УДК 658.6.01156:681.3.06

В.А. Драганов, А.Я. Клименко, А.В. Фофонов

Многослойная архитектура как основа для создания автоматизированных коммерческо-технологических систем

Дано определение и выявлены особенности проектирования и реализации автоматизированных коммерческо-технологических систем. Выделены основные требования к качествам систем подобного рода. На основе этих требований представлена многослойная архитектура коммерческо-технологической системы.

Ключевые слова: разработка программного обеспечения, многослойная архитектура, торговые системы, автоматизация, типовые решения.

Введение

Основным назначением автоматизированный коммерческо-технологической системы (АКТС) являются автоматизация как процесса совершения сделки (передача прав собственности на товар) так и управление технологией передачи самого товара, зачастую с использованием достаточно сложных технических средств. Сегодня на рынке представлено большое разнообразие АКТС. Это торговые системы супермаркетов, системы отпуска топлива на автозаправочных станциях и нефтебазах, системы бронирования и продажи билетов, системы контроля доступа и др. Однако большинство представленных решений являются узкоспециализированными, что, вкупе с недостаточной гибкостью, приводит к сокращению периода их жизни.

Известно, что основным этапом разработки, определяющим качества будущего программного продукта (надежность, гибкость, масштабируемость, безопасность и пр.), является моделирование его архитектуры. В настоящей статье изложены результаты многолетних исследований в этой области, которые позволили создать библиотеку типовых программных решений, позволяющих существенно сократить сроки разработки и внедрения АКТС.

Требования к системе. Первый взгляд на архитектуру

Разработка любой системы начинается с определения набора функциональных и нефункциональных требований, предъявляемых к ней со стороны заказчиков [1]. Мы не будем касаться последних, так как они представлены в разнообразных стандартах, регламентирующих разработку АСУ. Функциональные требования к АКТС представлены следу-ющим наборов бизнес-задач:

1. Хранение данных и манипулирование информацией о предметной области (участниках и предмете сделки) в виде справочников покупателей, продавцов, поставщиков товаров и услуг.
2. Обеспечение полного контроля над процессом проведения сделки со стороны оператора.
3. Обеспечение взаимодействия со специализированными устройствами (например, отпуск топлива с помощью топливо-раздаточных колонок, использование считывателя карт для идентификации клиента, печать чека на кассовом регистраторе, управление турникетом и т.п.).
4. Одновременный доступ к ресурсам системы со стороны нескольких пользователей.
5. Предоставление разнообразных отчетов о проведенных транзакциях для контроля над проведенными операциями, анализа и планирования.
6. Интеграция с вышестоящими системами, позволяющая обобщить результаты эксплуатации нескольких систем.

Эти требования позволяют сделать ряд выводов об архитектуре АКТС (под архитектурой мы понимаем организационную структуру системы, включающую ее разбиение на части, связи между этими частями, механизмы взаимодействия и основные принципы проектирования системы [2]).

Предполагается, что система должна поддерживать ведение справочников и учет транзакций. В таком случае типичным подходом является использование системы управления базой данных (СУБД). Требование одновременной работы нескольких пользователей говорит о необходимости поддержки территориальной распределенности системы. Кроме этого, одновременный доступ к данным и устройствам предполагает использование многопо-

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

точной модели параллельных вычислений. Таким образом, АКТС следует рассматривать как распределенную систему, построенную с использованием СУБД и поддерживающую параллельное выполнение операций.

Расслоение архитектуры

Разработчики больших программных систем неизбежно сталкиваются с проблемой сложности. Типичным методом борьбы со сложностью является декомпозиция системы на ряд подсистем, каждая из которых обеспечивает выполнение определенной функциональности. Концепция «Расслоение системы» (*multi-layer architecture*) помогает организовать работу подсистем наиболее эффективным образом. Ее суть заключается в том, что самые важные подсистемы представляются в виде слоев (*layers*), расположенных друг над другом. Каждый вышестоящий слой использует функции, предоставляемые нижестоящим слоем, однако нижние слои «не осведомлены» о существовании верхних. Такая организация позволяет манипулировать подсистемами независимо друг от друга, дорабатывая, изменяя или замещая их. В результате архитектура системы становится более гибкой, что позволяет разработчикам постоянно адаптировать ее к изменяющимся окружающим условиям [2].

