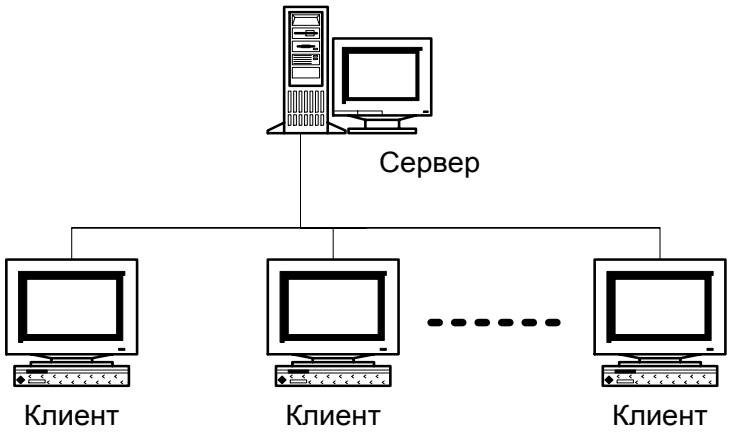
Раздел 7. Модели и методы управления в информационных системах

В последние десятилетия информационные системы строятся по сетевой технологии и на концепции баз данных. Как показывает мировой опыт, это направление останется доминирующим и в ближайшей перспективе, а изменения возможны лишь в подходах к его реализации. Наиболее передовой технологией построения баз данных является **технология «клиент-сервер»** (рис. 7.1). Клиент-сервер - это не только архитектура, это – новая парадигма, пришедшая на смену устаревшим концепциям. Суть ее заключается в том, что клиент (исполняемый модуль) запрашивает те или иные сервисы в соответствии с определенным протоколом обмена данными. При этом нет необходимости в использовании прямых путей операционной системы: клиент их «не знает», ему «известны» лишь имя источника данных и другие специальные сведения, используемые для авторизации клиента на сервере. Сервер, который физически может находиться на том же компьютере, а может – на другом конце земного шара, обрабатывает запрос клиента и, произведя соответствующие манипуляции с данными, передает клиенту запрашиваемую порцию данных.

Эволюционно сложилось несколько моделей и методов данной технологии:

* модель и метод файлового сервера (*File Server - FS*);
* модель и метод доступа к удаленным данным (*Remote Data Access - RDA*);
* модель и метод сервера базы данных (*DataBase Server - DBS*);
* модель и метод сервера приложений (*Application Server - AS*).



Архитектура "клиент-сервер"

Рис. 7.1. Архитектура технологии «клиент-сервер»

7.1. Базовая модель и метод «клиент-сервер»

Модель файлового сервера (FS-модель) является базовой для локальных сетей персональных компьютеров (рис. 7.2). Исторически – это первая архитектура информационных систем. Как исполняемые модули, так и данные размещаются в отдельных файлах операционной системы. Доступ к данным осуществляется путем указания пути (path) и использования файловых операций (открыть, считать, записать). Для хранения данных используется выделенный сервер (отдельный компьютер), который и является файловым сервером. Исполняемые модули хранятся либо на рабочих станциях, либо на файловом сервере. В последнем случае упрощается процедура их администрирования, но при этом возрастают требования к надежности сети.

Суть модели заключена в следующем. Один из компьютеров в сети считается файловым сервером и предоставляет услуги по обработке файлов другим компьютерам. Файловый сервер работает под управлением сетевой операционной системы и осуществляет доступ к информационным ресурсам (то есть к файлам). На других компьютерах в сети функционирует приложение. Протокол обмена представляет собой набор низкоуровневых вызовов, обеспечивающих приложению доступ к файловой системе на файл-сервере. FS-модель послужила фундаментом для расширения возможностей персональных систем управления базами данных (СУБД) в направлении поддержки многопользовательского режима. В таких системах на нескольких персональных компьютерах выполняется как прикладная программа, так и копия СУБД, а базы данных содержатся в разделяемых файлах, которые находятся на файловом сервере. Когда прикладная программа обращается к базе данных, СУБД направляет запрос на файловый сервер. В этом запросе указаны файлы, где находятся запрашиваемые данные. В ответ на запрос файловый сервер направляет по сети требуемый блок данных. СУБД, получив его, выполняет над данными действия, которые были декларированы в прикладной программе.

К недостаткам модели относят высокий сетевой трафик (передача множества файлов, необходимых приложению), узкий спектр операций манипуляции с данными, отсутствие адекватных средств безопасности доступа к данным (защита только на уровне файловой системы) и т.д.

7.2. Модификации модели и метода «клиент-сервер»

7.2.1. Модель и метод доступа к удаленным данным

Более технологичные **метод и модель доступа к удаленным данным** (RDA-модель) (рис. 7.2) существенно отличаются от FS-модели характером доступа к информационным ресурсам. Это обеспечивается операторами специального языка (например, SQL-Structured Query Language). Клиент направляет запросы к

информационным ресурсам (например, к базам данных) по сети удаленному компьютеру. На нем функционирует ядро СУБД, которое обрабатывает запросы, выполняя предписанные в них действия, и возвращает клиенту результат, оформленный как блок данных. При этом инициатором манипуляций с данными выступают программы, выполняющиеся на компьютерах-клиентах, в то время как ядру СУБД отводится пассивная роль – обслуживание запросов и обработка данных. Такое распределение обязанностей между клиентами и сервером базы данных не догма – сервер БД может играть более активную роль, чем та, которая предписана ему традиционной парадигмой.

RDA-модель избавляет от недостатков, присущих как системам с централизованной архитектурой, так и системам с файловым сервером. Сервер БД освобождается от несвойственных ему функций; процессор или процессоры сервера целиком загружаются операциями обработки данных, запросов и транзакций. Это становится возможным благодаря отказу от терминалов и оснащению рабочих мест компьютерами, которые обладают собственными локальными вычислительными ресурсами, полностью используемыми программами переднего плана. С другой стороны, резко уменьшается загрузка сети, так как по ней передаются от клиента к серверу не запросы на ввод-вывод (как в системах с файловым сервером), а запросы на языке SQL, их объем существенно меньше.

