

**Jet**

**INFO**

**МАТЕРИАЛ  
МАТЕРА  
НОМЕРА**

**Архитектура сервера  
INFORMIX-OnLine  
Dynamic Server 7.1  
и коммуникационные  
средства**

**А ТАКЖЕ:**

- **НОВОСТИ INTERNET**
- **ЧТО ЕЩЕ ПОЧИТАТЬ**
- **ПАРТНЕРЫ JET INFOSYSTEMS**

**2**  
**1995**

Ведущий рубрики — Александр Гагин.



### OSF/1 — МАВР?

Операционная система OSF/1 закончила свое существование. На встрече членов Open Software Foundation в Бостоне президент Дэвид Тори официаль-

но объявил, что OSF/1 более развиваться не будет, а OSF сосредоточит свои усилия на разработке программного обеспечения для микроядра Mach МК6 и МК7.

### СПЕЦИФИКАЦИИ 100Base-T СТАЛИ СТАНДАРТОМ IEEE

Представители Альянса быстрого Ethernet объявили 14 июня о завершении процедуры принятия спецификаций быстрого Ethernet в качестве официального стандарта IEEE. Напомним, что

предшественник 100Base-T, стандарт IEEE 802.3, является основой наиболее популярной технологии локальных сетей, насчитывающих в своих рядах более 60 миллионов узлов.

### ПОРА ПРИМЕРЯТЬ TUXEDO 6

Как заявили представители компании Novell Inc., в новую версию популярной системы Tuxedo, которая, вероятно, получит название System 6, будет включена поддержка службы директо-

рий (NDS) и протокола SNMP. Как средство контроля доступа вводится механизм списков управления доступом. Начало продаж Tuxedo 6 запланировано на четвертый квартал 1995 года.

### НА УОЛЛСТРИТЕ ОБЪЕКТЫ УЖЕ РАСПРЕДЕЛЯЮТ. А У НАС?

Представители компании SunSoft Inc. объявили 14 июня о начале предварительных поставок своей распределенной объектной среды на базе ОС Solaris — DOE (Distributed Objects Environment или, что звучит лучше, Distributed Objects Everywhere — распределенные объекты повсюду). Новый продукт получили около ста тщательно отобранных организаций, среди которых крупнейшие фирмы Уоллстрита, поставщики телекоммуникационных услуг и программного обеспечения. По мнению специалистов SunSoft, DOE позволит фирмам быстрее реагировать на изменения рыночной ситуации путем построения распределенных корпоратив-

ных информационных систем из многократно используемых сетевых объектов. Такие системы (по сравнению с традиционными) гораздо легче подгонять под нужды конкретных клиентов, модифицировать, администрировать и сопровождать.

Специалисты компании SunSoft считают, что им удалось создать первое операционное окружение, удовлетворяющее спецификациям CORBA (Common Object Request Broker Architecture), являющееся устойчивым и масштабируемым, способным поддерживать корпоративные системы в архитектуре клиент/сервер.

### ДАЖЕ ДОМОХОЗЯЙКА СМОЖЕТ АДМИНИСТРИРОВАТЬ NETRA

Представители компании Sun Microsystems Computer Company (SMCC) объявили о выпуске версии 2.0 Internet-сервера Netra. Ее отличительная черта — простота установки и администрирования, достигающаяся за счет использования графического интерфейса и гипертекстовых возможностей.

В комплект поставки входят Web-сервер, средства поддержки службы

имен, электронной почты, удаленного доступа и т.д. Среди необязательных компонентов упомянем брандмауэр (средство защиты локальной сети) Firewall-1 и ряд продуктов семейства Netscape (в частности, торговый сервер). В третьем квартале 1995 года к этому набору предполагается добавить поддержку клиентов под NetWare.



# Архитектура сервера INFORMIX-OnLine Dynamic Server 7.1 и коммуникационные средства

## Содержание

1. ВВЕДЕНИЕ
2. СЕРВЕР INFORMIX-ONLINE DYNAMIC SERVER 7.1
  - 2.1 Динамическая масштабируемая архитектура
  - 2.2 Организация разделяемой памяти
  - 2.3 Организация операций обмена с дисками
  - 2.4 Поддержка фрагментации таблиц и индексов
  - 2.5 Параллельная обработка запросов
  - 2.6 Оптимизатор выполнения запросов по стоимости
  - 2.7 Средства обеспечения надежности
  - 2.8 Динамическое администрирование
  - 2.9 Распределенные вычисления
  - 2.10 Поддержка национальных языков
  - 2.11 Средства безопасности класса C2
3. INFORMIX-ENTERPRISE GATEWAY 7.1
  - 3.1 Технология и компоненты EDA/SQL
  - 3.2 Возможности Enterprise Gateway
4. БИБЛИОТЕКИ СОПРЯЖЕНИЯ СЕРВЕРА INFORMIX-ONLINE DS С МЕНЕДЖЕРАМИ ТРАНЗАКЦИЙ: INFORMIX-TP/XA И INFORMIX-TP/TOOLKIT
5. ЗАКЛЮЧЕНИЕ
6. ЛИТЕРАТУРА

## 1. Введение

Реляционная СУБД INFORMIX может служить основой для самых разных по масштабу и назначению информационных систем. Благодаря значительным усовершенствованиям, реализован-

ной в последних версиях, она способна обслуживать одновременно работающие приложения оперативной обработки транзакций и системы поддержки принятия решений для локальных и распределенных баз данных с большим числом пользователей. Отличительные черты последних версий INFORMIX — высокая производительность, надежность, возможность безостановочного функционирования.

Первая версия СУБД INFORMIX была реализована в 1980 г. для микрокомпьютера под управлением ОС UNIX, и получила признание как компактная и эффективная система по сравнению с распространенными в то время СУБД для платформы VAX/VMS. В 1985 г. вышла версия, в которой в качестве языка доступа к данным использовался SQL. Эта версия была названа INFORMIX-SQL и в дальнейшем была перенесена на несколько десятков платформ, от мейнфреймов до персональных машин. СУБД INFORMIX получила распространение, главным образом, на платформах, работающих под управлением ОС UNIX, хотя были созданы ее версии для MS-DOS и VMS.

СУБД INFORMIX традиционно использовалась для создания информационных систем малого или среднего масштаба, работающих в режиме оперативной обработки транзакций. Считалось, что ОС UNIX не способна обслуживать крупные аппаратные платформы. Однако, с развитием аппаратных архитектур, особенно с появлением симметричных многопроцессорных систем, а также с

развитием самой ОС UNIX ситуация изменилась. Компьютеры, работающие под управлением ОС UNIX, обладают высокой производительностью, и в связи с этим возникла необходимость пересмотреть архитектуру сервера баз данных INFORMIX. Начиная с версии 6.0, сервер баз данных INFORMIX-OnLine Dynamic Server (INFORMIX-OnLine DS) имеет многопоточную динамическую масштабируемую архитектуру (DSA — Dynamic Scalable Architecture), которая была разработана в содружестве с компанией Sequent. Эта архитектура призвана обеспечить максимальную поддержку систем SMP с масштабируемостью в соответствии с числом процессоров и других ресурсов.

Многопоточная архитектура сервера послужила базой для реализации технологии параллельной обработки запросов (Parallel Data Query — PDQ), которая включена в версию 7. Технология PDQ обеспечивает эффективное выполнение сложных запросов, характерных для систем поддержки принятия решений. Тестирования показали значительное превосходство INFORMIX в производительности на многопроцессорных платформах над его основными конкурентами в данной области.

Спектр продуктов СУБД INFORMIX включает три типа серверов, разнообразные инструменты разработки и средства доступа для конечных пользователей, коммуникационные средства. Ниже приведена краткая характеристика продуктов INFORMIX.

#### Серверы:

- INFORMIX-OnLine Dynamic Server — основной тип сервера, поддерживающий наиболее широкий спектр функциональных возможностей и класс безопасности C2.

- INFORMIX-Standard Engine (SE) — облегченный вариант сервера, простой в использовании, конфигурировании и администрировании, но не поддерживающий некоторых возможностей, в частности, больших бинарных объектов (Binary Large Objects — BLOB).

- INFORMIX-OnLine/Secure — версия сервера OnLine с усиленными средствами безопасности.

#### Инструменты разработки и средства доступа:

- INFORMIX-ESQL/C — среда программирования на языке C со встроенным SQL.

- INFORMIX-ESQL/COBOL — среда программирования на языке COBOL со встроенным SQL.

- INFORMIX-SQL — инструмент программирования на языке SQL.

- INFORMIX-4GL — инструмент программирования на языке 4-го поколения (4GL) со встроенным SQL, включает компилятор языка 4GL, средства построения экранных форм и меню.

- INFORMIX-4GL-RDS+ID — версия INFORMIX-4GL со средствами быстрой разработки (RDS) и интерактивной отладки (ID).

- INFORMIX-ViewPoint — средство доступа к базам данных, позволяет создавать графические формы, отчеты, запросы.

- INFORMIX-NewEra — графический объектно-ориентированный инструмент групповых прикладных разработок, основан на расширении языка 4GL — 4GL++.

#### Коммуникационные средства:

- INFORMIX-Enterprise Gateway — шлюз для доступа к многочисленным разнородным источникам данных из приложений, построенных средствами разработки INFORMIX или других фирм.

## 2. Сервер Informix-OnLine Dynamic Server 7.1

К СУБД, претендующим на роль информационной основы современных предприятий, предъявляются все новые и более жесткие требования. К числу важнейших можно отнести следующие:

- высокая производительность
- масштабируемость
- смешанная загрузка сервера разными типами задач
- непрерывная доступность данных

Данный раздел посвящен, главным образом, рассмотрению архитектурных особенностей и механизмов сервера INFORMIX-OnLine DS, направленных на удовлетворение перечисленных выше

требований. Приводится также информация о средствах распределенных вычислений, безопасности, поддержки национальной среды.