Поскольку АКТС использует базу данных для хранения информации предметной области, выполняет манипуляции над ней и организует пользовательский интерфейс, в ее основе лежит общепринятая трехслойная модель «Представление» – «Логика предметной области» – «Источник данных». Однако особенности АКТС требуют выделения нескольких дополнительных слоев (рис. 1):

1.«Устройства»: требование организации работы с разнообразным оборудованием предполагает выделение специального слоя, обеспечивающего прозрачное управление устройствами.

2.«Коммуникации»: так как АКТС представляет собой распределенную систему, наиболее гибким решением будет организация такого взаимодействия в отдельном слое «Коммуникации».

3.«Взаимодействие с бэк-офисом»: современные тенденции в развитии АКТС предполагают разделение системы на фронт- и бэк-офис. При этом к функциям фронт-офиса относят обеспечение взаимодействия с клиентом и кассовые функции, а бэк-офису поручают роль аналитического центра с функциями генерации отчетов. Бэк-офис часто поставляют в роли самостоятельного приложения, которое может взаимодействовать с несколькими фронт-офисными точками, осуществляя сбор и накопление необходимой информации. В архитектуре АКТС такое разделение выражается в появлении дополнительного слоя «Взаимодействие с бэк-офисом».

Рис. 1. Многослойная архитектура АКТС

Детальное описание слоев

Дальнейший процесс декомпозиции приводит к выделению основных пакетов классов, обеспечивающих выполнение функциональности каждого слоя. Гибкость моделируемой архитектуры на этом уровне заключается в правильной группировке связанных классов и снижении зависимости между этими группами. Это достигается применением типовых решений уровня проектирования, аккумулирующих накопленный опыт разработки программных систем [3, 4].

Слой «Представление». Данный слой отвечает за взаимодействие с пользователем: отображение хранящейся в системе информации и интерпретацию вводимых пользователем команд с преобразованием их в соответствующие операции над данными системы.

Функционально данный слой разделяется на два пакета (рис. 2): «Отображение» и «Координаторы приложения». Пакет «Отображение» содержит элементы, отвечающие за визуализацию информации объектов предметной области. Они используют функции программного интерфейса операционной системы (API) или готовые каркасы (MFC, WTL и прочие) для размещения и компоновки отображаемых элементов. Действия пользователя по манипулированию информацией передаются элементам пакета «Координаторы приложения».

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

Координатор приложения – это специализированный объект, отвечающий за формирование команды или набора команд объектам предметной области, а также кэширование информации для отображения. Координаторы необходимы для оптимизации обмена данными между разнесенными частями системы. При запуске приложения они загружают требуемую информацию из слоя, отвечающего за работу предметной области, и следят за ее изменением. Таким образом, за счет введения дополнительного элемента обеспечивается полное отделение отображения от предметной области. Кроме этого, координаторы могут выполнять функции подготовки загруженной информации к визуализации без отправления множества запросов в слой «Логика предметной области».

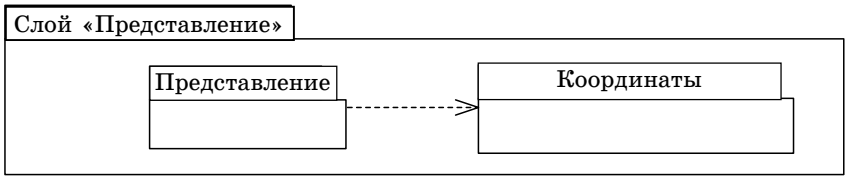

Рис. 2. Слой «Представление»

Слой «Коммуникации». Данный слой отвечает за обеспечение распределенного взаимодействия между элементами системы (рис. 3).