Основное достоинство RDA-модели – унификация интерфейса «клиент-сервер» в виде языка SQL.

7.2.2. Модель и метод сервера базы данных

Наряду с RDA-моделью все большую популярность приобретает перспективная модель и метод сервера базы данных (DBS-модель) (рис.7.2). Ее основу составляет механизм хранимых процедур – средство программирования SQL-сервера. Процедуры хранятся в словаре базы данных, разделяются между несколькими клиентами и выполняются на том же компьютере, где функционирует SQL-сервер. Язык, на котором разрабатываются хранимые процедуры, представляет собой процедурное расширение языка запросов SQL и уникален для каждой конкретной СУБД.

Достоинства DBS-модели очевидны: это возможность централизованного администрирования прикладных функций, снижение трафика (вместо SQL-запросов по сети направляются вызовы хранимых процедур), возможность разделения процедуры между несколькими приложениями, экономия ресурсов компьютера за счет использования единожды созданного плана выполнения процедуры. К недостаткам модели можно отнести ограниченность средств, используемых для написания хранимых процедур, которые представляют собой разнообразные процедурные расширения SQL, не

выдерживающие сравнения по средствам и функциональным возможностям с языками третьего поколения, такими как C, C++ или Pascal. Сфера их использования ограничена конкретной СУБД, в большинстве СУБД отсутствуют возможности отладки и тестирования разработанных хранимых процедур.

На практике часто используется смешанные модели, когда поддержка целостности базы данных и некоторые простейшие прикладные функции поддерживаются хранимыми процедурами (DBS-модель), а более сложные функции реализуются непосредственно в прикладной программе, которая выполняется на компьютере-клиенте (RDA-модель).

7.2.3. Модель и метод сервера приложений

В модели сервера приложений (AS-модели) процесс, выполняющийся на компьютере-клиенте, как обычно отвечает за интерфейс с пользователем. Прикладные функции выполняются сервером приложения. Все операции над информационными ресурсами выполняются сервером баз данных. RDA- и DBS-модели опираются на двухзвенную схему разделения функций. В AS-модели реализована трехзвенная схема разделения функций, где прикладной компонент выделен как важнейший изолированный элемент приложения (рис. 7.2).

С развитием интранет-интернет технологий появилась разновидность трехслойной архитектуры на основании использования ***web*-технологий**. В этой разновидности роль сервера приложений играет web-сервер, а в качестве клиента используется стандартный web-браузер. Достоинства – в пониженных требованиях к клиенту и в легкой встраиваемости данной архитектуры в мировые информационные сети. Основной недостаток – известные ограничения, накладываемые на интерфейс пользователя web-браузерами.

Эволюция моделей взаимодействия клиента и сервера показывает, что их совершенствование направлено на снижение трафика в сети, что повышает производительность системы. Однако реализация перечисленных моделей технологии «клиент-сервер» для построения баз данных в распределенных системах с использованием мобильных каналов связи вызывает значительные трудности. Дело в том, что в отличие от высокоскоростных и надежных локальных сетей, телефонные и особенно радиоканалы имеют намного более низкую пропускную способность при более высоких уровнях помех. Даже в современных системах «клиент-сервер», где минимизируется количество данных, передаваемых между клиентским приложением и сервером БД, применяемые протоколы взаимодействия клиента и сервера оказываются для этих каналов слишком ресурсоемкими, что в итоге приводит к значительному времени отклика при выполнении транзакций. К тому же снижается вероятность их успешного

завершения из-за возможного обрыва сеанса. Для примера, только время установления сеанса с сервером может составлять от одной до нескольких минут.

Для разрешения проблемы большого времени отклика, возникающей при работе мобильных пользователей в режиме клиент-сервер по медленным каналам связи, используются мобильные агенты (рис. 7.3). С целью уменьшения обмена по низкоскоростным каналам уже известная модель «клиент-сервер» трансформирована в новую - «клиент-агент-сервер».

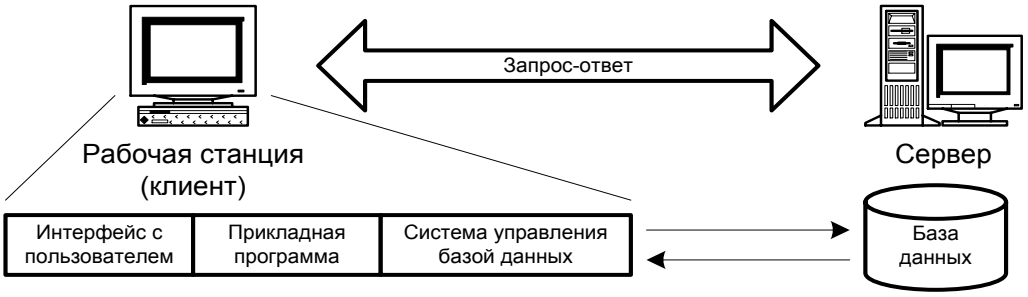
7.3. Программы-агенты и модель «клиент-агент-сервер»

Правильная ориентация в современной компьютерной сети становится чрезвычайно трудной. Решение этой задачи видится учеными и инженерами на пути использования технологии агентов. Предполагается, что по мере развития технологии агентов на узлах сети будет автоматически появляться подобранная с учетом индивидуальных потребностей пользователя информация. Процесс индивидуализации будет происходить с использованием агентов незаметно для пользователя. Лучшие из агентов смогут самостоятельно обучаться, подражая примеру пользователя. В них будет заложена способность отслеживать последовательность действий, выполняемых во время сеанса просмотра, и накапливать информацию об интересующих вопросах, внося соответствующие коррективы в свое поведение.