## 2.1. Динамическая масштабируемая архитектура

Архитектура сервера INFORMIX-OnLine DS получила название "динамическая масштабируемая архитектура" (DSA). Суть ее заключается в том, что одновременно выполняется относительно небольшое число серверных процессов (виртуальных процессоров), которые разделяют между собой работу по обслуживанию множества клиентов. По сравнению с более ранними моделями сервера INFORMIX, где для каждого клиента создавался индивидуальный серверный процесс (рис. 1), новая модель обладает рядом преимуществ:

- снижение нагрузки на операционную систему (число серверных процессов невелико);
- сокращение совокупной потребности клиентов в оперативной памяти;
- снижение конкуренции при одновременном использовании системных ресурсов;
- более рациональное по сравнению с ОС назначение приоритетов и планирование;

Для многопроцессорных платформ:

- равномерная загрузка наличных процессоров;
- ускорение обработки сложных запросов за счет параллельного выполнения на нескольких процессорах.

Архитектура DSA полностью использует возможности симметричных многопроцессорных платформ SMP (Symmetric Multiprocessing systems), и может работать на однопроцессорных платформах. В последующих версиях предполагается расширить архитектуру сервера, обеспечив поддержку слабосвязанных систем и систем с массовым параллелизмом (MPP). Все базовые технологии DSA являются встроенными, они включены в библиотеки сервера, и их применение не зависит от особенностей ОС или аппаратных платформ различных поставщиков.

### 2.1.1. Потoki

Архитектуру INFORMIX-OnLine DS называют также многопоточковой. Для каждого клиента создается так называемый поток, или нить (thread). Поток — это подзадача, выполняемая в рамках одного из серверных процессов.

В некоторых случаях для обслуживания одного клиентского запроса создается несколько параллельных потоков. Потoki создаются также для выполнения внутренних задач сервера — ввода-вывода, журнализации, администрирования и др. Таким образом, одновременно выполняется множество потоков, которые распределяются между наличными виртуальными процессорами (рис. 2).

INFORMIX-OnLine DS не полагается на механизмы потоков, имеющиеся в некоторых операционных системах. Он формирует потоки, специфичные для задач обработки баз данных, оптимальные в отношении выделяемой под них



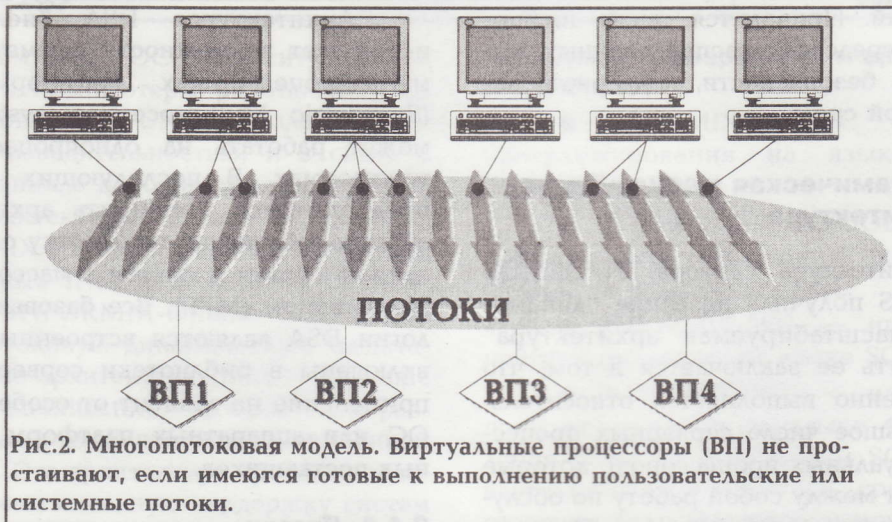


Рис.2. Многопоточковая модель. Виртуальные процессоры (ВП) не простаивают, если имеются готовые к выполнению пользовательские или системные потоки.

памяти, методов планирования и числа инструкций, затрачиваемых на переключение между потоками.

### 2.1.2. Виртуальные процессоры

Виртуальным процессором называется процесс сервера баз данных. Виртуальный процессор можно сравнить с операционной системой. Поток по отношению к нему выступает как процесс, подобно тому, как сам виртуальный процессор является процессом с точки зрения операционной системы.

Виртуальные процессоры (ВП) являются специализированными — они подразделяются на классы в соответствии с типом потоков, для выполнения которых они предназначены. Примеры классов ВП:

CPU — Потоки обслуживания клиентов, реализуют оптимизацию и логику выполнения запросов. К этому классу относятся и некоторые системные потоки.

АЮ — Операции асинхронного обмена с диском.

ADM — Административные функции, например, системный таймер.

TLI — Контроль сетевого взаимодействия посредством интерфейса TLI (Transport Layer Interface).

В отличие от операционной системы, которая должна обеспечивать выполнение произвольных процессов, классы виртуальных процессоров спроектированы для наиболее оптимального выполнения заданий определенного вида.

Начальное число виртуальных процессоров каждого класса, создаваемых

при запуске INFORMIX-OnLine DS, задается в конфигурационном файле. Однако, потребности в каждом виде обработки не всегда предсказуемы. Инструменты администрирования позволяют динамически, не останавливая сервер, запустить дополнительные виртуальные процессоры. Например, если растет очередь потоков к виртуальным CPU-процессорам, то можно увеличить их число. Точно так же, возможно добавление виртуальных процессоров обмена с дисками, сетевых процессоров взаимодействия с клиентами, создание процессора обмена с оптическим диском, если он отсутствовал в начальной конфигурации. Динамически сократить можно только число виртуальных процессоров класса CPU.

На некоторых мультипроцессорных платформах, где OnLine DS поддерживает родство процессоров (processor affinity), допускается привязка виртуальных CPU-процессоров к определенным центральным процессорам компьютера. В результате производительность виртуального CPU-процессора повышается, поскольку операционная система реже производит переключение процессов. Привязка позволяет также изолировать работу с базой данных, выделяя для этой цели определенные процессоры, в то время как остальные будут заняты другими задачами.

### 2.1.3. Планирование потоков

Сервер осведомлен о степени значимости различных потоков и в соответствии с этим назначает для них приоритеты. Например, потоки ввода-вывода получают приоритеты следую-

щим образом:

- ввод-вывод логической журнализации — наивысший приоритет;
- ввод-вывод физической журнализации — второй по значимости приоритет;
- прочие операции ввода-вывода — низший приоритет.

Таким образом, гарантируется, что операции записи в логический журнал, от которых зависит восстановление базы данных в случае сбоя, не окажутся в очереди позади операции вывода во временный рабочий файл.

Сами виртуальные процессоры выполняются как высокоприоритетные процессы операционной системы, которые не прерываются, пока не пусты очереди готовых к выполнению потоков.

Выполнение потока не откладывается по истечении заданного кванта времени, как это происходит с процессами в операционной системе. Поток откладывается в двух случаях:

- когда он временно не может выполняться, например, если необходимо дождаться завершения обмена с диском, ввода данных от клиента, снятия блокировки.

- когда в коде потока встречаются обращения к функции `yield`. Обращения к ней вставляются при компиляции запросов, требующих длительной обработки, чтобы их выполнение не тормозило прохождение других потоков. Для этого выбираются точки, наиболее безболезненные для выполнения потока.

#### **2.1.4. Разделение потоков между виртуальными процессорами.**

Для каждого класса поддерживаются три очереди потоков, которые разделяются всеми виртуальными процессорами данного класса:

- Очередь готовых к выполнению потоков.

- Очередь спящих потоков. В нее помещается, например, CPU-поток, которому требуется доступ к диску. Предварительно CPU-поток порождает запрос на обмен с диском, для обслуживания которого формируется АІО-поток. Завершив обмен с диском, АІО-поток оповещает об этом виртуальный процессор

CPU, который "будит" спящий CPU-поток и перемещает его в очередь готовых потоков.

- Очередь ждущих потоков. Эта очередь служит для координации доступа потоков к разделяемым ресурсам. В нее помещаются потоки, ожидающие какого-либо события, например, освобождения заблокированного ресурса. Когда поток, заблокировавший этот ресурс, готов освободить его, просматривается очередь ждущих потоков. Если в ней есть поток, ожидающий именно этот ресурс, то он перемещается в очередь готовых.

Если выполняемый поток завершается, засыпает или откладывается, то освободившийся виртуальный процессор выбирает из очереди готовых очередной поток с наивысшим приоритетом. Как правило, OnLine DS стремится выполнять поток на одном и том же виртуальном процессоре, поскольку передача его другому процессору требует пересылки некоторого объема данных. Тем не менее, если поток готов к выполнению, он может быть продолжен другим процессором, с целью исключения простоев и обеспечения общего баланса загрузки.

#### **2.1.5. Экономия памяти и других ресурсов**

Рациональное использование ресурсов операционной системы достигается за счет того, что потоки разделяют ресурсы (память, коммуникационные порты, файлы) виртуального процессора, на котором они выполняются. Виртуальный процессор сам координирует доступ потоков к своим ресурсам. Процессы же, в отличие от потоков, имеют индивидуальные наборы ресурсов, и, если ресурс требуется нескольким процессам, то доступ к нему регулируется операционной системой.

Переключение виртуального процессора с одного потока на другой, в целом, происходит быстрее, чем переключение операционной системы с одного процесса на другой. Операционная система должна прервать один процесс, выполняемый центральным процессором, сохранить его текущее состояние (контекст) и запустить другой процесс, пред-

варительно поместив в ядро его контекст, что требует физической перезаписи фрагментов памяти. Поскольку потоки разделяют виртуальную память и дескрипторы файлов, то переключение виртуального процессора с потока на поток может сводиться к перезаписи небольшого управляющего блока потока, что соответствует выполнению примерно 20 машинных команд. При этом виртуальный процессор как процесс операционной системы продолжает выполняться без прерывания.

