет возможность реализации. Выполнены и требования КОмм2, т.к. имеется возможность предоставления базовых (системных) классов, реализующих наиболее общий функционал. Разработанная иерархия классов представляет атомарные литеральные типы, которые описывают распространенные типы данных современных ООЯП, т.е. выполнены требования КОмм3. Следовательно, разработанная иерархия удовлетворяет всем выделенным критериям оптимальности метамодели.

3. Организация уровня доступа к данным

Общий процесс разработки корпоративной информационной системы предполагает создание экземпляров классов, образующих метамодель. Затем самой системой выполняется интерпретация этой информации и построения физического уровня доступа к данным, который реализуется в виде набора таблиц РСУБД. Это выполняется на основе реализации методов объектно-реляционного отображения (ОРО).

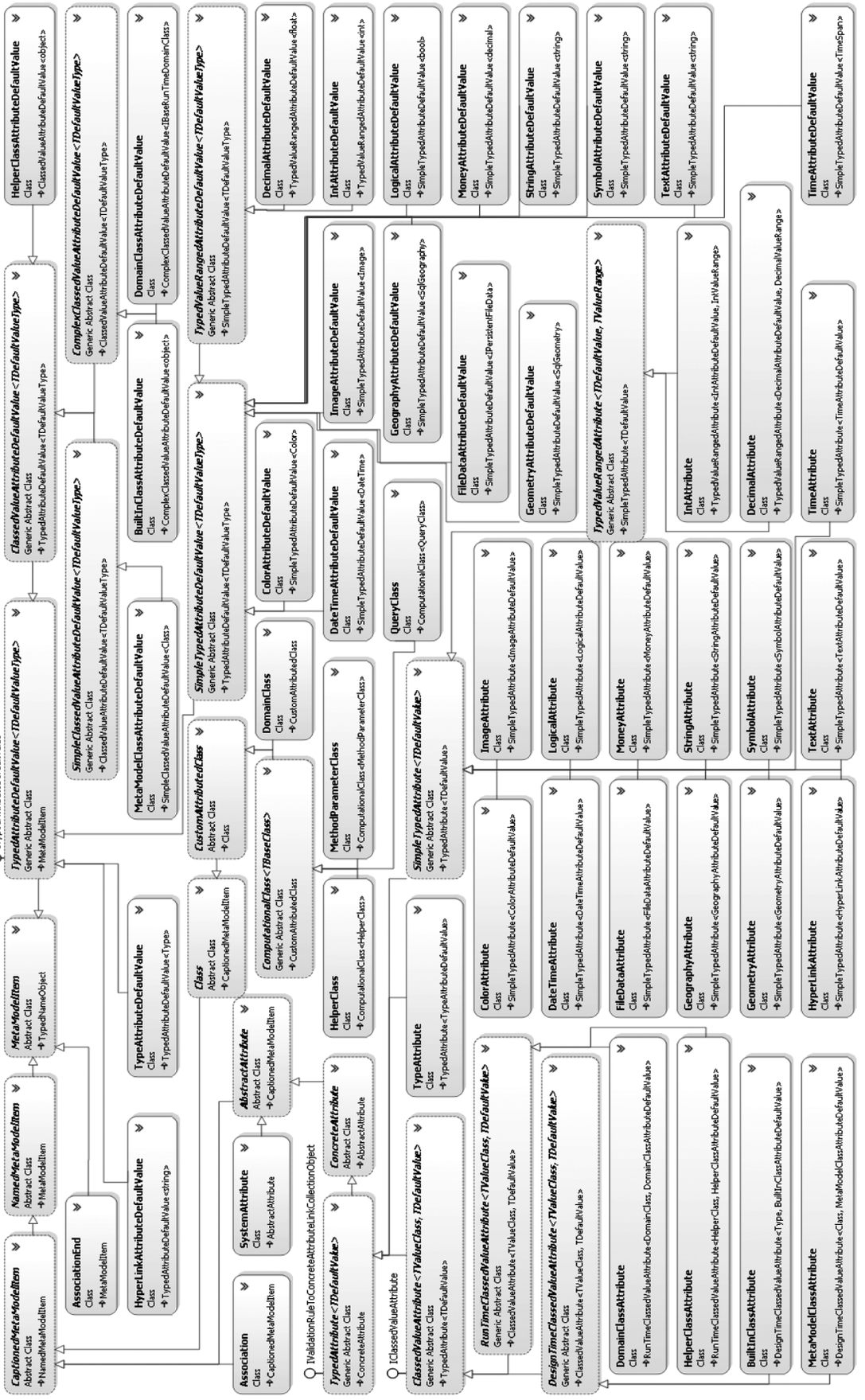

Рис. 1. Иерархия классов метамодели объектной системы

Рассмотрим основные методы ОРО, используемые для представления иерархии классов в виде таблиц РБД (*рис. 2*) [5-6, 15, 17]. Метод Наследование с одной таблицей (Single Table Inheritance) физически представляет иерархию наследования классов в виде одной таблицы реляционной базы данных, столбцы которой соответствуют атрибутам всех классов, входящих в иерархию и позволяет отобразить структуру наследования и минимизировать количество соединений, которые необходимо выполнить для извлечения информации. В данном методе каждому экземпляру класса соответствует одна строка таблицы. При создании объекта значения заносятся только в те столбцы таблицы, которые соответствуют атрибутам класса, а все остальные остаются пустыми (имеют null-значения).

Метод имеет достоинства:

✦ В структуру БД присутствует лишь одна таблица, представляющая все классы иерархии.

✦ Для извлечения экземпляров классов иерархии нет необходимости выполнять соединение таблиц.

✦ Перемещение полей из базового класса в производный (так же из производного в базовый) не требует внесения изменений в структуру таблицы.

Методу присущи недостатки:

✧ При изучении структуры таблиц БД могут возникнуть проблемы, ведь не все имеющиеся столбцы таблицы предназначены для описания каждого класса предметной области. Это усложняет процесс доработки системы в будущем.

✧ При наличии глубокой иерархии наследования с большим количеством атрибутов многие столбцы могут иметь пустые значения (null-маркеры). Это приводит к неэффективному использованию свободного пространства в БД. Однако современные СУБД способны сжимать строки, содержащие большое количество отсутствующих значений.

✧ Итоговая таблица может оказаться слишком большой и содержать огромное число столбцов. Основной способ оптимизировать запрос (сократить время выполнения) – создать покрывающий индекс. Однако множество индексов и большое количество запросов к одной таблице способны привести к частыми блокировками, что будет оказывать негативное влияние на производительность программного приложения.