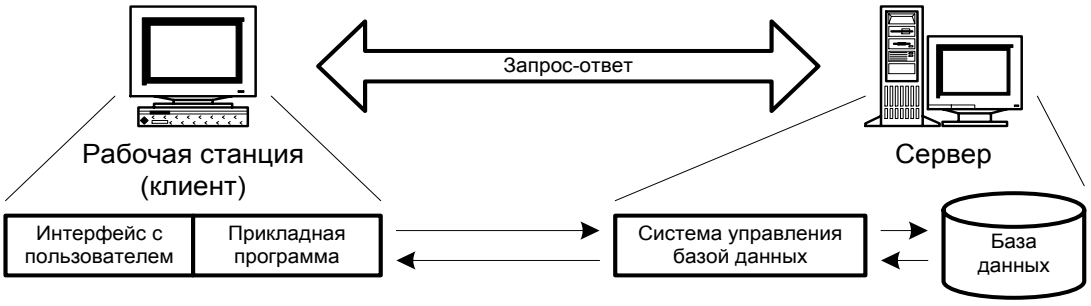
Агенты разделяются на две группы, исходя из того, где они находятся и функционируют. Стационарные агенты работают в основном на стороне клиента или на стороне сервера. Эти программы функционируют во взаимодействии с браузером - программой просмотра сети и автоматизируют сеансы просмотра. Мобильные агенты относятся к более совершенной и многообещающей категории программных продуктов. Такие агенты способны самостоятельно перемещаться от сервера к серверу в поисках нужной информации. Это выполняющиеся программы, несущие в себе информацию о собственном состоянии, т. е. вполне автономные. Сегодня концепция программ, извлекающих ресурсы из удаленных точек сети, представляется естественной, однако идея программ, перемещающихся от сервера к серверу, отличается новизной. Особенность мобильных агентов – в их автономности. Перемещаясь по сети, они несут информацию о своем состоянии (все, что необходимо для их функционирования). Обнаружив необходимые данные, они могут послать сообщение исходному клиенту или серверу. Безусловно, в таком случае важной задачей становится обеспечение информационной безопасности. Разработчики агентов всячески стараются исключить возможность превращения их в информационное оружие.

В модели «клиент-агент-сервер» работы каждое приложение разбивается на две взаимодействующие части. Первая - «клиент», находится на мобильном ПК, обеспечивая пользовательский интерфейс для ввода данных и представления результатов и, возможно, некоторую локальную обработку. Вторая - «агент», располагается на компьютере в локальной сети и, выступая там представителем первой, выполняет по ее запросам доступ к серверам БД, другим источникам данных, различного рода сервису типа электронной почты, печати, передачи факсов и т.п. В отличие от

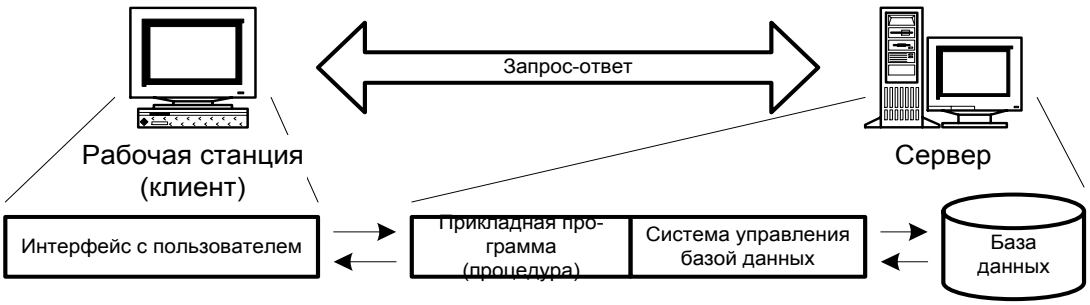
клиент-серверной модели «клиент-сервер» здесь не организуется сеанс работы пользователя с БД. Вместо этого «клиент» посылает своему «агенту» короткие сообщения-запросы. Получив сообщение, «агент» работает самостоятельно и выполняет запрошенные действия, по окончании которых возвращает «клиенту» сообщение-ответ, содержащее требуемые данные или же просто информацию об успешном завершении операции. В новой модели работа мобильных клиентов по низкоскоростным линиям сводится к минимуму с переносом основного трафика в компьютерную сеть.



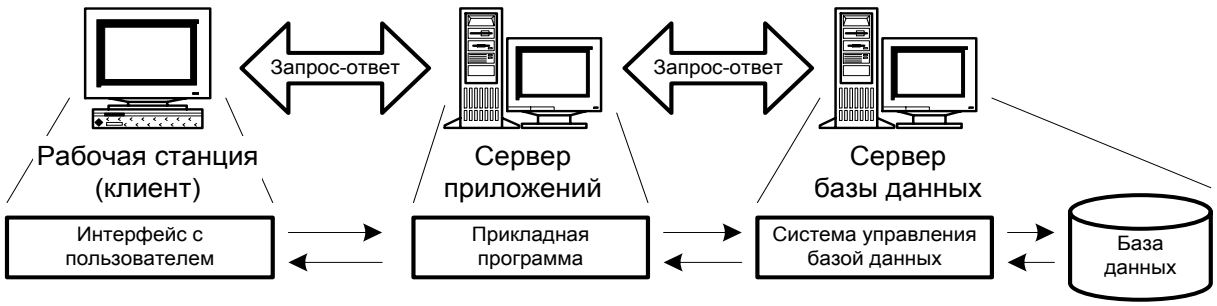
Модель "файл-сервер"