## 2.2. Организация разделяемой памяти

Разделяемая память — это механизм операционной системы, на котором основано разделение данных между виртуальными процессорами и потоками сервера. Разделение данных позволяет:

- Снизить общее потребление памяти, поскольку участвующим в разделении процессам, т. е. виртуальным процессорам, нет нужды поддерживать свои копии информации, находящейся в разделяемой памяти.

- Сократить число обменов с дисками, потому что буферы ввода-вывода сбрасываются на диск не для каждого процесса в отдельности, а образуют один общий для всего сервера баз данных пул. Виртуальный процессор зачастую избегает выполнения или обращения за результатами операций ввода с диска, поскольку нужная таблица уже прочитана другим процессором.

- Организовать быстрое взаимодействие между процессами. Через разделяемую память, в частности, обмениваются данными потоки, участвующие в параллельной обработке сложного запроса. Разделяемая память используется также для организации взаимодействия между локальным клиентом и сервером.

Управление разделяемой памятью реализовано таким образом, что ее фрагментация минимизируется, поэтому производительность сервера при ее использовании не деградирует с течением времени. Изначально выделенные сегменты разделяемой памяти наращиваются по мере надобности автоматически или вручную. При освобождении памя-

ти, занятой сервером, она возвращается операционной системе.

В разделяемой памяти находится информация обо всех выполняемых потоках, поэтому потоки относительно быстро переключаются между виртуальными процессорами. В частности, в разделяемой памяти выделяется область стеков потоков. Стек хранит данные для функций, выполняемых потоком, и другую информацию о состоянии пользовательского сеанса. Размер стека для каждого сеанса устанавливается при помощи переменной окружения.

Важный оптимизирующий механизм сервера — кэши хранимых процедур и словарей данных. Словари данных (system catalog), доступные только на чтение, а также хранимые процедуры, разделяются между всеми пользователями сервера, что позволяет оптимизировать совокупное использование памяти. При загрузке в разделяемую память словарь данных записывается в структуры, обеспечивающие быстрый доступ к информации, а хранимые процедуры преобразуются в выполняемый формат. Все это может существенно ускорить выполнение приложений, обращающихся ко многим таблицам с большим числом столбцов и/или ко многим хранимым процедурам.

## 2.3. Организация операций обмена с дисками

Операции ввода-вывода, как правило, образуют наиболее медленную компоненту обработки баз данных. Поэтому от их реализации существенно зависит общая продуктивность сервера. Для оптимизации ввода-вывода и повышения надежности в сервере INFORMIX-OnLine DS применяются следующие механизмы:

- собственное управление дисковой памятью;
- асинхронный ввод-вывод;
- опережающее чтение.

### 2.3.1. Управление дисковой памятью

INFORMIX-OnLine DS поддерживает как собственный механизм управления дисковой памятью, так и управление средствами файловой системы ОС



UNIX. Преимущества собственного механизма управления дисковой памятью:

- Снятие ограничений операционной системы на число одновременно читаемых таблиц.
- Оптимизация размещения таблиц — для таблиц выделяются большие области последовательных физических блоков, в результате ускоряется доступ к ним.
- Снижение накладных расходов при чтении — данные с дисков считываются непосредственно в разделяемую память, минуя буферы ОС.
- Повышение надежности. При использовании файловой системы INFORMIX-OnLine DS не может гарантировать, что в случае сбоя данные журнала транзакций не пропадут из-за того, что они остались в буферах ОС и не успели записаться на диск. Поэтому процедура быстрого восстановления, вызываемая при перезапуске системы, не обеспечит в этом случае целостности данных.

Файловую систему используют в ситуациях, когда нет возможности выделить под базы данных специальные разделы на дисках, или если перечисленные соображения не критичны.

### 2.3.2. Асинхронный ввод-вывод

Для ускорения операций ввода-вывода сервер использует собственный пакет асинхронного ввода-вывода (AIO) или пакет асинхронного ввода-вывода ядра ОС (KAIO), если он доступен. Пользовательские запросы на ввод-вывод обрабатываются асинхронно, поэтому виртуальным процессорам CPU не приходится ждать завершения операции обмена, чтобы продолжить обработку.

### 2.3.3. Опережающее чтение

Сервер OnLine DS может быть сконфигурирован таким образом, чтобы при чтении последовательной таблицы или индексного файла обеспечивалось опережающее чтение нескольких страниц в то время, пока обрабатываются уже прочитанные в разделяемую память данные. Таким образом, сокращается время ожидания обмена с диском, и пользователь быстрее получает результаты запроса.

### 2.4. Поддержка фрагментации таблиц и индексов

INFORMIX-OnLine DS поддерживает горизонтальную локальную фрагмен-



Рис.3. Операции чтения-записи фрагментированной таблицы выполняются параллельно, в результате время обработки сокращается пропорционально числу фрагментов.

тацию таблиц. Это такой способ хранения таблицы, когда совокупность ее строк разбивается на несколько групп согласно некоторому правилу, и эти группы хранятся на разных дисковых разделах. Фрагментация таблиц способствует достижению следующих целей:

- Сокращается время обработки одного запроса. Встроенный в INFORMIX-OnLine DS механизм PDQ при обработке запросов использует информацию о фрагментации таблиц и создает для сканирования таблицы несколько параллельных потоков. Если стратегия фрагментации выбрана удачно, то ускорение при выборке из таблицы практически линейно зависит от числа фрагментов (рис. 3).

- Снижается уровень конкуренции при одновременном обращении нескольких запросов к одной таблице. INFORMIX-OnLine DS анализирует правило фрагментации таблицы и во многих случаях способен определить, что данный запрос относится только к одному ее фрагменту. Если фрагменты хранятся на разных дисковых устройствах, то разным запросам будут соответствовать обращения к разным дискам.

- Повышается готовность (доступность) приложений. Даже если некоторые фрагменты таблицы недоступны из-за того, что соответствующие диски вышли из строя, запросы к ней, тем не менее, во многих случаях могут выполняться.

- Улучшаются характеристики административных операций, таких как архивирование-восстановление, загрузка-выгрузка данных, поскольку они применимы к отдельным фрагментам таблиц. Если таблица разбита на малые фрагменты, то ее восстановление при выходе из строя одного фрагмента выполняется значительно оперативнее, чем полное восстановление нефрагментированной таблицы. Полные операции архивирования, восстановления, загрузки, выгрузки данных также ускоряются, поскольку операции ввода-вывода для фрагментов таблицы выполняются параллельно.

Различаются два типа правил фрагментации таблиц:

- **Равномерное распределение (round robin)** — это встроенный в INFORMIX-OnLine DS механизм фрагментации, который обеспечивает примерно равное число записей в каждом фрагменте.

- **Распределение по выражению (by expression)** — для каждого фрагмента задается некоторое выражение, зависящее от значений полей записи; истинность выражения определяет, попадет ли запись в данный фрагмент.

Правило разбиения таблицы задается в SQL-инструкциях CREATE TABLE (создать таблицу), ALTER TABLE (изменить таблицу). Пример:

```
CREATE TABLE account ...
FRAGMENT BY EXPRESSION
  id_num > 0 AND id_num <= 20
    IN dbsp1
  id_num >20 AND id_num <= 40
    IN dbsp2
  REMAINDER IN dbsp3
```

Здесь dbsp1, dbsp2, dbsp3 — имена областей дискового пространства, выделенного под БД.

INFORMIX-OnLine DS поддерживает также фрагментацию индексов. Различаются два вида фрагментации индексов — зависящая (соответствующая фрагментации таблицы) и независимая. Фрагментированной таблице может соответствовать нефрагментированный индекс. Создание индекса с правилом фрагментации, не совпадающим с правилом фрагментации таблицы, полезно в тех случаях, когда в разных приложениях выборки из таблицы осуществляются на основе разных подмножеств ее атрибутов.

Стратегия фрагментации таблиц и индексов выбирается в зависимости от цели, которая преследуется, от структуры таблицы и характера запросов к ней. Различные стратегии подробно описаны в документации. Например, если основной целью является уменьшение конкуренции при одновременном доступе к таблице, то оптимальной будет фрагментация таблицы по диапазонам значения ключа (или другого столбца, на основе которого производится основной доступ к таблице) и зависящая фрагментация индекса.

INFORMIX-OnLine DS предоставляет средства наблюдения, позволяющие оценить эффективность фрагментации таблиц и индексов по следующим параметрам:

- Распределение данных по фрагментам;
- Баланс запросов на ввод-вывод по фрагментам;
- Статус дисковых областей, содержащих фрагменты.

Если наблюдения показывают, что выбранная стратегия не удовлетворяет поставленной цели, то правила фрагментации могут быть изменены динамически, без остановки сервера.

Важно, что фрагментация таблиц и индексов прозрачна для приложений, работающих с базой данных. Изменение правил фрагментации не требует никаких изменений в прикладных системах — оно лишь повышает (или снижает) скорость и экономичность их выполнения.

## 2.5. Параллельная обработка запросов

Параллельная обработка запросов (Parallel Data Query, PDQ) — это технология, которая позволяет распределить обработку одного сложного запроса на несколько процессоров, мобилизовать для его выполнения максимально доступные системные ресурсы, во много раз сокращая время получения результата. Основные типы заданий, на которых проявляется эффект технологии PDQ:

- обработка сложных запросов, включающих сканирование больших таблиц, сортировку, соединения, группирование, массовые вставки;
- построение индексов;
- сохранение и восстановление данных;
- загрузка, выгрузка данных, реорганизация баз данных;
- массовые операции вставки, удаления, модификации данных.

