В.С. БЕЛОВ, М.В. БЕЛОВ

«КЛИЕНТ-СЕРВЕРНЫЕ» ПАРАДИГМЫ В СОВРЕМЕННЫХ РАСПРЕДЕЛЕННЫХ WEB-СИСТЕМАХ

Приведено описание классических моделей организации взаимодействия в системах вида «клиент–сервер» и описано расширение этого модельного представления в переложении на условия обмена данными и информационного взаимодействия в распределенных Web-системах.

Родившаяся в 1991г. глобальная гипертекстовая распределенная система World Wide Web (WWW) в настоящее время является одной из самых популярных информационных служб сети Internet. Более половины данных, циркулирующих в глобальной сети, приходится на долю Web. Только появление Web-технологии, использующей принцип прямого визуального управления ресурсами удаленного узла «указал–нажал», позволило пользователю четко формулировать свои запросы к информационным ресурсам глобальной сети и выбирать в ней именно те сведения и данные, которые необходимы.

В основе архитектурных решений, используемых при построении современные распределенных Web-системы, лежит классическая вычислительная модель «клиент–сервер» [1], которая исходно связана с парадигмой открытых систем. Сам термин «клиент–сервер» исходно применялся к архитектуре программного обеспечения, которое описывало распределение процесса выполнения по принципу взаимодействия двух программных процессов, один из которых в этой модели назывался «клиентом», а другой — «сервером». Клиентский процесс запрашивал некоторые услуги, а серверный процесс обеспечивал их выполнение. Изначально при этом предполагалось, что один серверный процесс может обслужить множество клиентских процессов.

Первоначально приложение (пользовательская программа) не разделялась на части, оно выполнялось некоторым монолитным блоком. Однако такое решение для применения в рамках распределенной вычислительной среды оказалось нерациональным, т.к. при монолитном исполнении используются ресурсы только одного компьютера, а остальные компьютеры в сети рассматриваются как терминалы (это централизованная архитектура или архитектура Main-фреймов). Но теперь, в отличие от эпохи Main-фреймов, все компьютеры в сети обладают собственными ресурсами и разумно так распределить нагрузку на них, чтобы максимальным образом использовать их информационные и вычислительные ресурсы. Это еще более актуально в масштабах глобальных сетей, использующих Web-технологии.

Как и в промышленности, здесь были использованы приемы и методики разделения обязанностей, распределения функций между участниками процесса обмена данными. Применительно к технологии «клиент–сервер» в классическом ее понимании в рамках стандартного интерактивного приложения обычно выделяют пять групп функций [1]:

- Функции ввода и отображения данных (Presentation Logic);
- Прикладные функции и алгоритмы решения задач приложения (Business Logic);
- Функции обработки данных внутри приложения (Database Logic);
- Функции управления информационными ресурсами (Database Manager System);
- Служебные функции, играющие роль связок между функциями первых четырех групп.

В централизованной архитектуре (Host-based processing) эти части приложения располагаются в единой среде и комбинируются внутри одной исполняемой программы.

В децентрализованной архитектуре эти задачи могут быть по-разному распределены между серверным и клиентским процессами (см. рис. 1), при этом в зависимости от характера распределения выделяют следующие модели распределений [1,2]:

- Распределенная презентация (Distribution presentation, DP);
- Удаленная презентация (Remote Presentation, RP);
- Распределенная бизнес-логика (Remote business logic, RBL);
- Распределенное управление данными (Distributed data management, DDM);
- Удаленное управление данными (Remote data management, RDA);
- Управление данными с сервером приложения (Data management Application Server, DMAS).