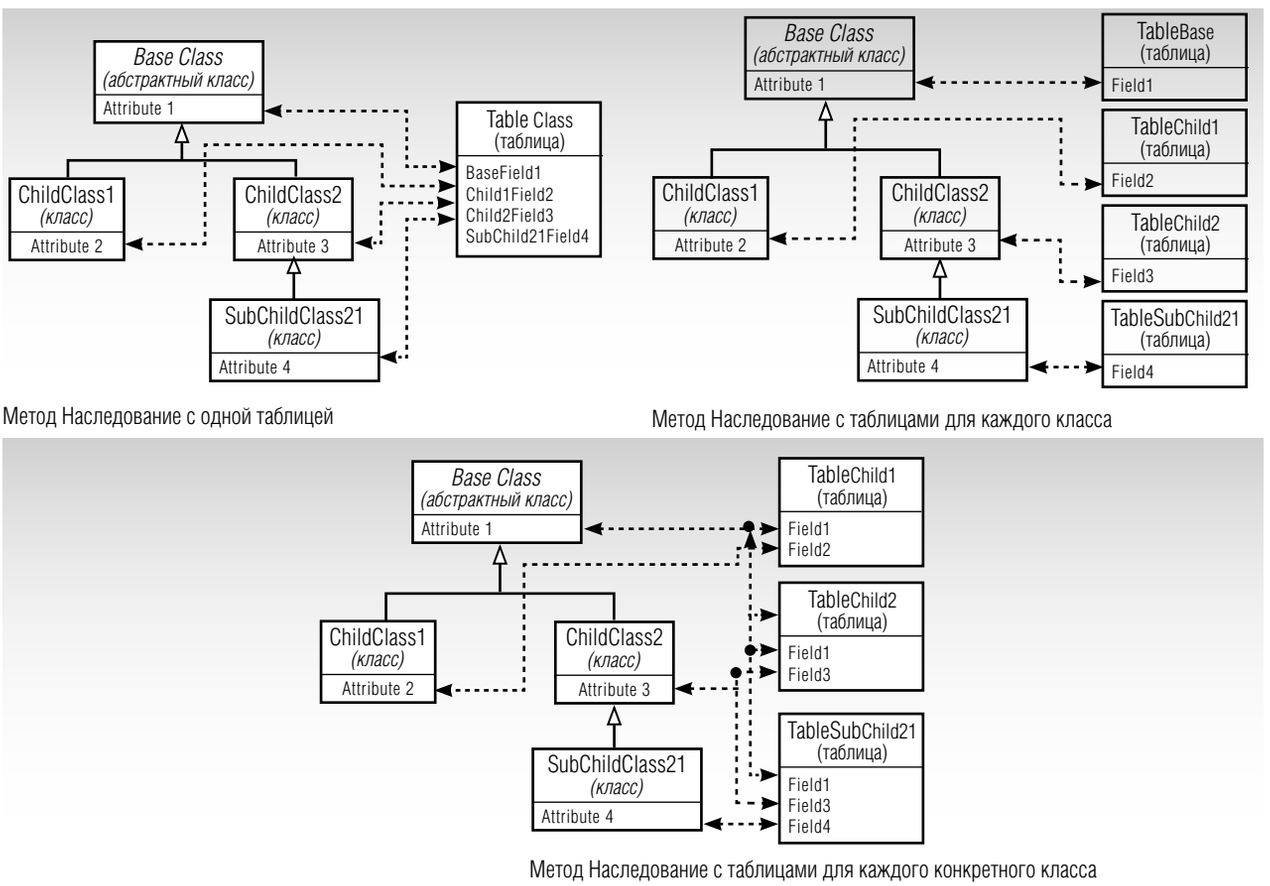

Рис. 2. Методы объектно-реляционного отображения

Альтернативным является метод Наследование с таблицами для каждого класса (Class Table Inheritance), представляющий иерархию классов по одной таблице для любого класса (как абстрактного так и реализованного). Атрибуты класса отображаются непосредственно на столбцы соответствующей таблицы. При использовании данного метода ключевой является задача соединения соответствующих строк нескольких таблиц БД, представляющих единый объект предметной области.

Методу присущи преимущества:

✦ В каждой таблице присутствуют лишь поля, соответствующие атрибутам определённого класса. Поэтому таблицы легки в понимании и занимают мало места на жёстком диске.

✦ Взаимосвязь между объектной моделью и схемой базы данных проста и понятна.

Однако имеются недостатки:

✧ При создании экземпляра определённого класса необходимо выполнить загрузку данных из несколько таблиц, что требует либо их естественного соединения, либо множества обращений к базе данных с последующим соединением результатов в оперативной памяти.

✧ Перемещение полей в производный класс или суперкласс требует изменения структуры нескольких таблиц.

✧ Таблицы суперклассов могут стать слабым местом в вопросах производительности, поскольку доступ к таким таблицам будет осуществляться слишком часто, что приведёт к множеству блокировок.

✧ Высокая степень нормализации может стать препятствием к выполнению незапланированных заранее запросов.

Метод Наследование с таблицами для каждого конкретного класса (Concrete Table Inheritance) представляет иерархию наследования классов, использующий по одной таблице для каждого реализованного (неабстрактного) класса этой иерархии. С практической точки зрения данный метод предполагает, что каждый экземпляр класса (объект), который находится в оперативной памяти, будет отображён на отдельную строку таблицы. При этом каждая таблица в нашем случае содержит столбцы, соответствующие атрибутам как конкретного класса, так всех его предков.