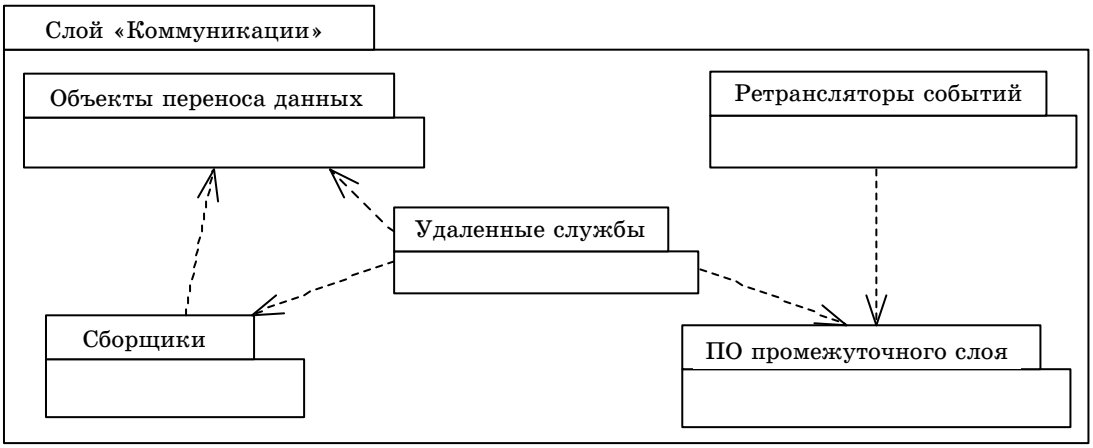

Рис. 3. Слой «Коммуникации»

Все используемые здесь типовые решения подчинены идее оптимизации такого взаимодействия за счет снижения количества запросов данных. Для этого применяются специализированные «интерфейсы удаленного доступа» [2], позволяющие сгруппировать запрашиваемую информацию. В тесном контакте с ними используются «объекты переноса данных» [2], гарантирующие возможность передачи максимального количества информации за один вызов. Для заполнения объектов переноса данных информацией из предметной области применяются специализированные сборщики, являющиеся вариантом типового решения «Преобразователь» [2].

Применение распределенной модели предполагает, что приложения-клиенты должны оповещаться о событиях, происходящих в системе. Выполнение этого требования реализуется с помощью широковещательной рассылки уведомлений о событиях через объекты-ретрансляторы.

Слой «Логика предметной области». Поскольку каждая система предъявляет индивидуальные функциональные требования, в общей архитектуре должен быть предусмотрен отдельный «участок», где эти требования будут реализованы. Очевидно, что данный слой должен быть предельно независим от остальных, чтобы настройка каких-либо параметров системы в наименьшей степени повлияла на логику ее работы. Это достигается применением типового решения «Отделенный интерфейс» [2] по отношению ко всем нижележащим слоям (рис. 4). Таким образом, в слое «Логика предметной области» определяется достаточно информации для его независимого существования.

Для обеспечения управления базой данных в слое «Бизнес-логика» должны быть представлены интерфейсы «Источника данных», позволяющие получать доступ к данным и управлять транзакциями. Кроме этого, требуется пакет, представляющий интерфейсы

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

устройств, доступных системе, и пакет, определяющий интерфейсы для взаимодействия с вышестоящей системой.