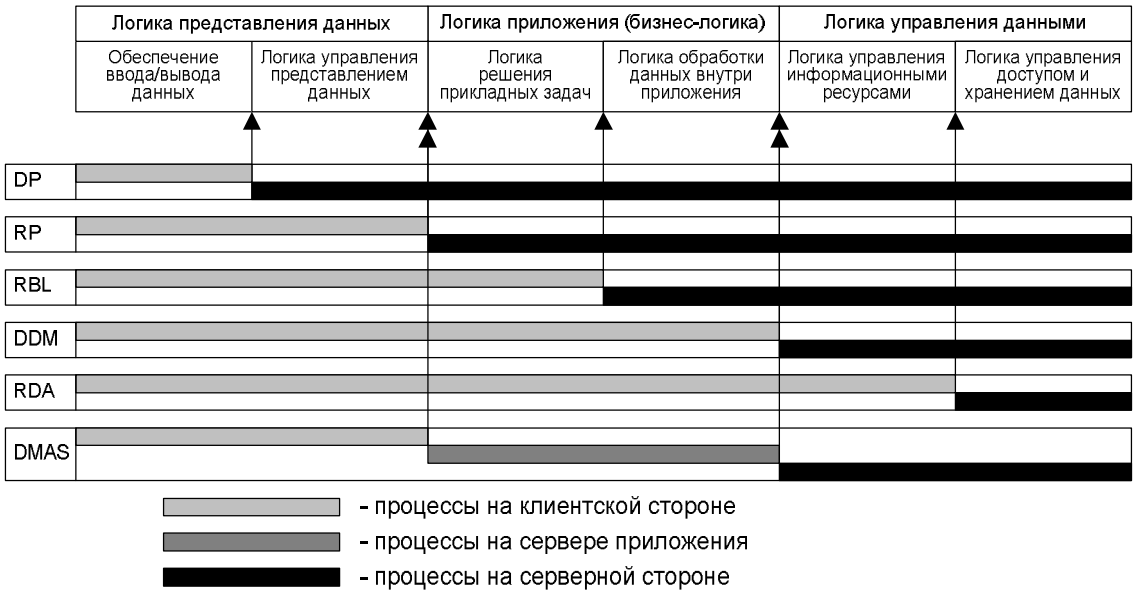

Рис.1. Распределение функций в классических архитектурах «клиент-сервер»

Такое условное представление показывает, как могут быть распределены отдельные задачи между серверными и клиентскими процессами и его в рамках в настоящее время выделяют следующие типовые модели взаимодействия (рис. 2) [1]:

- Двухуровневая модель с файловым сервером (File Server, FS);
- Двухуровневая модель удаленного доступа к данным (Remote Data Access, RDA);
- Двухуровневая модель с активным сервером баз данных (DataBase Server. DBS);
- Трехуровневая модель с сервером приложений (Application Server, AS).

Рис.2. Двух- и трех-уровневые модели «клиент-серверной» архитектуры

Однако в распределенных Web-системах состав задач по обработке и управлению данными, циркулирующими в цепях обмена информацией между клиентской станцией и серверной стороной, следует дополнить рядом новых функций, присущих только распределенным глобальным средам. В частности, к подобным новым функциям относятся [2]:

- Функции управления взаимодействием между клиентом и Web-сервером;
- Функции обеспечения делового регламента (ограничений на действия пользователя);
- Функции организации удаленного доступа к базам (хранилищам) данных.

С учетом этих и ранее упомянутых функций, необходимых для организации обмена данными между клиентским и серверными процессами, а также существующих технических решений применения Web-технологий, можно сформировать ряд схем распределения функций между участниками взаимодействия (рис. 3):

- Классическая статическая двухуровневая Web-система (сх.1):
- Двухуровневая Web-система доступа к базам данных с использованием программных шлюзов (сх.2);
- Двухуровневая Web-система с расширенной обработкой данных на клиентской стороне (сх.3);

- Двухуровневая (трехуровневая) Web-система с расширенной обработкой данных на серверной стороне (сх.4а, сх.4б);
- Двухуровневая Web-система с расширенной обработкой данных на клиентской и на серверной сторонах (сх.5);
- Трехуровневая Web-система доступа к базам данных с использованием программных шлюзов (сх.6);
- Трехуровневая Web-система интеграции серверов баз данных (приложений) (сх.7);
- Трехуровневая Web-система информационных порталов (сх.8);
- Трехуровневая Web-система межпрограммного взаимодействия (сх.9а, сх.9б).
- Четырехуровневая Web-система межпрограммного взаимодействия (сх.10).
- Четырехуровневая Web-система обработки данных (сх.11);
- Пятиуровневая Web-система обработки данных (сх.12).