Преимуществами являются то, что:

✦ Каждая таблица не содержит лишних полей, вследствие чего ее удобно использовать в других приложениях, в которых не использован инструмент объектно-реляционного отображения.

✦ При создании объектов определённого класса в памяти приложения и извлечения данных из реляционной базы данных выборка выполнятся из одной таблицы, т.е. не требуется выполнять реляционные соединения.

✦ Доступ к таблице осуществляется только в случае доступа к конкретному классу, что позволяет снизить количество блокировок, накладываемых на таблицу и распределить нагрузку на систему.

При этом имеются недостатки:

✧ Первичные ключи могут быть неудобны в обработке.

✧ Отсутствует возможность моделировать отношения (связи) между абстрактными классами.

✧ Если атрибуты классов перемещаются между суперклассами и производными классами, требуется изменять структуру нескольких таблиц. Эти изменения будут не так часты, как в случае наследования с таблицами для каждого класса, однако их нельзя игнорировать (в отличие от метода Наследование с одной таблицей, в котором эти изменения вообще отсутствуют).

✧ Если в суперклассе изменить определение хотя бы одного атрибута (например, поменять тип данных), это потребует изменить структуру каждой таблицы, представляющей производный класс, поскольку поля суперкласса дублируются во всех таблицах его производных классов.

✧ При реализации метода поиска данных в абстрактном классе потребуется просматривать все таблицы, представляющие экземпляры производных классов. Это требует большого количества обращений к базе данных.

