Архитектура корпоративных информационных систем.

Опыт последних лет разработки ПО показывает, что архитектура информационной системы должна выбираться с учетом нужд бизнеса, а не личных пристрастий разработчиков. Далее рассматриваются существующие архитектуры построения информационных систем.

Не секрет, что правильная и четкая организация информационных бизнес-решений является слагающим фактором успеха любой компании. Особенно важным этот фактор является для предприятий среднего и малого бизнеса, которым необходима система, которая способна предоставить весь объем бизнес-логики для решения задач компании. В то же время, такие системы для компаний со средним и малым масштабом сетей часто попадают под критерий “цена - качество”, то есть должны обладать максимальной производительностью и надежностью при доступной цене.

Исторически первыми появились информационные системы с использованием файл-сервера. Файл-сервер только извлекает данные из файла (файлов) базы данных и передает их клиенту для дальнейшей обработки (рис. 1).

В процессе работы из базы данных клиенту передаются большие объемы информации. Значительный сетевой трафик иногда особенно сильно сказывается при одновременной работе даже уже нескольких клиентов. В файл-серверной архитектуре всегда передаются избыточные данные. Неважно, сколько записей из базы данных нужны клиенту — файлы базы данных передаются в самом общем случае целиком. Что касается MS Access, то нагрузку на сеть добавляют еще и объекты приложения, такие как формы, отчеты и т. д. Они вместе с данными хранятся в одном файле на компьютере-сервере.

Рис. 1. Структура информационной системы с файл-сервером

В MS Access 2010 у разработчика имеется возможность разделить данные и приложение, работающее с этими данными. В этом случае приложение тиражируется на компьютерах-клиентах, а база данных остается на компьютере-сервере.

Применение архитектуры «файл-сервер» привлекает своей простотой, удобством использования и доступностью. Она представляет интерес для малых рабочих групп, а нередко до сих пор используется и в информационных системах масштаба небольшого предприятия.

Клиент-серверная архитектура.

Рис. 2. Двухуровневая клиент-серверная архитектура

Двухуровневая клиент-серверная архитектура характеризуется наличием двух взаимодействующих самостоятельных модулей - автоматизированного рабочего места (АРМа) и сервера базы данных, в качестве которого может выступать Microsoft SQL Server, Oracle, Sybase и другие. Сервер БД отвечает за хранение, управление и целостность данных, а также обеспечивает возможность одновременного доступа нескольких пользователей. Клиентская часть представлена так называемым “толстым” клиентом, то есть приложением (АРМ) на котором сконцентрированы основные правила работы системы и расположен пользовательский интерфейс программы. При всей простоте построения такой архитектуры, она обладает множеством недостатков, наиболее существенные из которых - это высокие требования к сетевым ресурсам и пропускной способности сети компании, а также сложность обновления программного обеспечения из-за “размазанной” бизнес-логики между АРМом и сервером БД. Кроме того, при большом количестве АРМов возрастают требования к аппаратному обеспечению сервера БД, а это, как известно, самый дорогостоящий узел в любой информационной системе.

Как видим, минусов у такой архитектуры достаточно, а решение тривиально - нужно отделить бизнес-логику от клиентской части и СУБД, выделив ее в отдельный слой. Так и поступили разработчики и следующим шагом развития клиент-серверной архитектуры стало внедрение среднего уровня, реализующего задачи бизнес-логики и управления механизмами доступа к БД (рис. 3).

​

Трехуровневая клиент-серверная архитектура (Three-tier architecture)

Рис. 3. Трехуровневая клиент-серверная архитектура

Плюсы данной архитектуры очевидны. Благодаря концентрации бизнес-логики на сервере приложений, стало возможно подключать различные БД. Теперь, сервер базы данных освобожден от задач распараллеливания работы между различными пользователями, что существенно снижает его аппаратные требования. Также снизились требования к клиентским машинам за счет выполнения ресурсоемких операций сервером приложений и решающих теперь только задачи визуализации данных. Именно поэтому такую схему построения информационных систем часто называют архитектурой “тонкого” клиента.

Но, тем не менее, узким местом, как и в двухуровневой клиент-серверной архитектуре, остаются повышенные требования к пропускной способности сети, что в свою очередь накладывает жесткие ограничения на использование таких систем в сетях с неустойчивой связью и малой пропускной способностью (Internet, GPRS, мобильная связь).

Существует еще один важный момент использования систем, построенных на такой архитектуре. Самый верхний уровень (АРМ), в целом обладающий огромной вычислительной мощностью, на самом деле простаивает, занимаясь лишь выводом информации на экран пользователя. Так почему бы не использовать этот потенциал в работе всей системы? Рассмотрим следующую архитектуру (Рис. 4), которая позволяет решить эту задачу.

​

Распределенная архитектура системы

Еще два-три года назад реализация такой архитектуры системы для среднего и малого бизнеса была бы не возможна из-за отсутствия соответствующих недорогих аппаратных средств. Сегодня хороший ноутбук обладает мощностью, которой несколько лет назад обладал сервер крупной корпорации, и позволял рассчитывать множество важных и судьбоносных отчетов для всех сотрудников этой корпорации.

Более 95 % данных, используемых в управлении предприятием, могут быть размещены на одном персональном компьютере, обеспечив возможность его независимой работы. Поток исправлений и дополнений, создаваемый на этом компьютере, ничтожен по сравнению с объемом данных, используемых при этом. Поэтому если хранить непрерывно используемые данные на самих компьютерах, и организовать обмен между ними исправлениями и дополнениями к хранящимся данным, то суммарный передаваемый трафик резко снизиться. Это позволяет понизить требования к каналам связи между компьютерами и чаще использовать асинхронную связь, и благодаря этому создавать надежно функционирующие распределенные информационные системы, использующие для связи отдельных элементов неустойчивую связь типа Интернета, мобильную связь, коммерческие спутниковые каналы. А минимизация трафика между элементами сделает вполне доступной стоимость эксплуатации такой связи. Конечно, реализация такой системы не элементарна, и требует решения ряда проблем, одна из которых своевременная синхронизация данных.

Каждый АРМ независим, содержит только ту информацию, с которой должен работать, а актуальность данных во всей системе обеспечивается благодаря непрерывному обмену сообщениями с другими АРМами. Обмен сообщениями между АРМами может быть реализован различными способами, от отправки данных по электронной почте до передачи данных по сетям.

Еще одним из преимуществ такой схемы эксплуатации и архитектуры системы, является обеспечение возможности персональной ответственности за сохранность данных. Так как данные, доступные на конкретном рабочем месте, находятся только на этом компьютере, при использовании средств шифрования и личных аппаратных ключей исключается доступ к данным посторонних, в том числе и IT администраторов.

Такая архитектура системы также позволяет организовать распределенные вычисления между клиентскими машинами. Например, расчет какой-либо задачи, требующей больших вычислений, можно распределить между соседними АРМами благодаря тому, что они, как правило, обладают одной информацией в своих БД и, таким образом, добиться максимальной