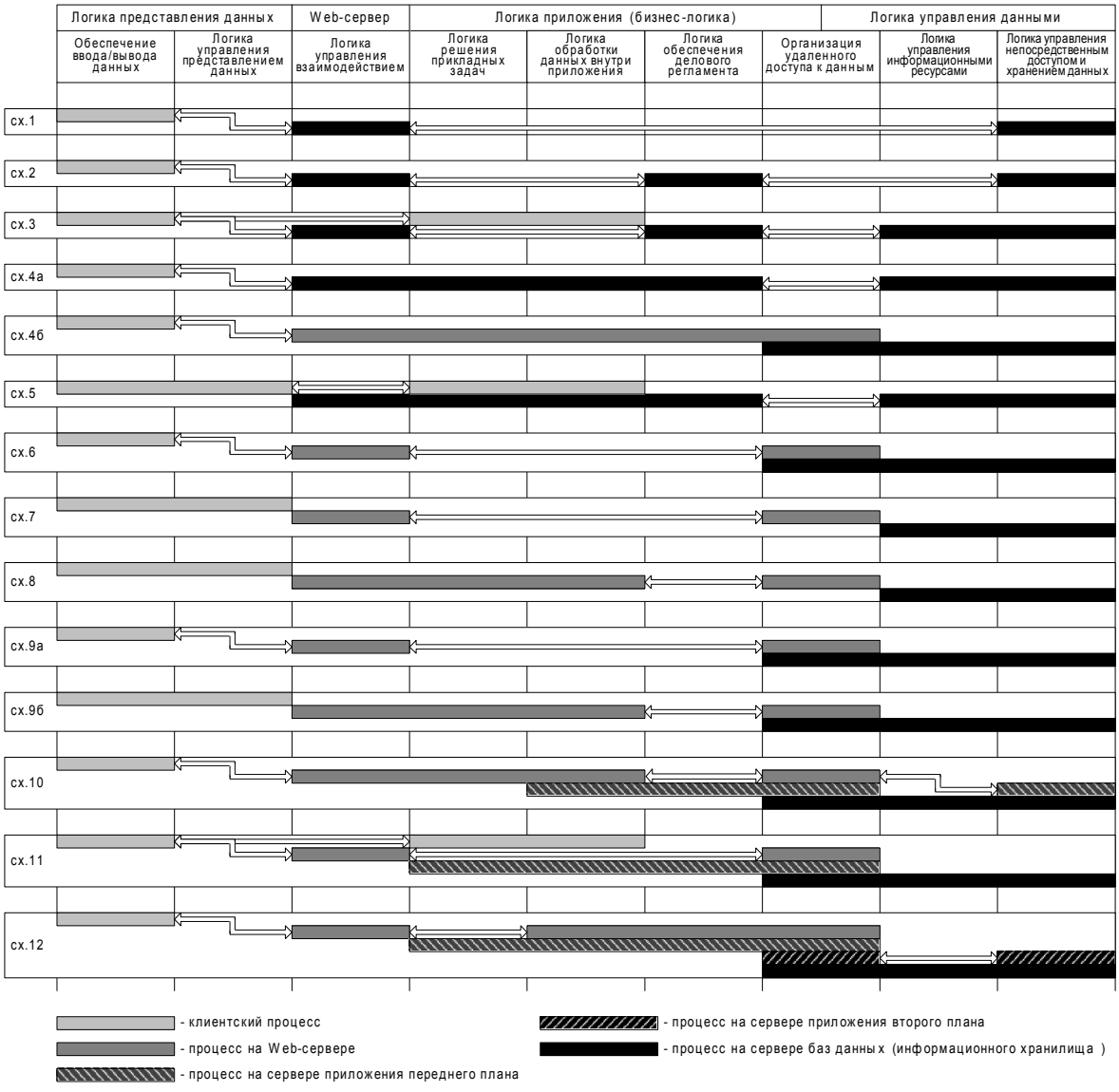

Рис.3. Модели «клиент-серверной» архитектуры в распределенных Web-системах

На основе приведенных на рис.3 моделей организации обмена данными и реализуются большинство современных Web-систем.