Модель удаленного доступа

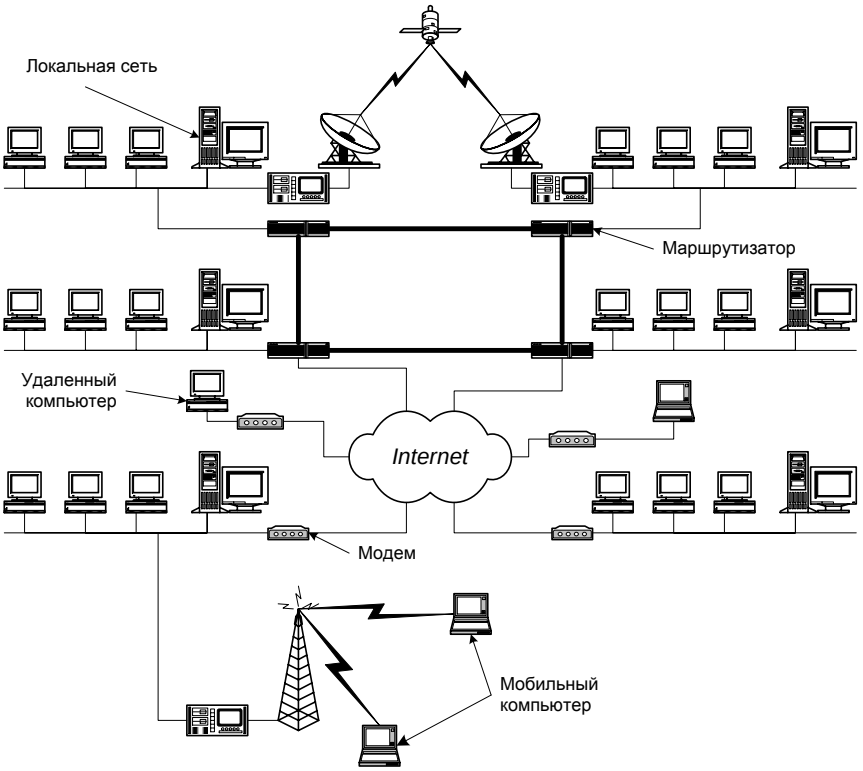


Модель сервера базы данных



Модель сервера приложений

Рис. 7.2. Модели доступа в современных информационных системах



Фрагмент глобальной компьютерной сети

Рис. 7.3. Примеры доступа компьютеров в глобальную сеть

7.4. Особенности управления в распределенных информационных системах

Объединение компьютеров в единые вычислительные системы на основе технологии компьютерных сетей поставило перед разработчиками информационных систем новые задачи и предопределило необходимость реализации принципиально новых подходов к организации информационного процесса в компьютерных сетях. Качественный скачок в этой области возможен благодаря созданию гибких, высокопроизводительных и экономичных *распределенных информационных систем*.

Распространение концепции баз данных (БД) на новый уровень позволяет определить *распределенную информационную систему* как комплекс логически интегрированных и территориально рассредоточенных БД, технических, программных, языковых и организационных средств, предназначенных для накопления, ведения и использования информации. В свою очередь, *распределенная база данных* (РБД) определяется как интегрированная БД, физически размещаемая на нескольких территориально распределенных компьютерах сети.

Для таких систем характерными являются следующие функции:

накопление, обновление и хранение данных в географически удаленных узлах сети;

логическая интеграция территориально распределенных данных, процессов обработки, обновления и поиска информации;

обеспечение автоматического взаимодействия между локальными базами данных в процессе исполнения запросов и решения задач пользователей.

Управление выполнением основных функций ИС осуществляет комплекс программных средств, который включает *систему управления распределенными базами данных* (СУРБД) в качестве основного компонента, а также сетевую операционную систему, систему разграничения доступа и др.

Под СУРБД понимают систему, реализующую принцип логической интеграции данных, физически распределенных между различными взаимосвязанными компьютерами. К основным функциям СУРБД можно отнести анализ и распределение запросов, а также управление их прохождением.

Следует отметить, что реализация концепции распределенных БД обеспечивает независимость программ по отношению к распределению данных в вычислительной сети. Это означает, что пользователь может иметь доступ к любым данным сети в пределах своих полномочий как к собственной локальной базе данных. В этом контексте существенный интерес представляет рассмотрение различных способов распределения данных по узлам сети для обеспечения эффективного доступа к ним.

Стратегии распределения данных по узлам компьютерной сети могут классифицироваться по различным признакам. К основным из них можно отнести наличие дублирования информации и количество узлов, содержащих копии данных. В соответствии с указанными классификационными признаками представляется возможным выделить четыре альтернативных стратегии распределения:

централизация (единственная копия БД, расположенная в одном узле);

расчленение (единственная копия БД, непересекающиеся фрагменты которой распределены по нескольким узлам);

дублирование (несколько копий БД, в каждом узле располагается полная копия всей базы);

смешанная (несколько копий БД, в каждом узле располагается произвольный фрагмент базы).

Ранжирование стратегий по предпочтительности их использования в общем случае не представляется возможным. Определение ситуаций, в которых та или иная стратегия является наиболее подходящей, целесообразно после выявления преимуществ и недостатков каждой из них.

Стратегия централизации. Названная стратегия характеризует предельный случай распределения данных. Строго говоря, БД, построенная по этой стратегии, не является распределенной. Тем не менее системы, реализующие стратегию централизации, могут быть вынесены в класс систем распределенной обработки, в которых рассредоточены вычислительные ресурсы и программное обеспечение.

К несомненным достоинствам централизованной БД следует отнести ее простоту, так как все операции обращения к данным выполняются под управлением одного узла.

В то же время недостатки рассматриваемой стратегии связаны именно с локальным расположением данных. Естественно, что объем базы данных ограничен емкостью запоминающих устройств центрального узла. К тому же запросы на операции над данными должны направляться только в этот узел со всеми вытекающими транспортными издержками. Центральный узел становится узким местом всей сети по следующим причинам: Во-первых, скорость обработки данных ограничивается его быстродействием; во-вторых, БД становится недоступной для других узлов при появлении неисправностей в сети передачи данных и полностью отказывает при выходе из строя центрального узла.

Любая из трех других стратегий помогает преодолеть указанные недостатки ценой определенных затрат.