Практически это означает, что отчет или ответ на сложный запрос, от которого зависит принятие ответственного решения, можно получить не завтра (после ночной обработки), а непосред-

ственно во время обычной оперативной дневной работы. Снимаются проблемы, связанные с обработкой и обслуживанием (архивированием, копированием) очень больших таблиц — благодаря фрагментации, параллельной обработке и возможностям выполнения административных действий в оперативном режиме. В результате расширяется класс потенциальных приложений, и, соответственно, круг пользователей, более гибким становится режим работы ИС, причем все это достигается не на узкоспециализированных, а на обычных широко распространенных аппаратных платформах. Таким образом, можно говорить о новом качестве, которое привносит с собой технология PDQ.

Максимальные преимущества эта технология дает на многопроцессорных платформах в условиях применения фрагментации таблиц, где время выполнения запроса сокращается в десятки раз; однако выигрыш в производительности достигается и на однопроцессорных машинах и нефрагментированных таблицах за счет того, что доступ к дискам осуществляется параллельно с другими видами обработки, и за счет максимально полного использования памяти.

### 2.5.1. На чем основана технология PDQ

Реализация запроса состоит из отдельных действий — сканирования, сортировки, группирования и др. Эти действия называются итераторами. Итераторы образуют дерево реализации запроса в том смысле, что результаты выполнения одних итераторов являются исходными данными для других. При обычной обработке итераторы выполняются последовательно. В основе технологии PDQ лежат следующие виды оптимизации и регулирования:

- Параллельный ввод и вывод (на основе горизонтальной фрагментации таблиц).
- Распараллеливание отдельных итераторов (на основе методов разбиения данных).
- Распараллеливание плана выполнения запроса (путем разбиения дерева реализации запроса на независимые поддерева; за счет применения техники потоков данных).

- Снижение вычислительной сложности алгоритмов (применение основанных на хешировании алгоритмов сортировки, соединения, вычисления агрегатных функций (sum, min, max, avg, ...)).
- Управление ресурсами, регулирование степени распараллеливания (под PDQ выделяется определенная доля системных ресурсов).

### 2.5.2. Итераторы

Итератор — это программный объект, который осуществляет итеративную (циклическую) обработку некоторого множества данных. Итераторы различаются типом производимой обработки, но имеют единообразный внешний интерфейс (рис. 4). Каждый итератор открывает один (или более) входных потоков данных (data flow), последовательно считывает их и, после обработки, помещает результаты в выходной поток. Итератору безразличен источник входного потока и назначение выходного потока — это может быть диск, другой итератор, сетевое соединение. Мы будем говорить о поставщиках и потребителях потоков данных. Ниже перечислены типы итераторов, применяемые в INFORMIX-OnLine DS:

SCAN — Сканирует фрагментированные и нефрагментированные таблицы и индексы.

NESTED LOOP JOIN — Реализует стандартную логику соединений методом вложенных циклов (читает строку из одной таблицы, находит все совпадения

во второй таблице, читает следующую строку из первой таблицы и т. д.).

MERGE JOIN — Выполняет фазу слияния для соединения методом сортировки и слияния.

HASH JOIN — Реализует новый метод соединений с хешированием. Для одной из двух соединяемых таблиц строится хеш-таблица, вторая таблица зондируется. Оптимизатор решает, какая из таблиц будет хешироваться.

GROUP — Группирует данные (GROUP BY) и вычисляет агрегатные функции.

SORT — Сортирует данные.

MERGE — Выполняет объединения UNION и UNION ALL (для UNION используется комбинация итераторов MERGE и SORT).

REMOTE — Реализует удаленные сканирования для операторов SELECT.

Итератор как программный объект состоит из статических и динамических структур данных. Статическая структура содержит ссылки на функции (или методы), применимые к итератору. Динамическая структура содержит информацию о текущем состоянии итератора (открыт, закрыт, выполняет очередную итерацию), одну или две ссылки на поставщиков.

#### Методы итератора

CREATE() — Создает итератор. Выделяет память для итератора, инициализирует его структуры, а также

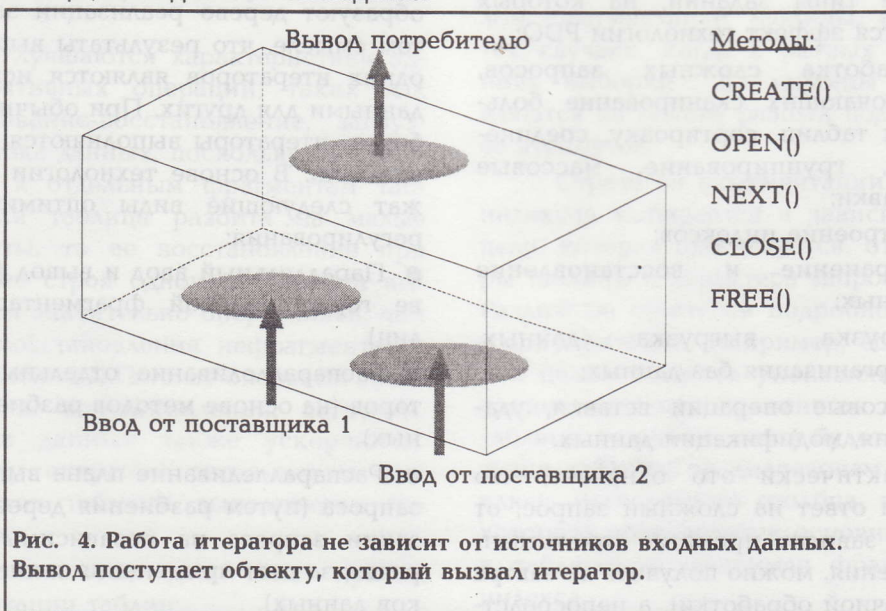


Рис. 4. Работа итератора не зависит от источников входных данных. Вывод поступает объекту, который вызвал итератор.

остальные методы (open(), next(), close(), free()), т.е. устанавливает ссылки на функции, соответствующие данному типу итератора. Затем вызывает метод create() для своих итераторов-поставщиков, которые создадут своих поставщиков, если таковые имеются, и т.д. Таким образом, вызов метода create() для корневого итератора приводит к созданию всего дерева итераторов.

OPEN() — Запускает итератор. Выполняются специфические для данного типа итератора инициализирующие действия, возможно, запрос дополнительной памяти. Например, при запуске итератора сканирования определяется, какие фрагменты необходимо сканировать, устанавливается указатель на первый из них, создается временная таблица (если она нужна), посылается сообщение MGM (MGM — компонента сервера, которая регулирует выделение ресурсов под запросы, обрабатываемые средствами PDQ; см. об этом ниже, п. "Баланс между OLTP и DSS-приложениями") о запуске потока сканирования. Далее применяется метод open() по отношению к поставщикам итератора, которые применят его к своим поставщикам и т.д. Таким образом, для запуска всего дерева итераторов достаточно применить метод open() к корневому итератору.

NEXT() — Выполняет одну итерацию. Выполнение начинается с того, что итератор применяет метод next() к своим поставщикам, заставляя их также применить next() к своим поставщикам и т.д., пока не сработают итерации поставщиков нижнего уровня. Затем данные поднимаются снизу вверх — каждый итератор, получив данные от своего поставщика, применяет к ним свой специфический вид обработки и передает результат своему потребителю. Метод next() применяется циклически, пока не поступит признак конца потока данных.

CLOSE() — Закрывает итератор. Высвобождает память, выделенную при запуске. Фактически, эта память могла уже быть высвобождена методом next(), когда он получил признак конца данных, поскольку общий принцип состоит в том, чтобы освободить память сразу же, как только она становится не нужна. Однако, это не всегда возможно. Поэтому на метод close() возлагается ответственность за то, чтобы память в любом случае была освобождена.

Метод close() рекурсивно применяется к поставщикам, тем самым, закрывается все дерево итераторов.

FREE() — Освобождает итератор. Высвобождает память, выделенную при создании. Применяет free() к постав-

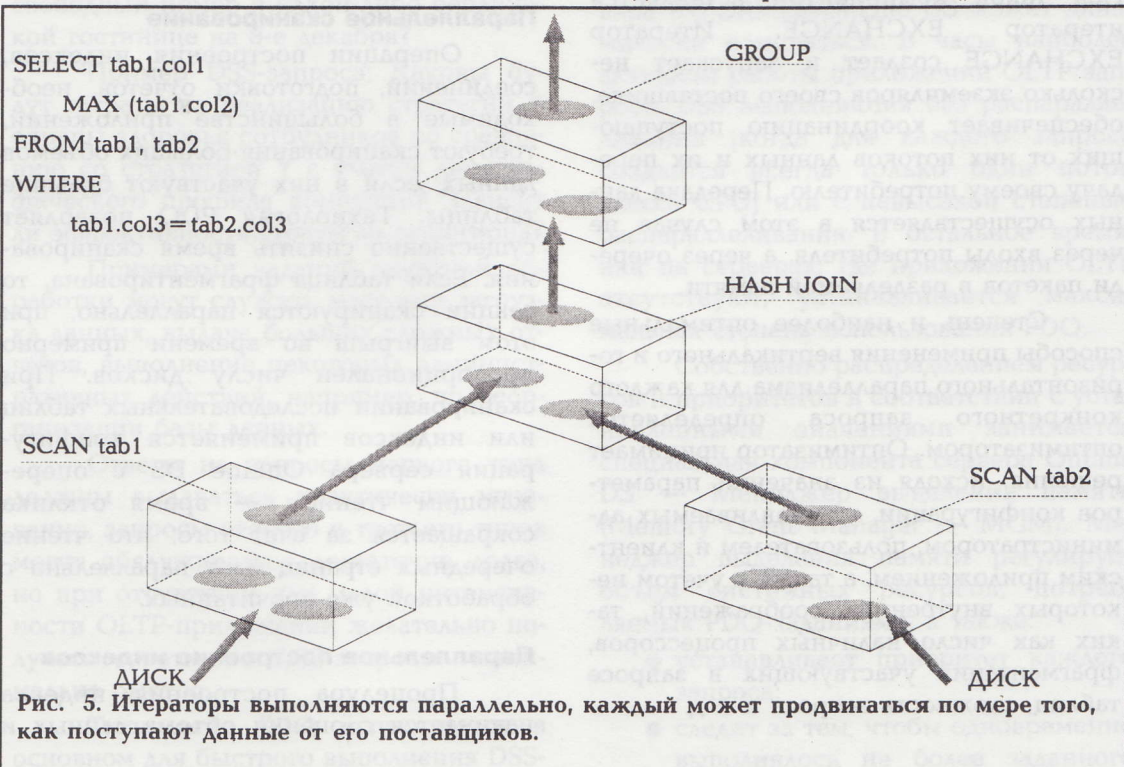


Рис. 5. Итераторы выполняются параллельно, каждый может продвигаться по мере того, как поступают данные от его поставщиков.