Так по схеме 1 реализуется классическая статическая Web-систем., работающая в режиме файлового обмена с информационным хранилищем. Она по сути своей соответствует схеме файлового сервера, отличие только в том, что состав функций, реализуемых как на клиентской, так и на серверной стороне, минимизирован. Объекты данных при представляли собой

электронные документы (файлы), созданные средствами языка гипертекстовой разметки HTML (HyperText Markup Language).

Схема 2 соответствует варианту построения Web-системы с несложной обработкой данных на серверной стороне с использованием компонентов CGI (Common Gateway Interface), FastCGI либо API (Application Programming Interface). Наиболее известными API являются: ISAPI (Internet Server API), NSAPI (Netscape Server API) и ICAPI (Internet Connection API) [3]. Такая обработка была предназначена в первую очередь для организации доступа к данным, имеющим не HTML форматы.

Схема 3 является модельным представлением Web-систем с расширенной обработкой на клиентской стороне. Одно из архитектурных решений основано на технологии мигрирующих программ [3]. Существуют три основных вида мигрирующих программ, которые могут быть связаны с Web-документами и передаваться вместе с ними с Web-сервера на рабочую станцию клиента для дальнейшего выполнения, а именно; Java-апплеты, получаемые клиентом в составе Web-страницы и исполняемые им в среде виртуального Java-процессора; Программы-скрипты, написанные на языке сценариев JavaScript, VBScript (Visual Basic Scripting), код которых внедряется в Web-страницу и интерпретируются на стороне клиента при загрузке этой страницы; Программные компоненты ActiveX Controls, представляющие собой обычные исполняемые программы, загружаемые в составе Web-страницы, как в контейнере, и выполняемые на рабочей станции в виде специальных COM-объектов, реализующих практически любую функциональность. В рамках этой же модели реализуются Web-технологии, использующие новые языки разметки Web-страницы – DHTML (Dynamic HTML), расширяющего встроенную функциональность HTML-страниц и обеспечивающего возможность повторного использования компонентов управления на клиентской стороне, и расширяемого языка XML (eXtensible Markup Language), позволяющего описать документы, содержащие структурированную информацию, и предоставляющего стандартные методы пересылки этих данных по сети Internet, причем интерпретация данных осуществляется Web-клиентом при их обработке.

Следующие модели (схемы 4а и 4б) реализованы в Web-системах с расширенной обработкой на серверной стороне [2]. Такова и есть технология ASP (Active Server Page) – технология генерации HTML-страниц на основе обработки активных серверных страниц. ASP реализует идею размещения внутри Web-страницы кода, выполняемого Web-сервером, а также технология JSP (Java Server Page), основной идеей которой является однократная компиляция Java-кода (сервлета) при первом обращении к нему, выполнение методов этого сервлета и помещение результатов выполнения этих методов в набор данных, отправляемых на клиентскую станцию.

Модель взаимодействия по схеме 5 является вариантом, объединяющим в себе достоинства схем 3 и 4, т.е. обеспечивающую расширенную обработку данных на обоих сторона – у клиента и на сервере, при этом. предусматривается активный диалог между клиентом и сервером в процессе сеанса взаимодействия. К подобным технологиям, в частности, относится AJAX. При обращении к Web-серверу, генерируется Web-страница, которая будет отображаться на клиентской станции, и предлагать клиентусовершить интересующую его последовательность действий. Выбранные коиентом действия в виде запросов передается Web-сервер, который обращается к AJAX-модулю, Этот AJAX-модуль и будет производить все интересующие вычисления и работу с сервером как таковым, возвращая результат обработки в XML-коде клиенту. Основное отличие этой модели взаимодействия в том, что имеется возможность динамически обращаться к серверным процессам и выполнять с их помощью интересующие клиента действия.