Стратегия расчленения. При использовании этой стратегии база разделяется на непересекающиеся подмножества данных, называемые *логическими фрагментами*. Каждый такой фрагмент

размещается в отдельном узле. Число узлов, хранящих данные, может быть произвольным, в предельном случае равным общему числу узлов сети.

Преимущества расчленения данных перед централизованным размещением следующие. Размер базы ограничивается объемом запоминающих устройств всей сети (ее части). Снижается чувствительность к узким местам в сети передачи данных, поскольку нагрузка на нее распределяется более равномерно. Повышается доступность данных. Даже если сеть передачи данных полностью или частично выходит из строя, то доступными для пользователей могут оставаться фрагменты БД в отдельных узлах или узлах, остающихся связанными. Это обеспечивает реализацию функций информационной системы, хотя и в сокращенном объеме.

Важную роль при определении потенциальной доступности данных в критических ситуациях играет степень локализации ссылок. Под *локализацией ссылок* понимают характер расположения данных относительно запросов пользователей. Наивысшая степень локализации ссылок существует в случае, если данные, расположенные в одном узле, запрашиваются исключительно пользователями этого узла. Напротив, степень локализации ссылок мала, если подобное расчленение базы невозможно. Таким образом, если запрос пользователя может быть исполнен с помощью локально хранимых данных (степень локализации ссылок высока), то неисправности в других узлах или в сети передачи данных не окажут влияния на результат. Если же степень локализации ссылок незначительна или запросы являются сложными, то в большинстве случаев пользователям могут потребоваться данные от других узлов. Недоступность хотя бы одного узла приведет к неисполнению запроса пользователя. В подобной ситуации доступность данных в расчлененной базе может быть хуже, чем в централизованной, так как вероятность того, что по меньшей мере один узел сети будет недоступен, выше вероятности недоступности единственного конкретного узла. Следовательно, стратегия расчленения обеспечит меньшую по времени доступность БД, чем стратегия централизации.

Степень локализации ссылок влияет и на временные характеристики расчлененной БД. Так, если большая часть запросов обслуживается локально, уменьшаются затраты времени на передачу данных. С другой стороны, если исполнение запроса требует доступа ко многим узлам, среднее время задержки может превышать аналогичный показатель централизованной базы. Тем не менее, уменьшение времени реакции может быть достигнуто, если в полной мере используется параллелизм в вычислительной сети.

В целом стратегия расчленения наиболее приемлема в случаях, когда либо объем внешней памяти в каждом узле существенно меньше размера БД, либо недостаточна надежность центрального узла или

сети передачи данных. Также использование рассматриваемой стратегии рекомендуется, если запросы обладают высокой степенью локализации ссылок. В противном случае эффективность функционирования базы может оказаться слишком низкой вследствие больших затрат на передачу данных.

Стратегия дублирования. Реализация этой стратегии предполагает размещение полной копии БД в нескольких узлах сети (в предельном случае в каждом узле). Отсутствует задача определения, какую часть базы должен содержать тот или иной узел. Однако первоочередное значение приобретают вопросы, связанные с согласованием многочисленных копий данных.

Рассматриваемая стратегия обеспечивает наивысший уровень доступности и надежности данных, но при явных затратах внешней памяти сети. Объем базы данных ограничивается наименьшей емкостью запоминающих устройств в узлах, хранящих данные. Обработка данных большей частью проводится локально, но согласование множества копий требует синхронизации. Способы согласования в конкретных приложениях могут быть различными, а уровень транспортных издержек существенно зависит от степени постоянства данных.

Необходимость согласования копий и сложность управления этим процессом являются причиной того, что стратегия дублирования позволяет реализовать параллельную обработку не столь просто, как стратегия расчленения. В свою очередь, каждый узел может работать асинхронно, а время реакции на запросы может быть небольшим, особенно при исполнении запросов, не требующих согласования копий, например при выполнении поисковых процедур. Налицо простота замены разрушенной копии или восстановления процесса обработки в случае выхода из строя узла. Согласованная копия может быть получена из любого рабочего узла, после чего обращение к восстановленному узлу может проводиться в полном объеме. Следует заметить, что если часть сети недоступна по какой-либо причине, необходимым становится наложение ограничений на выполнение операций обновления данных для поддержания глобальной согласованности копий базы данных. Действительно, если разрешить две операции обновления в двух узлах при отсутствии согласования, то при возобновлении нормального функционирования сети возможно нарушение согласованности базы данных.

Таким образом, стратегия дублирования наиболее применима в ситуациях, когда объем данных невелик, фактор доступности и надежности обращения к данным играет значительную роль, а интенсивность обновления данных невысока. Последнее обстоятельство наиболее характерно для информационно-поисковых систем.

Смешанная стратегия. Эта стратегия распределения данных объединяет подходы, используемые при расчленении и дублировании,

с целью приобретения преимуществ последних. Но в то же время смешанная стратегия не лишена недостатков, присущих своим прототипам.

Использование смешанной стратегии предполагает деление БД на логические фрагменты (расчленение) и наличие в сети произвольного числа их физических копий, называемых *хранимыми фрагментами* (дублирование). Общность стратегии состоит в том, что любая часть базы может быть дублирована произвольное число раз, при этом в каждом узле может храниться желаемое подмножество данных.

Основным преимуществом смешанной стратегии, несомненно, является гибкость. Это важное свойство позволяет достичь компромисса между объемом памяти, используемой в целом и в каждом узле, с одной стороны, и обеспечиваемым уровнем надежности и оперативности, с другой. Например, архивные данные могут храниться в одном узле, а часто используемые могут быть дублированы. При дублировании логического фрагмента усложняются процедуры обновления хранимых фрагментов, но значительно большее количество данных становится локально доступным, то есть повышается степень локализации ссылок. Последнее ведет к снижению транспортных издержек за счет уменьшения числа пересылок. Стратегия допускает довольно простую организацию параллельной обработки, что ведет к уменьшению времени реакции на запросы. Появляется возможность обхода узких мест в сети передачи данных.