щикам, таким образом, освобождается все дерево итераторов.

Благодаря единообразию интерфейса, итераторы разных типов могут соединяться друг с другом произвольным образом (рис. 5). Итератор не заботится о том, какой тип имеют его поставщики, поскольку он взаимодействует с ними только посредством методов. Из описания методов следует, что запуск дерева, составленного из итераторов, реализует их параллельное выполнение. Для каждого итератора создается поток выполнения, который продвигается по мере того, как получает данные от своих поставщиков. Таким образом в сервере реализуется вертикальный параллелизм — одновременное, конвейерное выполнение различных итераторов.

Другой вид параллелизма — горизонтальный — заключается в том, что вместо одного итератора (например, сканирования) создается несколько однотипных параллельно выполняемых итераторов. Горизонтальный параллелизм реализуется при помощи итераторов специального вида — итераторов обмена (EXCHANGE). После того, как дерево реализации запроса построено, оптимизатор определяет, какие его компоненты имеет смысл распараллелить. Над такой компонентой вставляется итератор EXCHANGE. Итератор EXCHANGE создает и запускает несколько экземпляров своего поставщика, обеспечивает координацию поступающих от них потоков данных и их передачу своему потребителю. Передача данных осуществляется в этом случае не через входы потребителя, а через очереди пакетов в разделяемой памяти.

Степень и наиболее оптимальные способы применения вертикального и горизонтального параллелизма для каждого конкретного запроса определяется оптимизатором. Оптимизатор принимает решения, исходя из значений параметров конфигурации, устанавливаемых администратором, пользователем и клиентским приложением, а также с учетом некоторых внутренних соображений, таких как число наличных процессоров, фрагментация участвующих в запросе таблиц, сложность запроса и т. д.

Результаты тестов показывают, что механизмы PDQ и оптимизации INFORMIX-OnLine DS обеспечивают с увеличением числа процессоров практически пропорциональный рост производительности.

### 2.5.3. Примеры применения параллелизма

#### Параллельная сортировка

Сортировка — это фундаментальная операция обработки баз данных, применяемая при выполнении таких действий, как построение индексов, соединение методом сортировки и слияния, группирование; поэтому ускорение сортировки улучшает качество многих приложений.

При параллельной сортировке совокупность данных разбивается на секции, которые передаются для сортировки нескольким процессорам. Затем выполняется слияние отсортированных секций.

На практике скорость сортировки ограничивается временем сканирования данных из таблиц. Это ограничение в значительной мере снимается применением PDQ-алгоритмов параллельного сканирования.

#### Параллельное сканирование

Операции построения индексов, соединений, подготовки отчетов, необходимые в большинстве приложений, требуют сканирования больших объемов данных, если в них участвуют большие таблицы. Технология PDQ позволяет существенно снизить время сканирования. Если таблица фрагментирована, то секции сканируются параллельно, при этом выигрыш во времени примерно пропорционален числу дисков. При сканировании последовательных таблиц или индексов применяется конфигурация сервера OnLine DS с опережающим чтением — время отклика сокращается за счет того, что чтение очередных страниц идет параллельно с обработкой уже прочитанных.

#### Параллельное построение индексов

Процедура построения индекса начинается с оценки объема данных и

определения числа потоков, необходимых для их сканирования. Затем выполняется параллельное сканирование данных с применением, там, где это возможно, опережающего чтения. Считанные данные помещаются в участки разделяемой памяти, и запускается параллельная сортировка участков, для каждого из которых строится В-поддерево; затем из них формируется общий индекс. Потоки сортировки начинают выполняться, не дожидаясь завершения всех потоков сканирования, точно так же, поток построения индекса не ожидает завершения всех потоков сортировки — все, что можно, выполняется параллельно. В результате достигается ускорение, вплоть до десятикратного, по сравнению с последовательными методами построения индексов — в зависимости от объемов данных, числа используемых дисков и доступной памяти.

#### 2.5.4. Баланс между OLTP и DSS-приложениями

В современных информационных системах, как правило, требуется одновременное выполнение разных по характеру запросов к базе данных. Выделяются приложения обработки данных типа OLTP, DSS и пакетной обработки.

Пример OLTP-запроса: Есть ли свободный номер в какой-либо берлинской гостинице на 8-е декабря?

Пример DSS-запроса: Каковы будут затраты на реализацию стратегии X охраны здоровья сотрудников по сравнению со стратегией Y с учетом демографического профиля компании? Зависит ли эффективность стратегии от региона?

Примерами заданий пакетной обработки могут служить массовая загрузка данных, выдача больших сложных отчетов, выполнение некоторых административных действий, например, по реорганизации базы данных.

Ответы на запросы первого типа должны выдаваться практически мгновенно, запросы второго и третьего типов могут обслуживаться достаточно долго, но при отсутствии или малой интенсивности OLTP-приложений желательно получать ответы на DSS-запросы максимально быстро.

Технология PDQ используется в основном для быстрого выполнения DSS-

запросов и пакетных приложений. Если ее применение ничем не ограничено, то сильно распараллеленное выполнение нескольких сложных запросов приводит к недопустимому замедлению OLTP-приложений, выполняющихся на том же сервере. Управление степенью распараллеливания запросов и долей системных ресурсов, выделяемых для PDQ-обработки, в среде INFORMIX-OnLine DS осуществляется при помощи нескольких параметров конфигурирования и переменных окружения, значения которых динамически настраиваемы. Значения этих параметров и переменных устанавливаются системными администраторами и, в определенной степени, прикладными программистами и пользователями.

Программист или пользователь задает тип каждого запроса (обычный или PDQ) и желаемую степень распараллеливания для PDQ-запросов. Администратор, со своей стороны, динамически ограничивает максимальную допустимую степень распараллеливания PDQ-запросов, а также определяет долю системных ресурсов, выделяемых под обработку PDQ-запросов. Параллельная сортировка применяется для любых запросов, в том числе, обычных.

Таким образом, режим работы сервера INFORMIX-OnLine DS может динамически изменяться. В часы наиболее активной работы приложений OLTP запросы DSS выполняются без распараллеливания (когда для каждого запроса создается всегда только один поток класса CPU) или с невысокой степенью распараллеливания. В остальное время, или на серверах, где приложения OLTP отсутствуют, устанавливается максимальная степень использования PDQ.

Собственно распределением ресурсов и приоритетов в соответствии с установленными значениями занимается специальная компонента сервера OnLine DS — Менеджер выделения памяти (Memory Grant Manager — MGM). Менеджер выделения памяти регулирует объем системных ресурсов, потребляемых PDQ-заданиями, а также:

- устанавливает приоритет каждого запроса;
- следит за тем, чтобы одновременно выполнялось не более заданного

числа PDQ-запросов;

- следит за тем, чтобы объем разделяемой памяти, занятой под обработку сложных запросов, не превышал заданного уровня;
- совместно с оптимизатором запросов обеспечивает максимальную при заданных параметрах степень параллелизма на всех уровнях.

## **2.6. Оптимизатор выполнения запросов по стоимости**

Оптимизатор запросов определяет наиболее оптимальный с точки зрения затрат системных ресурсов план реализации каждого запроса к базе данных. Учитывается число обменов с диском, затраты разделяемой памяти, затраты на пересылку данных по сети и др. План может включать параллельное выполнение операций или быть строго последовательным, что зависит как от структуры запроса, так и от ресурсов, выделяемых MGM. Оптимизатор опирается на статистическую информацию о распределении данных по столбцам таблиц, периодическим сбором которой управляет администратор.

Например, если требуется выполнить соединение двух таблиц, находящихся в разных узлах сети, то оптимизатор спланирует эту операцию таким образом, что меньшая по объему таблица будет передана на сервер, содержащий большую таблицу, где и будет выполнено соединение (не обязательно выполнять его на том сервере, к которому произведено первое подключение). Дополнительная оптимизация достигается за счет фильтрации таблицы перед ее пересылкой, т. е. изъятия из нее не участвующих в данной операции соединения строк и/или столбцов.

Оптимизатор дает возможность разработчику предварительно получить план выполнения запроса, в том числе, распределенной транзакции. Получив такой план, разработчик может выяснить, что не располагает достаточной памятью, чтобы сохранить полученные в результате данные, или что выполнение запроса потребует слишком больших затрат системных ресурсов. В такой ситуации он либо отложит выполнение запроса на другое время, либо переформулирует

запрос так, чтобы сузить объем возвращаемых данных, либо примет какое-то другое решение.

Прикладной программист или пользователь устанавливает один из двух возможных уровней оптимизации — высокий или низкий. Высокий уровень оптимизации предполагает перебор большого числа возможных вариантов и сам требует больших затрат системных ресурсов, в частности, памяти. Оптимизация низкого уровня обходится дешевле, поскольку перебирается небольшое число предположительно оптимальных вариантов, но остается вероятность "упустить" наилучший вариант. Например, план выполнения хранимой процедуры вычисляется заранее с высоким уровнем оптимизации и сохраняется, после чего устанавливается низкий уровень — тогда при обращении к процедуре используется построенный заранее наиболее оптимальный план.