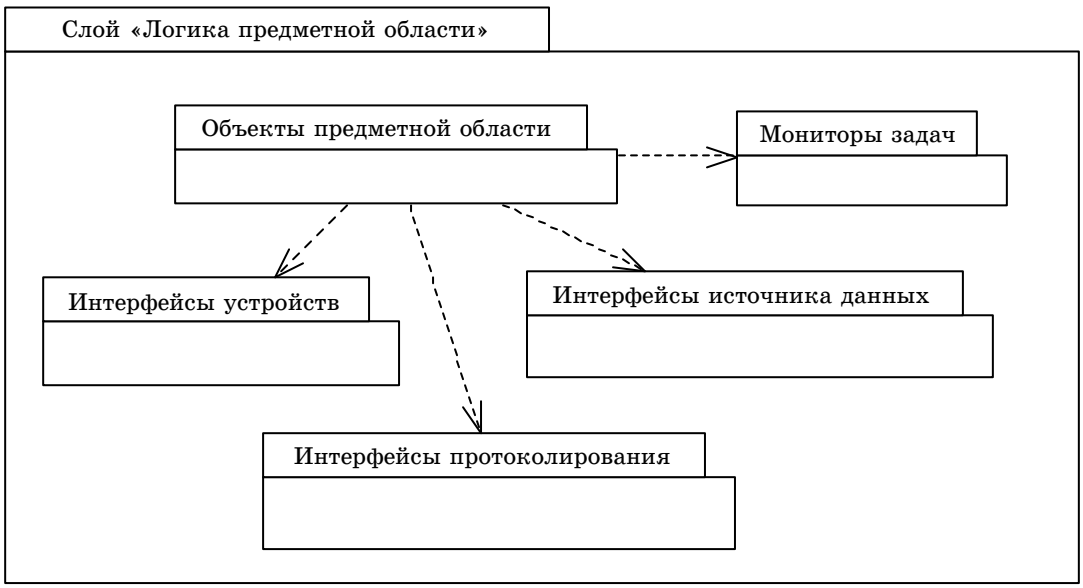

Рис. 4. Слой «Логика предметной области»

Работа с устройствами предполагает возможность асинхронного обмена данными и командами. Поэтому для эффективного функционирования система должна обеспечивать выполнение параллельных задач, работающих длительное время на протяжении нескольких транзакций. Для решения этой проблемы вводится концепция объектов-мониторов [5], следящих за выполнением длительной задачи. Монитор может открывать и закрывать транзакции для обновления данных в базе, при этом предполагается, что в системе в один момент времени выполняется только одна транзакция.

Слой «Источник данных» несет ответственность за функции взаимодействия с СУБД, такие как доступ к данным, мониторинг транзакций и пр. (рис. 5).

Рис. 5. Слой «Источник данных»

Основная сложность в реализации данного слоя состоит в том, чтобы обеспечить преобразование данных из табличной формы, которая используется для хранения в реляционных БД в объектное представление, удобное для организации работы слоя «Логика предметной области».

Общий подход к организации системы на основе базы данных состоит в написании классов-оберток для сущностей, представляемых таблицами, и использовании курсоров для управления данными. Однако для систем, обладающих сложной логикой предметной области (наследование, использование коллекций вложенных объектов, применение составных объектов, сохраняющихся в разных таблицах), такой подход становится неэффективным, так как снижает возможности объектно-ориентированной разработки.

Для решения этой проблемы используется целый набор типовых решений: «Преобразователи» [2] производят сложное отображение данных из таблиц в объекты; «Коллекция объектов» позволяет обеспечить уникальность загруженного экземпляра объекта, выступая при этом в качестве кэша извлеченных данных. Типовое решение «Единица работы» [2] обеспечивает транзакционную работу системы и координацию записи изменений в ба-

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

зу данных. Для минимизации лишних обращений к БД используется механизм «Загрузки по требованию» [2].

Способы отображения сложных объектов в записи таблиц достаточно разнообразны и многочисленны, и использование того или иного способа определяется многими параметрами в каждом конкретном случае.

Слой «Устройства» описывает особенности управления разнообразным оборудованием, подключенным к системе (рис. 6).