Недостатком смешанной стратегии является необходимость хранения информации о местонахождении данных в сети. Нетрудно заметить, что без наличия такой справочной информации невозможна реализация стратегии расчленения. Однако в последнем случае всегда можно указать однозначное соответствие между логическим фрагментом, с одной стороны, и узлом, в котором расположена его физическая копия, с другой.

В свою очередь, смешанная стратегия характеризуется многозначностью подобного соответствия, что существенно увеличивает объем справочных данных и требует введения процедур оптимизации доступа к хранимым фрагментам. Сложности возникают также при согласовании произвольного числа хранимых фрагментов, связанных с каждым логическим фрагментом. Обработка и оптимизация запросов являются нетривиальными задачами.

Рассматриваемая стратегия может быть рекомендована к реализации тогда, когда ни одна из более простых стратегий не признается удовлетворительной. Например, необходимо обеспечить выполнение высоких требований по надежности, предъявляемых к отдельным частям большой по объему БД. При этом каждый узел может обращаться к некоторым частям базы довольно часто, а к

остальным – значительно реже. В этом случае стратегия расчленения не может обеспечить достаточной надежности, а стратегия дублирования может быть неприемлемой из-за требований на объем памяти в узлах.

Подводя итог рассмотрению стратегий, следует заметить, что первоочередные проблемы, связанные с распределением данных в сети, подлежат разрешению на стадии *проектирования РБД*, которая имеет ряд существенных особенностей по сравнению с проектированием локальных БД.

В распределенной информационной системе логически целостная БД может быть фрагментирована и широко распределена по сети с целью улучшения производительности и надежности ИС. Фрагментация и распределение данных без централизованного планирования часто являются причиной несогласованного использования данных. Поэтому последовательность этапов проектирования РБД должна учитывать этот факт.

Приведем краткую характеристику этапов проектирования РБД.

Этап анализа предметной области. Его реализация предполагает изучение и описание предметной области, а также анализ пользовательских потребностей в информации.

Этап концептуального проектирования. По результатам предыдущего этапа разрабатывается инфологическая модель (информационная структура) предметной области.

Этап логического проектирования (проектирования реализации). Во время его проведения осуществляется выбор системы управления РБД и наложение ограничений выбранной СУРБД на информационную структуру. Результатом логического проектирования выступает глобальная структура БД.

Этап расчленения базы данных. Рассматриваемый этап связан с делением глобальной БД на логические фрагменты. При решении задачи расчленения учитывают требования к обработке данных, характеристики выбранной СУРБД, а также характеристики технических и программных средств в узлах сети. Результатами являются совокупность логических фрагментов и размер каждого из них.

Этап размещения базы данных. На этом этапе решается задача выбора узлов сети для размещения в них хранимых фрагментов, соответствующих логическим фрагментам БД. Определенные ограничения накладывают требования по обработке данных и разграничению доступа, особенности сети передачи данных (ее топология, пропускная способность каналов), а также характеристики аппаратуры и программного обеспечения узлов вычислительной сети. Решение представляет собой перечень узлов, с каждым из которых связан список фрагментов БД.

Этап проектирования локальных баз данных. Является заключительным в рассматриваемой последовательности. На нем осуществляется проектирование физических структур локальных БД, образованных в результате выполнения всех процедур предшествующих шагов.

Главное различие процессов проектирования РБД и локальных БД состоит в наличии этапов расчленения и размещения БД, которые поэтому заслуживают более подробного рассмотрения.

При *расчленении* исходная глобальная база разделяется на множество логических фрагментов, называемых также разделами. Естественно, что должно выполняться требование о сохранении информации, то есть разделы должны содержать все сведения, имеющиеся в исходной глобальной базе. Дополнительно на процесс формирования разделов накладываются ограничения по их допустимому размеру, времени реакции на запрос и надежности обращения. Вследствие этого в раздел рекомендуется объединять такие часто используемые совместные записи, чтобы он улучшал характеристики времени ответа на запрос. Также следует стремиться к получению требуемого уровня надежности, используя по возможности меньшую кратность дублирования, то есть степень локализации ссылок должна быть высокой при минимальном числе копий хранимых фрагментов. Допустимый размер каждого раздела как неделимой совокупности данных определяется фиксированным объемом памяти в каждом узле сети. И в общем случае ограничения на класс допустимых расчленений накладывает емкость внешних запоминающих устройств в узлах сети.

Задача *размещения* распределенной базы данных решается сравнительно просто для двух стратегий: централизации и дублирования. Вполне очевидно, что отпадает необходимость в выполнении процедуры расчленения базы данных. Если принята первая стратегия, то перед разработчиком стоит один вопрос: в каком узле следует разместить базу данных? Реализация второй стратегии для каждого узла сети требует решения вопроса: размещать или не размещать полную копию базы? Ответы на поставленные вопросы зачастую предопределены и зависят от структуры сети, объема памяти в ее узлах, перечня альтернатив и здравого смысла.

Значительно сложнее представляется задача размещения при использовании стратегии расчленения, особенно сложной при смешанной стратегии. В случае реализации стратегии расчленения необходимо: во-первых, расчленить базу на логические фрагменты и, во-вторых, разместить каждый фрагмент в конкретном узле с учетом ограничений на размещение. Задача является итеративной, и возможно, что расчленение базы данных потребуется проводить неоднократно. Если же используется смешанная стратегия, решение становится более сложным: каждый логический фрагмент может быть

размещен в любом числе узлов. Количество перестановок фрагментов растет очень быстро, и это является одной из причин того, что ограничиваются нахождением не оптимального, а рационального размещения.