## **2.7. Средства обеспечения надежности**

Сервер INFORMIX-OnLine DS предоставляет следующие средства для восстановления после сбоев и обеспечения отказоустойчивости:

- Зеркалирование дисковых областей
- Полное тиражирование данных сервера
- Быстрое восстановление при включении системы
- Средства архивирования данных

### **2.7.1. Зеркалирование дисковых областей**

Зеркалирование в INFORMIX-OnLine DS — это дублирование связанной дисковой области, выделенной под базу данных, на такую же по размеру область. Исходная область называется первичной, а ее копия — зеркальной. Цели, для которых применяется зеркалирование — высокая готовность и оптимизация операций чтения.

Высокая готовность достигается за счет того, что при выходе из строя диска, на котором находится первичная область, сервер автоматически продолжает работу с оставшимся диском без перехо-



да сервера в режим off-line. Все операции чтения-записи происходят с зеркальной областью (при условии, что она находится на другом диске). Восстановление копии на первичном диске после его включения производится в оперативном режиме.

Затраты на зеркалирование складываются из затрат дискового пространства и затрат на дополнительные операции записи. В условиях, когда имеется несколько виртуальных процессоров обмена с диском, операции записи на оба диска производятся параллельно, и затраты второго рода сводятся к минимуму. К тому же они компенсируются оптимизацией операций чтения, о которой говорится ниже.

В идеальном случае зеркалирование должно быть обеспечено для всех областей базы данных. Крайне желательно поддерживать зеркалирование для критичных областей, составляющих корневое пространство базы данных и пространства, где хранятся логический и физический журналы. При выходе из строя любого из них, если нет зеркального дубля, сервер немедленно переводится в режим off-line. При отказе других незеркалируемых областей недоступными становятся только хранящиеся на них таблицы или фрагменты таблиц — до завершения процедуры их восстановления. Поэтому для наиболее критичных таблиц также желательно поддерживать зеркалирование.

Оптимизация операций чтения достигается за счет разбиения (split read). Страницы, относящиеся к начальной половине области, читаются с первичной области, а страницы из второй половины — с зеркальной. В результате ускоряется поиск страницы на диске, поскольку максимальный пробег дисковых головок сокращается вдвое.

### 2.7.2. Тиражирование

Тиражирование — это поддержание на другой вычислительной установке копии объектов базы данных. В INFORMIX-OnLine DS реализовано прозрачное тиражирование данных с основного сервера баз данных на вторичный (или поддерживающий) сервер, к которому разрешен доступ только на чтение

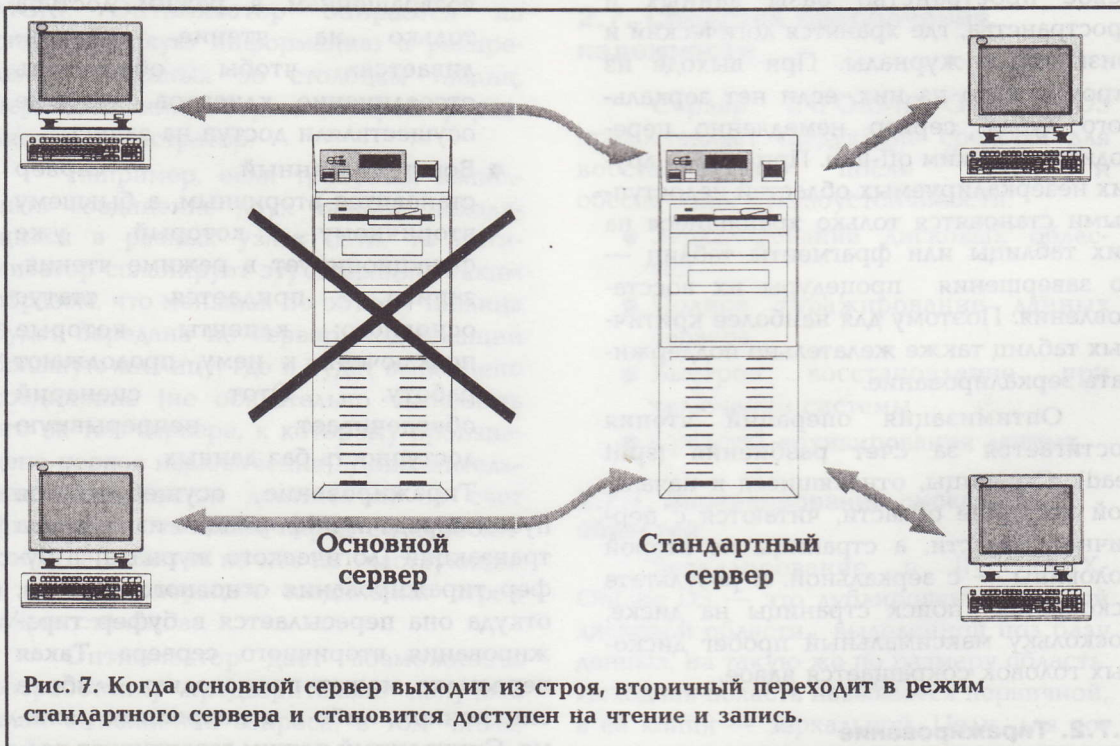
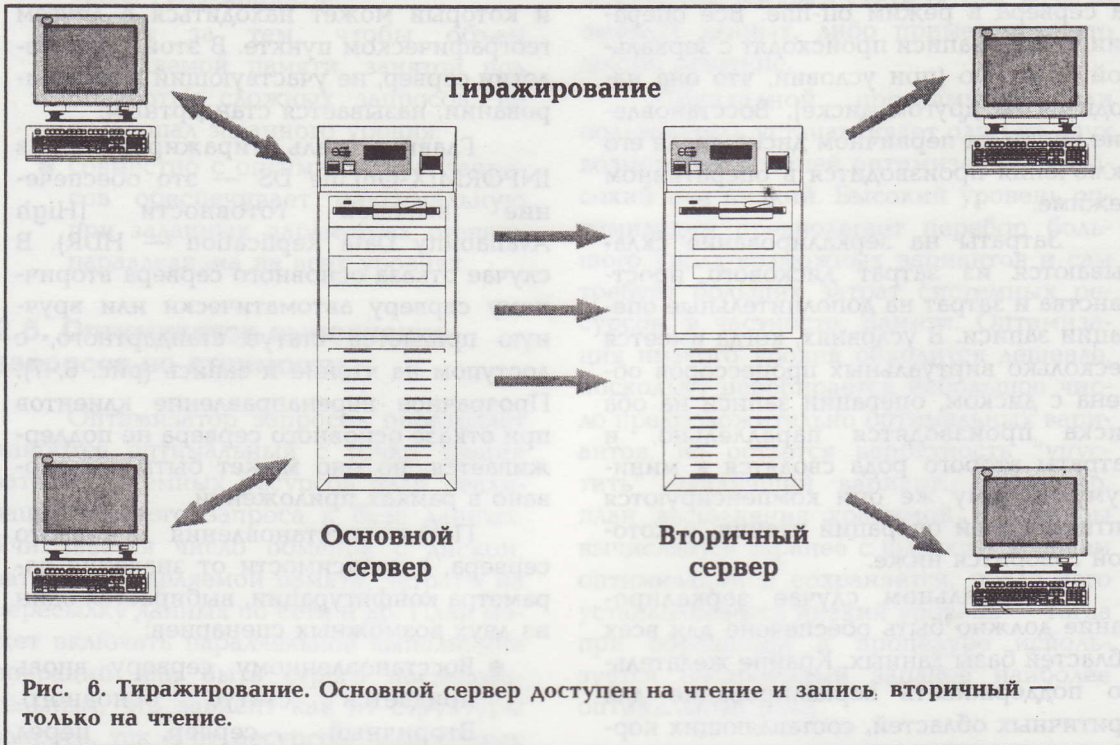
и который может находиться в другом географическом пункте. В этой терминологии сервер, не участвующий в тиражировании, называется стандартным.

Главная цель тиражирования в INFORMIX-OnLine DS — это обеспечение высокой готовности (High Availability Data Replication — HDR). В случае отказа основного сервера вторичному серверу автоматически или вручную придается статус стандартного, с доступом на чтение и запись (рис. 6, 7). Прозрачное перенаправление клиентов при отказе основного сервера не поддерживается, но оно может быть реализовано в рамках приложений.

После восстановления основного сервера, в зависимости от значения параметра конфигурации, выбирается один из двух возможных сценариев:

- Восстановленному серверу вновь придается статус основного. Вторичный сервер, перед возвращением в режим доступа только на чтение, останавливается, чтобы обеспечить отсоединение клиентов, которые осуществляли доступ на запись.
- Восстановленный сервер становится вторичным, а бывшему вторичному, который уже функционирует в режиме чтения-записи, придается статус основного; клиенты, которые подключены к нему, продолжают работу. Этот сценарий обеспечивает непрерывную доступность баз данных.

Тиражирование осуществляется путем передачи информации из журнала транзакций (логического журнала) в буфер тиражирования основного сервера, откуда она пересылается в буфер тиражирования вторичного сервера. Такая пересылка может происходить либо в синхронном, либо в асинхронном режиме. Синхронный режим гарантирует полную согласованность баз данных — ни одна транзакция, зафиксированная на основном сервере, не останется незафиксированной на вторичном, даже в случае сбоя основного сервера. Асинхронный режим не обеспечивает абсолютной согласованности, но улучшает рабочие характеристики системы.



Зеркалирование, которое также является прозрачным средством поддержания высокой готовности, обеспечивает только копирование дисковых областей в пределах одной установки сервера INFORMIX-OnLine DS и защищает только от дисковых сбоев. Механизм тиражирования обеспечивает поддержание

полной удаленной копии баз данных и защищает от всех видов отказов, включая полный крах одной из установок.

Помимо обеспечения отказоустойчивой работы, тиражирование дает следующие преимущества:

- более оперативный доступ к данным для локальных клиентов

вторичного сервера;

- возможность вынести приложения DSS преимущественно на вторичный сервер, где они выполняются с максимальным использованием PDQ, не подавляя приложений OLTP, выполняющихся на основном сервере.