Рис. 6. Слой «Устройства»

Сложность взаимодействия с множеством устройств заключается в необходимости поддержки полного или частичного функционирования конкретного типа устройства с возможностью гибкого масштабирования и конфигурирования. В такой ситуации для ядра системы необходимо предоставить унифицированный интерфейс нижнего уровня с набором оберток-адаптеров для конкретного типа устройства.

Наиболее рациональным подходом представляется выделение данного слоя в отдельный компонент с использованием типового решения «Дополнительный модуль» [2] для специализации управления каждым устройством. За счет этого осуществляется сокрытие сложностей логики работы с устройствами от остальной системы, что позволяет организовать в данном компоненте распределенное управление устройствами путем каскадирования, а также дает возможность неоднократного использования самого компонента в пределах системы.

Слой «Взаимодействие с бэк-офисом». Задача данного слоя (рис. 7) состоит в том, чтобы обеспечить передачу информации о проведенных сделках и произошедших событиях вышестоящей программе, а также загрузку поступающих данных (списков разрешений, правил работы и пр.).

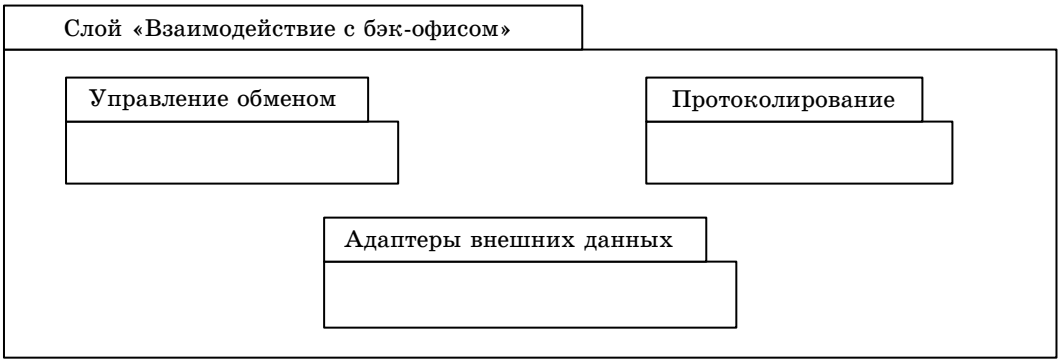

Рис. 7. Слой «Взаимодействие с бэк-офисом»

Пакет «Протоколирование» содержит классы, обеспечивающие сбор информации о сделках из объектов предметной области и преобразование их в форму протокола. Для повышения качества генерируемых отчетов дополнительно требуется передача информации об изменениях справочников объектов.

Для того чтобы организовать прием данных от вышестоящей программы (или бэк-офиса), требуется специальный управляющий модуль, осуществляющий проверку наличия новых данных, а также пакет загрузчиков, производящих загрузку и преобразование поступающей внешней информации в объектную модель системы.

Обеспечение взаимодействия между слоями. Помимо описания основных слоев, следует рассмотреть особенности обмена данными между ними. В большинстве случаев такой обмен является двунаправленным. Запрос информации вышестоящим слоем у нижестоя-

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

щего подобен обычной команде из-за того, что верхний слой осведомлен о существовании нижнего. Однако иногда возникают ситуации, когда требуется организовать обратный асинхронный обмен. Рассмотрим ситуацию, когда один из датчиков в слое «Устройства» завершил цикл опроса данных и обладает информацией, которую необходимо представить пользователю. При этом в соответствии с ограничениями многослойной архитектуры слой «Устройства» не знает, куда отправить полученные данные.

Для решения этой задачи применяется механизм «Издатель/Подписчик» [5], при котором вышестоящий слой производит подписку на интересующие его события нижестоящего слоя через обобщенный интерфейс. В случае возникновения события издатель (нижестоящий слой) уведомляет всех подписчиков.

Предлагаемая схема организации взаимодействия объектов представлена на рис. 8. Она основана на типовом решении «Наблюдатель» [3].