Решение задачи оптимального размещения фрагментов по узлам сети методами линейного программирования возможно при довольно жестких допущениях о характере потока запросов к базе, предопределенном числе и неизменности этих запросов, заранее известном числе хранимых фрагментов. Количество переменных и ограничений прогрессирует с увеличением числа узлов сети, и поэтому получение решения возможно лишь для задач малой размерности. Решение также усложняется при предъявлении менее жестких требований к характеру запросов пользователей. Подходы к решению используют в своей основе метод динамического программирования. Если же типовые запросы первоначально неизвестны, то используются статистические методы для определения этих запросов, а результаты их определения служат входными данными для решения задачи размещения.

В целом процесс разработки распределенных баз данных отличается высокой трудоемкостью и значительными материальными затратами. Задача выбора наилучшего соответствия между характеристиками РБД и методами распределения данных в сети требует всестороннего анализа, так как принятые проектные решения оказывают непосредственное влияние на реализацию и последующее функционирование информационной системы.

Очевидно, что распределенные БД, в отличие от локальных баз, должны потребовать большего числа уровней представления данных для реализации принципа независимости описания структуры баз от данных. Действительно, трехуровневая архитектура в случае РБД расширяется до пяти уровней (рис. 7.4).

Пользовательский уровень представления данных. Служит для описания части базы данных, доступной одному конкретному пользователю или группе пользователей, рассматриваемых как один. Эта часть многоуровневого представления является, по сути, внешней моделью в концепции *ANSI*. Каждый пользователь может иметь отличное от других пользователей представление, соответствующее его требованиям и требованиям разграничения доступа.

Глобальный логический уровень представления данных. Этот уровень подобен концептуальному уровню представления концепции *ANSI*. Используется для описания логической структуры всей распределенной базы данных, то есть в представлении администратора РБД. При описании РБД этому уровню ставится в соответствие глобальная представляющая схема.

Существование третьего и четвертого уровней объясняется распределенной природой РБД.

Фрагментный уровень представления данных. Используя этот уровень, администратор РБД определяет несвязанные подмножества базы данных, то есть логические фрагменты, и описывает их средствами СУБД в виде локальных представляющих схем.

Уровень представления распределения данных. На данном уровне определяется географическое расположение экземпляров каждого логического фрагмента. Уровень допускает существование нескольких физических фрагментов, соответствующих логическому фрагменту. Этому уровню соответствует схема распределения данных.

Уровень локального представления данных. Соответствует описанию той части базы данных, которая существует в конкретном узле. Несомненно, что эта локальная база может рассматриваться с точки зрения как логической, так и физической структуры. Однако локальное представление считается описанием логической структуры, при этом физическая структура является скрытой от администратора РБД. Локальная БД как база в полном понимании этого слова имеет несколько уровней представления, но в данном рассмотрении эти уровни не принимают участия.

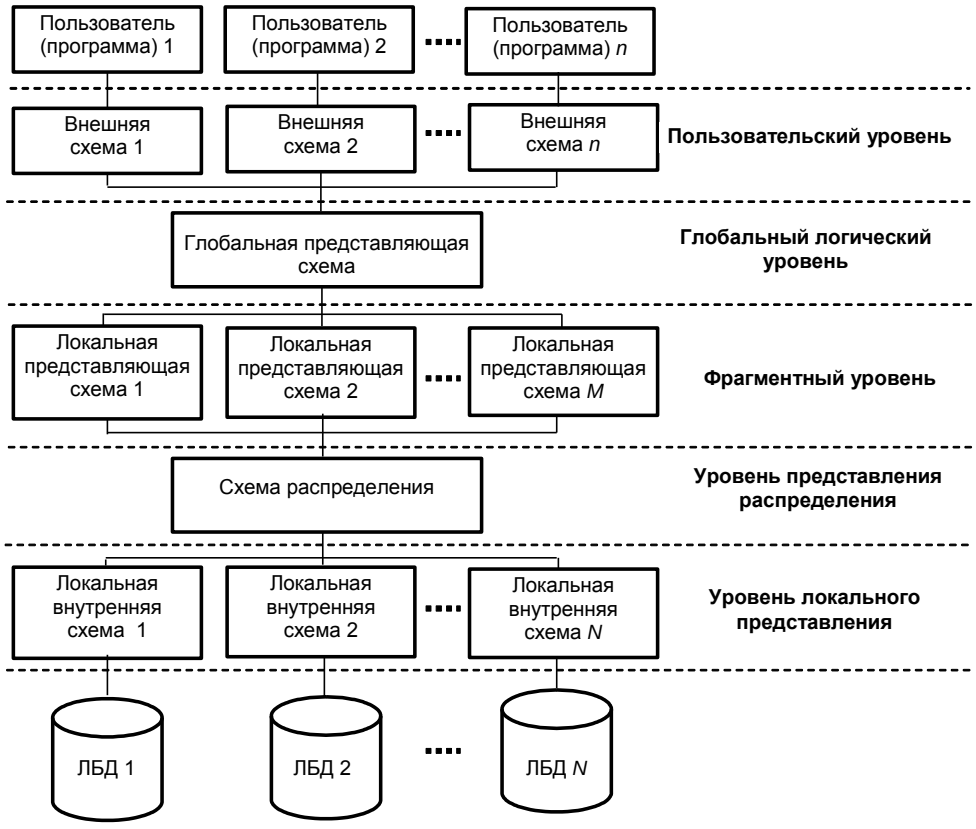


Рис. 7.4. Уровни представления данных в РБД

При обращении множества пользователей или прикладных программ к распределенной базе данных на первый план выдвигается *проблема управления параллельным выполнением запросов к СУРБД*. Обязательным условием является то, что каждый запрос, связанный с корректировкой данных, должен оставлять РБД в непротиворечивом

состоянии. В таком случае по окончании исполнения запроса необходимо установить, что СУРБД завершила выполнение всех подзапросов, изменяющих данные. При подобном подходе исключается возможность использования информации, определяемой переходным состоянием базы данных. Данное условие может выполняться по-разному в зависимости от времени проведения операций обновления данных. Оперативная актуализация проводится в реальном масштабе времени, а периодическая – в некоторые установленные моменты после накопления изменений данных. В последнем случае выполнение запросов на выборку данных при обновлении блокируется. Реализация периодической актуализации представляется довольно простой, однако данный подход не может быть признан удовлетворительным для использования в базах данных, являющихся информационными моделями систем, состояние которых подвержено частым изменениям.