Выбор паттерна ОРО зависит от опций, указанных разработчиком КИС в метамодели.

4. Реализованные проекты

Проект SharpArchitect RAD Studio разрабатывался в последние несколько лет и был использован как для научных целей, так и с целью практической разработки приложений. Для тестирования функциональных особенностей и корректности работы механизма ОРО создана соответствующая модель, представленная в работе [15]. В среде разработано 12 дипломных проектов, выполненных в качестве квалификационных работ в государственных

ВУЗах РФ: в ШИ (ф) ЮГТУ (НПИ) и ИГХТУ. Каждый программный продукт представляет из себя учетную систему, содержащую от 30 до 50 классов предметной области, каждый их которых хранит не менее 100 экземпляров.

В качестве первой крупной реализации создана информационная система проведения научных конференций SharpArchitect Scientific Conference Manager [23], которая используется при проведении конференции «Объектные системы» (www.objectsystems.ru). Работа ведется с 2010 года и реализованы все ключевые модули, позволяющие отслеживать жизненный цикл научной статьи от пересылки ее автором в Оргкомитет, рецензирования и до отслеживания статуса отосланного сборника научных трудов по почте [24-25]. В настоящий момент в системе храниться информация более чем о 300 авторах, 200 статьях и 7 проведенных конференций.

Вторым крупным проектом является реализация платформы создания и управления торговыми биржевыми роботами, архитектура которой представлена в работах [26]. На базе платформы создано более 20 высокочастотных торговых стратегий, одновременно торгующих в режиме реального времени на бирже РТС. Работа системы протестирована на основе реализации алгоритмов технического анализа, методов статистического анализа и различных их комбинаций. В качестве тестового массива данных использованы тиковые котировки, составляющие более 1 Гб данных в сутки.

Заключение

Все реализованные проекты представляют собой выполняемые Windows-приложения, предоставляющие развитый интуитивно-понятный пользовательский интерфейс. При этом использована возможность создания классических двухзвенных приложений на основе архитектуры «клиент-сервер» и трехзвенных, с выделенным сервером приложений. В настоящее время все большую популярность получают Веб-технологии. Поэтому ведутся работы по реализации Веб-модуля, позволяющего генерировать соответствующие приложения.

Также разрабатывается официальный Веб-сайт www.sharparchitect.com, который будет посвящен реализованным проектам в SharpArchitect RAD Studio. Планируется разработать мультимедиа руководство по функциональным возможностям и по принципам разработки приложений. Первый обучающий видео-ролик доступен заинтересованному читателю по адресу https://www.youtube.com/watch?v=BtfEGdSKHEo. ■

Литература

1. Олейник П. П. Корпоративные информационные системы: Учебник для вузов. Стандарт третьего поколения. – СПб.: Питер, 2012.
2. Коннолли Т., Бегг К. Базы данных. Проектирование, реализация и сопровождение. Теория и практика. 3-е издание: Пер. с англ. – М.: Издательский дом «Вильямс», 2003.
3. Дейт К.Дж. Введение в системы баз данных, 7-е издание: Пер. с англ. – М.: Издательский дом «Вильямс», 2001.
4. Грэхем И. Объектно-ориентированные методы. Принципы и практика. 3-е издание: Пер. с англ. – М.: Издательский дом «Вильямс», 2004.
5. Ambler S.W. Agile Database Techniques – Effective Strategies for the Agile Software. – John Wiley & Sons, 2003.
6. Флауер М. Архитектура корпоративных программных приложений: Пер. с англ. – М.: Издательский дом «Вильямс», 2004.
7. Олейник П.П. Представление метамодели объектной системы в реляционной базе данных // Известия высших учебных заведений. Северо-Кавказский регион. Спецвыпуск «Математическое моделирование и компьютерные технологии», 2005. – С. 3-8.
8. Олейник П.П. Организация метамодели объектной системы на основе реляционной СУБД // Научное творчество молодёжи: Материалы X Всероссийской научно-практической конференции (21-22 апреля 2006 г.) Ч. 1. – Томск: Изд-во Томского ун-та, 2006. – С. 87-88.
9. Олейник П.П. Критерии построения унифицированных хранимых процедур выборки в реляционной БД // Математические методы и информационные технологии в экономике, социологии и образовании: Сборник статей XIX Международной научно-технической конференции, г. Пенза, 24-25 мая 2007 г. / Пензенская гос. технолог. акад. – Пенза, 2007. – С. 167-170.