### 2.7.3. Быстрое восстановление при включении системы

При включении сервер всегда проверяет, не произошло ли последнее выключение системы аварийно. В этом случае база данных не разрушена, но множество транзакций, выполнявшихся в момент сбоя, остались в незавершенном, некорректном состоянии. Если сервер выявил такую ситуацию, то он запускает процедуру быстрого восстановления, которая обеспечивает возврат системы в корректное состояние.

### 2.7.4. Архивирование и восстановление данных

INFORMIX-OnLine DS позволяет создавать архивные копии данных, а в дальнейшем фиксировать те изменения, которые произошли на сервере с момента создания архива. Изменения сохраняются в файлах журнала транзакций. Архивные ленты и ленты с копиями журналов транзакций могут записываться параллельно с доступом пользователей к серверу. Процедура восстановления состоит из двух шагов — считывания данных из архивной копии и применения к ним тех изменений, которые были зафиксированы в журналах транзакций.

В состав сервера INFORMIX-OnLine DS входит утилита OnArchive, предоставляющая развитые и гибкие средства архивирования, копирования журналов транзакций и восстановления информации. Ниже перечислены основные возможности этой утилиты:

- Архивирование и восстановление на уровне дисковых областей (dbspace). Архивируется одна или несколько дисковых областей. С одной архивной ленты можно восстановить одну или несколько дисковых областей. Поддерживается инкрементальное архивирование (т. е. сохранение только информа-

ции, которая изменилась с момента создания последнего полного или инкрементального архива).

- Восстановление данных в оперативном режиме. В случае сбоя носителя, если не затронуты критические для работоспособности INFORMIX-OnLine DS дисковые области, пользователи продолжают взаимодействие с сервером. Доступ к данным, находящимся на вышедшем из строя носителе, возобновляется после завершения процедуры восстановления.
- Составление и просмотр расписания операций архивирования и резервирования. Операции производятся автоматически, по заданному расписанию.
- Метки на архивных лентах. Наличие меток сводит к минимуму риск таких ошибок администратора, как запись на ленту, на которой находится часть активного в данный момент архива, или восстановление данных с устаревшей архивной ленты.
- Интерактивный интерфейс с оператором, основанный на меню и экранных формах.
- Средства восстановления при катастрофах. В случае катастрофического краха информация каталога архивирования/восстановления, поддерживаемого сервером INFORMIX-OnLine DS, может быть прочитана с заголовка архивной ленты.
- Множественные копии архивов на лентах. Сервер INFORMIX-OnLine DS способен одновременно создавать множественные копии архивов и журналов транзакций на нескольких лентопротяжных устройствах. В случае краха восстановление выполняется с любой из копий.
- Опции шифрования и сжатия архивов и журналов транзакций при записи. За счет сжатия объем необходимой вторичной памяти сокращается на величину от 20 до 50 процентов.
- Контроль циклической избыточностью (CRC). Этот метод позво-

ляет контролировать правильность информации, считываемой с ленты. При вводе данных с архивной ленты контрольная сумма, хранящаяся на ленте, сверяется с вычисляемой контрольной суммой.

- Предоставление определенным пользователям прав на архивирование и восстановление некоторых дисковых областей.

## 2.8. Динамическое администрирование

В условиях, когда базы данных увеличиваются в размерах, становятся распределенными и служат основой для особо ответственных приложений масштаба предприятия, которые должны работать круглосуточно, возрастает роль развитых динамических средств администрирования. Эти средства должны позволять администраторам оперативно следить за такими характеристиками работы сервера, как использование памяти и виртуальных процессоров, очереди асинхронного ввода-вывода, очереди пакетных заданий и приложений DSS, наличие дискового пространства, эффективность схем фрагментации и т. п. Если какие-то из характеристик неудовлетворительны, то необходима возможность динамически, не останавливая системы, изменить параметры конфигурации или запустить необходимые административные утилиты.

Большинство параметров конфигурации сервера являются динамически настраиваемыми, их можно изменить, не останавливая сервера, при помощи утилиты ON-Monitor. Помимо рассмотренных выше менеджера памяти (MGM) и утилиты архивирования OnArchive, средства администрирования сервера INFORMIX-OnLine DS включают также следующие компоненты: интерфейс мониторинга системы, утилиты DB/Cockpit и OnPerf, утилита параллельной загрузки/выгрузки данных.

### 2.8.1. Интерфейс мониторинга системы

Во время инициализации сервера OnLine DS автоматически создается база данных SMI (System Monitoring Interface).

Эта база содержит таблицы, которые позволяют получать следующую информацию о состоянии сервера:

- имена баз данных, идентификаторы их создателей, статус журнализации;
- статус пользователей, ожидающих ресурсов баз данных;
- профиль выполнения сервера (счетчики различных вызовов и событий);
- использование процессоров пользователями и системой;
- распределение дискового пространства;
- состояние журналов транзакций;
- состояние дисковых пространств (dbspaces);
- блокировки;
- состояние экстендов — последовательных сегментов дискового пространства, выделенных под хранение таблиц.

Во время работы сервера информация в базе данных SMI динамически обновляется. Она используется административными утилитами, к ней также можно обращаться посредством SQL-инструкции SELECT.

### 2.8.2. Утилита DB/Cockpit

DB/Cockpit — это утилита, которая предоставляет администраторам баз данных графический интерфейс для слежения за состоянием баз данных и выполнения необходимых административных действий. Основные возможности:

- выдача предупреждений для администратора, если заданные системные параметры достигли установленных предельных значений;
- контроль уровней серьезности системных проблем;
- монитор активности, который предоставляет сведения об использовании различных системных ресурсов;
- запись исторической информации и ее анализ, позволяют администратору следить за изменением определенных элементов данных;
- экраны профилей, которые могут работать в текстовом, схема-

тическом или графическом режимах.

Гибкие средства для определения критических значений параметров, при достижении которых администратор должен получить соответствующее предупреждение, позволяют предотвратить аномальные состояния сервера и постоянно поддерживать его высокую работоспособность.

Утилита DB/Cockpit имеет архитектуру клиент/сервер, и позволяет администратору следить за удаленным сервером. Она состоит из двух основных компонент — зондирующей (probe) и интерфейсной. Зондирующая компонента работает на том же сервере, где установлен подлежащий наблюдению сервер INFORMIX-OnLine DS; она выбирает информацию из базы данных SMI и непосредственно из разделяемой памяти сервера. На основе этой информации зондирующая компонента инициирует предупреждения для администратора, записывает заказанную историческую информацию, пересылает данные для оперативного наблюдения по запросам интерфейсной компоненты. Интерфейсная компонента работает на любой машине в сети, в том числе, на той, где установлен сервер баз данных, Она обеспечивает пользовательский интерфейс для слежения за сервером INFORMIX-OnLine DS, посылает запросы на информацию о состоянии и конфигурации сервера, анализирует историческую информацию, выдает полученные от зондирующей компоненты предупреждения.

Утилита DB/Cockpit не требует больших затрат системных ресурсов. Существенно, что зондирующая компонента может работать независимо, и служить "сторожем" для сервера INFORMIX-OnLine DS.

### 2.8.3. Утилита OnPerf

OnPerf — утилита с графическим интерфейсом, которая является развитием имевшейся в предыдущих версиях INFORMIX-OnLine утилиты tbstat. Основные новые возможности:

- графический показ метрических значений в реальном времени;

- выбор метрик, за которыми нужно наблюдать;
- просмотр ранее собранной информации, чтобы проследить за тенденциями изменения метрики;
- сохранение данных в файле, последующий показ данных в имитируемом реальном времени.

При запуске OnPerf формируются два процесса — процесс OnPerf и процесс сбора данных. Процесс сбора данных подключается к разделяемой памяти INFORMIX-OnLine DS и считывает из нее метрики выполнения сервера. Собранные данные передаются процессу OnPerf, который обеспечивает их вывод в графической форме.

OnPerf позволяет администратору задать ряд метрик, которые необходимо буферизовать. Процесс сбора данных записывает такие метрики в буферы сбора данных, откуда администратор периодически сбрасывает информацию в файлы. Содержимое этих файлов можно затем просматривать при помощи утилиты OnPerf.

Выделяется несколько уровней метрик, доступных для слежения, — база данных, операционная система, центральный процессор, виртуальный процессор, пользовательский сеанс, дисковая область.

### 2.8.4. Утилита параллельной загрузки

Утилита параллельной загрузки способна параллельно считывать данные из нескольких источников, ускоряя за счет этого процедуры загрузки и выгрузки данных. Предоставляемый ею графический интерфейс позволяет администратору базы данных:

- указать тип файла (ASCII, COBOL, EBCDIC и др.), из которого производится загрузка, и задать необходимые преобразования (например, из EBCDIC в ASCII);
- задать соответствие между структурой загружаемого файла и схемой INFORMIX-базы данных;
- задать выборочную загрузку;
- вызвать просмотрщик (browser) загружаемого файла.

Утилита работает в одном из двух возможных режимов. В режиме быст-

рой загрузки, действия, обычно сопровождающие загрузку — проверка целостности по ссылкам, журнализация, построение индексов — выполняются не параллельно с загрузкой, а после ее завершения, что ускоряет сам процесс загрузки.

## 2.9. Распределенные вычисления

### 2.9.1. Взаимодействие клиент-сервер

Продукты INFORMIX построены на принципах архитектуры клиент/сервер. Это означает, что сервер INFORMIX-OnLine DS выполняется на одном компьютере, а клиентские приложения выполняются на других компьютерах, связанных с сервером сетью. При этом от клиентских приложений серверу по сети пересылаются только SQL-запросы, а от сервера на клиентские машины пересылаются результаты выполнения запросов. Преимущества такой архитектуры заключаются в том, что серверный компьютер, не загруженный выполнением клиентских приложений, способен эффективно обслужить большее число клиентов. Пользователи же в этом случае могут выбрать наиболее удобную для себя платформу, например, персональный компьютер с MS Windows. В частном случае клиент выполняется на той же машине, что и сервер.