При оперативной актуализации действия над данными, проводимые в соответствии с различными запросами, могут выполняться СУРБД параллельно. При этом возникает несколько ситуаций, когда операция над данными может быть не завершена. Это тупиковые ситуации и циклические рестарты. Тупиковые ситуации характеризуются тем, что операции в нескольких запросах (транзакциях) вынуждены ждать завершения друг друга. Явление циклического рестарта связано с периодическим попаданием транзакции в такое состояние, когда ее выполнение становится невозможным, после чего она отменяется, а затем производится повторный запуск. Механизм управления параллельным выполнением транзакций должен либо исключать подобные ситуации, либо обеспечивать их разрешение. К основным методам, позволяющим корректно завершать выполнение оперативной актуализации распределенной базы данных и оставлять ее в непротиворечивом состоянии, относят методы: блокировок данных, согласия большинства и предварительного анализа конфликта транзакций.

Метод блокировок данных. Данный метод является наиболее распространенным. На практике применяются две его модификации, различающиеся выбранным способом управления. При *централизованной блокировке* в сети выделяется главный распределитель ресурсов, которому направляются все требования транзакций по блокировке изменяемых данных. С получением права пользования ресурсом транзакция актуализации осуществляет доступ к данным в любом узле, хранящем копию. Для обеспечения гарантированного использования непротиворечивых данных транзакции, осуществляющие только операции чтения, должны придерживаться принятой дисциплины доступа к данным. Реализация *децентрализованной блокировки* устраняет главный недостаток предыдущей модификации – низкую степень параллелизма и малую

живучесть системы управления БД, но при этом не исключается возможность образования тупиков.

Метод согласия большинства. При использовании названного метода узлам предоставлено право решать вопрос об изменении данных по конкретному запросу «голосованием». Это позволит разрешить конфликтные ситуации, возникающие при обращении к данным, и полностью исключить тупиковые ситуации. Степень параллельного выполнения транзакций та же, что и при децентрализованной блокировке данных.

Сущность метода согласия большинства состоит в следующем. С каждым компонентом базы данных связывается временная метка, фиксирующая момент его последнего изменения. Операторы изменения данных принимаются к исполнению только в том случае, если они относятся к транзакциям, выполнение которых началось позднее времени, указанного в метке. В операторах изменения указываются: уникальный код транзакции, ее временная метка, список изменяемых ресурсов, новые значения данных, список базовых ресурсов (используемых при формировании изменения), значения их временных меток и т.д. При получении оператора узел обязан выполнить одно из следующих действий:

проголосовать *отрицательно* (отклонить оператор) в случае, если нарушено условие соотношения временных меток, указанное ранее, и временные метки базовых ресурсов, сообщенные в операторе, относятся к более ранним моментам, чем фактически зафиксированные в узле;

проголосовать *положительно* (принять оператор) при условии, что все соотношения временных меток являются удовлетворительными и данный оператор не вступает в конфликт с другим активным оператором. (Конфликт возможен, если активный оператор, то есть не принятый и не отклоненный, пытается изменить базовый ресурс рассматриваемого оператора или модифицируемый рассматриваемым оператором ресурс является базовым для любого активного оператора);

воздержаться от голосования, когда соотношение временных меток является удовлетворительным, но имеет место конфликт с другими активными операторами.

Метод предварительного анализа конфликта транзакций. Этот метод предполагает анализ возможности возникновения конфликта и его характера. Каждому виду конфликта ставится в соответствие специализированный протокол синхронизации транзакций, эффективно разрешающий возникшую ситуацию. В основе работы протокола находится техника временных меток. Предварительный анализ ситуации, проводимый центральным узлом, исключает образование тупиковых ситуаций.

Выбор того или иного метода для его последующей реализации зависит от конкретных приложений. Выработка общих рекомендаций является затруднительной вследствие отсутствия сведений о достигаемых значениях показателей эффективности функционирования различных СУРБД.

Выводы

1. В распределенных информационных системах задачи управления являются комплексными. Одной из задач управления является управление доступом к информационным ресурсам. Наиболее передовой технологией такого управления является технология «клиент-сервер». Суть ее заключается в том, что сервер обрабатывает запрос клиента и, произведя соответствующие манипуляции с данными, передает клиенту запрашиваемую порцию данных. Базовой моделью технологии «клиент-сервер» является модель файлового сервера (FS-модель).
2. Более технологичной является модель доступа к удаленным данным (RDA-модель), которая существенно отличаются от FS-модели характером доступа к информационным ресурсам.
3. Наряду с RDA-моделью все большую популярность приобретает перспективная модель и метод сервера базы данных (DBS-модель). Ее основу составляет механизм хранимых процедур. Хранимые процедуры представляет собой процедурное расширение языка запросов SQL и уникальны для каждой конкретной СУБД.
4. Управление выполнением основных функций распределенной информационной системы осуществляет комплекс программных средств, который включает систему управления распределенными базами данных (СУРБД) в качестве основного компонента, а также сетевую операционную систему, систему разграничения доступа и др. К основным функциям СУРБД можно отнести анализ и распределение запросов, а также управление их прохождением.
5. Стратегии распределения данных по узлам компьютерной сети могут классифицироваться по различным признакам. К основным из них можно отнести четыре альтернативных стратегии распределения: централизация; расчленение; дублирование и смешанная стратегия.

Литература

1. Новые информационные технологии в науке и образовании: Уч. пособие / Под ред. И.Б. Саенко – СПб.: ВАС, 2007.