10. Олейник П.П. Об одном способе реализации процедур выборки данных // Международная научно-практическая конференция ИТО Поволжье – 2007. Российский научный семинар «Методы информационных технологий, математического моделирования и компьютерной математики в фундаментальных и прикладных научных исследованиях». (18-21 июня 2007 г.) // Материалы конференции и труды семинара. – Казань: Изд-во «Филиантъ», 2007. – С. 65-67.
11. Олейник П.П. К вопросу о необходимости проектирования иерархии атомарных литеральных типов для объектной системы, организованной в РСУБД // Информационно-вычислительные технологии и их приложения. IX Международная научно-техническая конференция: сборник статей. – Пенза: РИО ПГСХА, 2008. – С. 201-205.
12. Олейник П.П. Организация иерархии атомарных литеральных типов в объектной системе, построенной на основе РСУБД // Программирование. – 2009. – № 4. – С. 73-80.
13. Oleynik P.P. Implementation of the Hierarchy of Atomic Literal Types in an Object System Based of RDBMS // Programming and Computer Software. – 2009. – Vol. 35, No.4. – P. 235-240.
14. Олейник П.П. Организация серверного уровня доступа к данным объектной системы, построенной на основе РСУБД // Автоматизация и современные технологии. Ежемесячный межотраслевой научно-технический журнал. – 2009. – № 6. – С. 15-19.
15. Олейник П.П. Унифицированная модель для тестирования инструментов объектно-реляционного отображения // Объектные системы - 2011: материалы III Международной научно-практической конференции (Ростов-на-Дону, 10-12 мая 2011 г.) / Под общ. ред. П.П. Олейника. – Ростов-на-Дону, 2011. – С. 65-69.
16. Олейник П.П. Иерархия классов метамодели объектной системы // Объектные системы – 2012: материалы VI Международной научно-практической конференции (Ростов-на-Дону, 10-12 мая 2012 г.) / Под общ. ред. П.П. Олейника. – Ростов-на-Дону: ШИ ЮРГТУ (НПИ), 2012. – С. 37-40.
17. Олейник П.П. Методика построения унифицированных трёхзвенных объектно-ориентированных приложений. – Saarbrücken, Germany: LAP LAMBERT Academic Publishing, 2011.
18. Фаулер М., Бек К., Брант Д., Робертс Д., Апдайк У. Рефакторинг: улучшение существующего кода. – М: Символ-Плюс, 2008.
19. Cattell R.G., Barry D.K. The Object Data Standard:ODMG 3.0. – Morgan Kaufmann Publishers, 2000.
20. Джордан Д. Обработка объектных баз данных на C++. Программирование по стандарту ODMG: Пер. с англ.: Уч. пос. – М.: Издательский дом «Вильямс», 2001.
21. Calero C., Ruiz F. and oth. An ontological approach to describe the SQL:2003 ob-ject-relational features, Ontologies for Software Engineering and Software Technology. – Berlin Heidelberg: Springer, 2006. – P. 197-215.
22. Новиков Ф.А., Иванов Д.Ю. Моделирование на UML. Теория, практика, видеокурс. – СПб.: Профессиональная литература, Наука и Техника, 2010.
23. Олейник П.П. Информационная система проведения научных конференций SharpArchitect Scientific Conference Manager // Свидетельство об официальной регистрации программы для ЭВМ № 2013610565, 9 января 2013 г. – М.: ВНТИЦ, 2013.
24. Олейник П.П. Опыт применения инструментов объектно-реляционного отображения при разработке информационной системы каталогизирования научных работ // Информационные технологии в науке, экономике и образовании: материалы 5-й Всероссийской научно-практической конференции 2-3 сентября 2010 года / под. ред. О.Б. Кудряшовой; Алт. гос. техн. ун-т, БТИ. – Бийск: Изд-во Алт. гос. техн. ун-та, 2010. – С. 102-105.
25. Олейник П.П., Игумнов Е.А., Свечкарёв Е.А. Опыт проектирования информационной системы для каталогизирования научных работ при проведении международных конференций // Объектные системы - 2010: материалы II Международной научно-практической конференции. Россия, Ростов-на-Дону, 10-12 ноября 2010 г., Ростов-на-Дону, 2010. – С. 48-51.
26. Oleynik P.P. The implementation of the framework to create and manage exchange trading robots // Object Systems – 2011 (English session): Proceedings of the Fourth International Theoretical and Practical Conference. Rostov-on-Don, Russia, 10-12 November, 2011. Edited by Pavel P. Oleynik, P. 42-45.