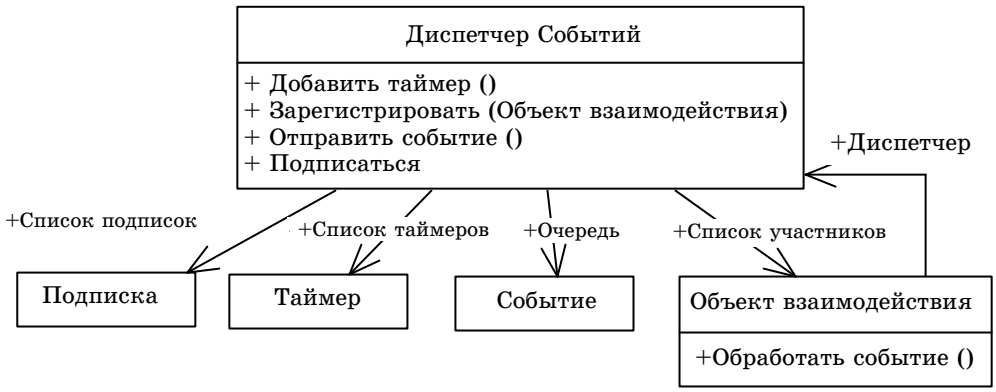

Рис. 8. Структура подсистемы межобъектного взаимодействия

В центре данной схемы находится диспетчер (брокер объектных запросов), поддерживающий очередь событий. Все объекты, желающие использовать очередь событий для обмена сообщениями, регистрируются у диспетчера. Общий принцип работы диспетчера представлен на рис. 9 в виде диаграммы отправки сообщения.