Следующие модели взаимодействия (схемы 6, 7, 8, 9а, 9б) является моделями взаимодействия, содержащими три уровня распределения функциональности по вычислительным компонентам Web-системы. Схема 6 является развитием базовой схемы 2 Web-доступа к разнородным источникам данных в плане обеспечения доступа к удаленным информационным хранилищам, при этом вся функции обработка информации сосредотачивается на третьем звене – серверам баз данных.Схема 7 соответствует модели взаимодействия, реализованной в Web-системах интеграции разнородных данных, обеспечивающих стандартизованный Web-доступ клиентам к разнородным корпоративным приложениям и базам данных. Схема 8 отражает особенности организации обмена данными и

распределение функций по подсистемам, реализованными в Web-системах реализующих технологии многомерных информационных хранилищ DW (Data Warehousing), в частности в системах оперативной аналитической обработки данных OLAP (On-Line Analytical Processing) [2]. Схемы 9а и 9б являются модельными представлениями, характеризующими распределение функциональности по уровням Web-порталов (схемы 9а) и Web-систем обработки транзакций в реальном масштабе времени типа OLTP (On-Line Transaction Processing systems).

Модели взаимодействия, реализованные по схемам 10 и 11 характеризуют четырехуровневую организацию распределенных Web-систем. В этих схемах среднее звено организуется в виде двух самостоятельных взаимодействующих серверов – Web-сервера и сервера приложения, который обеспечивает доступ к СУБД, выполняющимися на удаленном сервере. Здесь в настоящее время доминируют два стандарта [4]:

- EJB (Enterprize JavaBean), определяющий компонентную модель для разработки, развертывания и управления серверных Java-объектов многократного использования; управления транзакциями и процессами взаимодействия и базами данных различных видов (см. схему 10).
- ActiveX, который использует в качестве сервера приложения сервер автоматизации (OLE Automation) ASP (Active Server Pages), построенный на основе объектной модели программных компонентов COM (Component Object Model), и который устанавливает способ построения составных объектов и предоставления доступа к ним (см. схему 11).

Наконец, на схеме 12 приведена пятиуровневая модель взаимодействия, которая реализована, в частности, в Web-системах электронной коммерции, Web-системах электронного обмена деловыми документами EDI (Electronic Data Interexchange), Web-системах управления информационным наполнением Web-сайтов или Web-порталов CMS (Content Management Systems). Именно в этой модели выделяются сервера приложения двух видов – переднего и второго планов, что позволяет более рационально распределить функциональность среднего звена между тремя его вычислительными системами и тем самым повысить быстродействие процессов обработки разнородных данных и эффективность предоставления результатов обработки клиентам. Заметим при этом, что клиентские функции в схемах 10, 11, 12 сведены к минимуму и ограничиваются только организацией ввоза/вывода данных.

Описанные модели взаимодействия клиентских и серверных процессов далеко не полностью охватывают всевозможные технические решения в области построения Web-систем, т.к. именно область Web-технологий в настоящее время является наиболее быстроразвиваемой и прогрессирующей.

ЛИТЕРАТУРА:

1. **Карпова Т.С.** Базы данных Модели, разработка, реализация. – СПб.: Питер, 2002. – 304с.
2. **Крёнке Д.** Теория и практика построения баз данных. Пер. с англ.— СПб, Питер, 2003,. – 800с. (Серия "Классика Computer Science").
3. **Зима В.М., Молдовян А.А., Молдовян Н.А.** Безопасность глобальных сетевых технологий. — СПб.: изд. СПбУ, 1999. – 368с.
4. **Цимбал А.А., Аншина М.Л.** Технология создания распределенных систем. Для профессионалов. — СПб, Питер, 2003. – 576с.