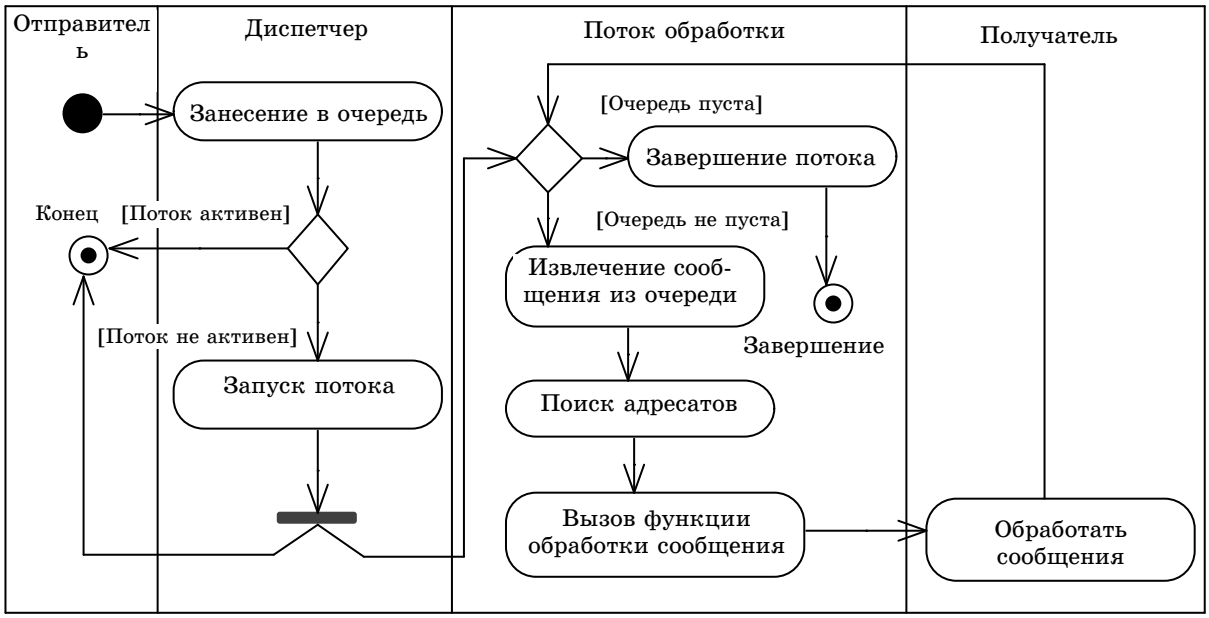

Рис. 9. Диаграмма процесса отправки сообщения

Данный механизм обеспечивает работу асинхронных сообщений. При отправке такого сообщения диспетчер подставляет его в очередь, которая обслуживается специальным потоком. Объект-получатель может запустить свой поток для обработки входящего сообщения либо воспользоваться потоком диспетчера, если не требуется длительная обработка.

Групповая рассылка сообщения осуществляется путем широковещания (через таблицу зарегистрированных объектов) или с помощью механизма подписок (через таблицу подпи-

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)

сок). Группировка получателей при использовании механизма подписок производится по регистрационному номеру объекта-издателя.

Заключение

Представленная в статье архитектура, по мнению авторов, может претендовать на роль типового решения в области автоматизации технологических и коммерческих процессов в крупных торговых и распределительных системах. Разумеется, ее вряд ли можно применить в промышленности, где количество готовых и опробованных на практике архитектур измеряется десятками. Построенная на известных типовых решениях, она настроена на решение специфичных задач большого класса систем управления сделками – области, еще недостаточно освещенной в открытых публикациях.

Описанная архитектура была опробована при создании АКТС автозаправочных станций и комплексов. Опыт эксплуатации полученного программного обеспечения продемонстрировал эффективность приведенной архитектуры и ее соответствие предъявляемым требованиям. В результате система была внедрена более чем на 100 объектах 10 городов Западной Сибири. По сравнению с имеющимися на этом рынке АКТС (например, системами IBS, «Сибинтек», «Петролеум Системс», «Мобильная карта» и др.), она показала большую гибкость (готовность к быстрой эволюции), масштабируемость (способность к изменению конфигурации при подключении новых модулей и рабочих мест) и мобильность (способность к перенастройке на использование различных СУБД, коммуникационных средств, технологических устройств).

Литература

1. Вигерс К. Разработка требований к программному обеспечению: / Пер. с англ. – М.: Изд.-торговый дом «Русская редакция», 2004. – 576 с.

2. Фаулер М. Архитектура корпоративных программных приложений: Пер. с англ. – М.: Изд. дом «Вильямс», 2004. – 544 с.

3. Гамма Э. Приемы объектно-ориентированного проектирования. Паттерны проектирования / Э. Гамма, Р. Хелм, Р. Джонсон, Дж. Влиссидес. – СПб.: Питер, 2001. – 386 с.

4. Ларман К. Применение UML и шаблонов проектирования. – М.: Изд. дом «Вильямс», 2001. – 496 с.

5. Гома Х. UML. Проектирование систем реального времени, параллельных и распределенных приложений. – М.: ДМКпресс, 2002. – 704 с.

Драганов Владимир Александрович
Мл. науч. сотр. 15-го отдела НИИАЭМ
Эл. почта: dva@snc.com.ru

Клименко Анатолий Яковлевич,
Д-р. техн. наук, зав. 15-м отделом НИИАЭМ
Эл. почта: klimenko@snc.com.ru

Фофонов Алексей Вячеславович,
Мл. науч. сотр. 15-го отдела НИИАЭМ
Тел.: (3822) 55-60-10
Эл. почта: fav1481@mail.ru

V.A. Draganov, A.J. Klimenko, A.V. Fofonov

Multilayered architecture as a basis for designing automated trading systems

Definition of the automated trading systems is given, and peculiarities in designing and implementing the systems are revealed. The main requirements to such systems properties are formulated. Based on the requirements, an example of the multilayered architecture of an automated trading system is presented.

Key words: software development, multilayered architecture, trading systems, automation, enterprise patterns.

Create PDF files without this message by purchasing novaPDF printer (http://www.novapdf.com)