Сервер INFORMIX-OnLine DS содержит все необходимые средства для организации взаимодействия локальных или удаленных клиентов с сервером базы данных, поэтому приобретение дополнительных продуктов не требуется.

Для организации взаимодействия клиентских приложений версий 5.0 или 4.1 с сервером INFORMIX-OnLine DS 7.1 в комплект поставки включен релейный модуль связи (Relay Module 7.1). Он может использоваться как для локального, так и для сетевого взаимодействия. Сетевое взаимодействие клиентских приложений версий меньше 6.0 с сервером INFORMIX-OnLine DS 7.1 возможно также при посредстве одного из коммуникационных продуктов INFORMIX-NET 5.0 или INFORMIX-STAR 5.0, который должен быть установлен на клиентской машине, в том числе, на PC.

Поддерживаются сетевые протоколы TCP/IP и SPX/IPX. Протокол TCP/IP реализуется посредством интерфейса сокетов UNIX или TLI, протокол SPX/IPX — посредством интерфейса TLI. Обработкой сетевого взаимодействия клиентов и серверов в INFORMIX-OnLine DS занимаются сетевые виртуальные процессоры. В конфигурацию сервера, в зависимости от интенсивности сетевого взаимодействия, включается необходимое число сетевых виртуальных процессоров. Обработка сетевого взаимодействия равномерно распределяется между сетевыми виртуальными процессорами.

Конфигурация разделяемой памяти включает коммуникационную область, через которую локальные клиенты могут взаимодействовать с сервером. Этот вид взаимодействия наиболее быстрый, и, кроме того, позволяет разгрузить сеть. Связь через разделяемую память осуществляется совместно с сетевыми подключениями для удаленных клиентов.

### 2.9.2. Прозрачность расположения данных

Если в сети имеется несколько серверов баз данных, то, в целях повышения эффективности доступа к данным или из других соображений, администраторы могут перемещать или дублировать базы данных или таблицы с одного сервера на другой. Механизм синонимов, поддерживаемый INFORMIX-OnLine DS, позволяет экранировать от прикладных программ изменения местоположения данных.

### 2.9.3. Распределенные базы данных и протокол двухфазовой фиксации транзакций

INFORMIX-OnLine DS поддерживает запросы к распределенным базам данных и автоматически применяет протокол двухфазовой фиксации для транзакций, которые модифицируют данные более чем на одном сервере баз данных, например:

```
CONNECT TO stores@italy
BEGIN WORK
UPDATE stores:manufact SET
    manu_code = 'SHM'
WHERE manu_name =
    'Shimara'
```

```
INSERT INTO
stores@france:manufact
VALUES ('SHM', 'Shimara',
30)
INSERT INTO
stores@australia:manufact
VALUES ('SHM', 'Shimara',
30)
COMMIT WORK
```

Здесь **BEGIN WORK**, **COMMIT WORK** — инструкции, отмечающие начало и конец транзакции, **stores** — имя базы данных, **italy**, **france**, **australia** — имена серверов.

Внешне такая транзакция выглядит как транзакция в локальной базе. На самом деле она состоит из ряда локальных транзакций, каждая из которых может быть либо зафиксирована, либо прервана. Распределенная транзакция фиксируется только в том случае, если зафиксированы все локальные транзакции. Если хотя бы одна из локальных транзакций была прервана, то необходимо прервать и все остальные.

Каждая транзакция, реализуемая согласно протоколу двухфазовой фиксации, выполняется под управлением одного сервера, называемого координатором. В качестве координатора выбирается текущий сервер. В примере выше это будет сервер **italy**, поскольку к нему относится оператор **CONNECT**.

Первая фаза начинается с того, что координатор, получив от пользователя инструкцию **COMMIT WORK**, рассылает серверам-участникам сообщения о том, что нужно подготовиться к фиксации. Каждый участник решает, может ли он зафиксировать свою часть транзакции, и посылает соответствующее сообщение координатору.

Вторая фаза начинается, когда координатор, получив сообщения от участников, принимает решение о фиксации или откате транзакции. Если все участники прислали положительные ответы, то координатор посылает им сообщения о том, чтобы они зафиксировали свои локальные транзакции. Если хотя бы один участник прислал отрицательный ответ или вообще не прислал ответа, то координатор прерывает транзакцию и посылает всем участникам

сообщение о том, что транзакцию нужно откатить.

### Процедура восстановления

Если один из серверов вышел из строя до завершения протокола двухфазовой фиксации транзакции, то необходимо восстановить совокупную согласованность распределенных данных. Для этой цели в **INFORMIX-OnLine DS** предусмотрены специальные процедуры восстановления, которые автоматически выполняют все необходимые действия с учетом того, в какой ситуации и на каком сервере произошел отказ. Единственное, что должен сделать в этой ситуации администратор — это перезапустить сервер.

### Оптимизация транзакций

При обработке распределенных транзакций **INFORMIX-OnLine DS** использует метод оптимизации, основанный на предположении о прерывании транзакции (**presumed abort optimization**). Смысл его заключается в том, что, если в журнале транзакций отсутствует информация о некоторой глобальной транзакции, то считается, что она прервана. Этот метод позволяет сократить число операций обмена с диском, а также число сообщений, пересылаемых между серверами.

Рассматриваемый метод оптимизации позволяет исключить два шага из классического протокола двухфазовой фиксации транзакций. Во-первых, координатор не производит синхронизированной записи на диск о начале транзакции. Синхронизированная запись на диск — дорогостоящая операция, и координатор производит ее только в двух случаях — когда все участники присылают сообщения "могу зафиксировать", и когда все участники присылают сообщения "транзакция зафиксирована". Если происходит отказ координатора до принятия решения о фиксации, и в журнале отсутствует информация о данной глобальной транзакции, то все участники считают, что она прервана, и откатывают свои части транзакции. Во-вторых, оптимизация достигается тем, что участники не должны посылать координатору подтверждения об откате

транзакции. Координатор, если он принял решение об откате, рассылает участникам соответствующие сообщения, и сразу же откатывает глобальную транзакцию, изымая информацию о ней из своей разделяемой памяти.

### Разрешение тупиковых ситуаций

Тупиковая ситуация возникает, например, когда работают два пользователя, и каждый блокирует объект данных, необходимый другому. Каждому из них, для того чтобы завершить обработку и разблокировать свой объект, необходимо получить доступ к объекту, заблокированному другим пользователем. Если оба объекта находятся на одном сервере, то INFORMIX-OnLine DS самостоятельно обнаруживает и предотвращает такие ситуации. При обработке распределенных запросов используется параметр конфигурации `DEADLOCK_TIMEOUT` — время, в течение которого INFORMIX-OnLine DS ожидает разблокирования объекта данных. По истечении этого периода одному из пользователей выдается сообщение об ошибке.

### 2.10. Поддержка национальных языков

Поддержка национальных языков (native language support — NLS) в INFORMIX основана на спецификации X/Open XPG3. Средства NLS в INFORMIX-OnLine DS поддерживают однобайтные 8-битные платформы NLS. Это позволяет осуществлять упорядочение текстовых данных, печатать и вводить даты и денежные величины по форматам и правилам, принятым в той стране, где используются продукты. Стандарт X/Open для NLS также обеспечивает миграцию приложений баз данных по странам, где используются разные языки, с сохранением исходной функциональности.

### 2.11. Средства безопасности класса C2

Реализованные в INFORMIX-OnLine DS средства протоколирования обеспечивают полную подотчетность лю-

бых манипуляций с объектами баз данных. Средства протоколирования полностью соответствуют требованиям класса безопасности C2, установленным Национальным центром компьютерной безопасности США. Имеется версия INFORMIX-OnLine/Secure, которая обеспечивает повышенный уровень безопасности.

Администратор может задавать как общие маски протоколирования, так и специфические маски для конкретных пользователей. Маска определяет, какие действия над объектами баз данных будут фиксироваться. Интерфейс с процедурой протоколирования осуществляется обращением к утилите `onaudit` из командной строки. Анализ регистрационного журнала производится при помощи утилиты `onshowaudit` или средствами SQL.

## 3. Informix-Enterprise Gateway 7.1

Шлюз INFORMIX-Enterprise Gateway обеспечивает для инструментальных средств и приложений баз данных, выполняемых под управлением операционной системы UNIX или Microsoft Windows, доступ к информации, хранящейся в базах данных разных типов. Доступ реализуется при помощи комплекта программных продуктов Enterprise Data Access SQL (EDA/SQL) фирмы Information Builders, Inc.

Основные возможности шлюза INFORMIX-Enterprise Gateway:

- Более 60 типов реляционных и нереляционных источников данных на 35 различных аппаратных платформах. Среди поддерживаемых источников данных — IMS, VSAM, CA-IDMS, Adabas, Oracle, Sybase, Ingres. Поддерживаемые операционные системы — UNIX, MVS, VM, VMS.
- Архитектура клиент/сервер.
- Прозрачный доступ посредством интерфейса SQL или удаленных вызовов процедур.
- Поддержка стандартов ANSI-92 SQL и ANSI-89 SQL.
- Роллируемые курсоры.



- Импорт данных из разнородных источников в базы данных *INFORMIX*.
- Распределенные соединения таблиц из разнородных баз данных.
- Средства безопасности.

В компаниях, которые ранее хранили и обрабатывали информацию на мэйнфреймах, формируются распределенные вычислительные среды, включающие разнородные аппаратные платформы и операционные системы, как открытые, так и собственные (proprietary), реляционные и нереляционные СУБД. Наличие такой среды — сложная проблема для отделов информационных систем, которые должны обеспечить своим пользователям единообразный доступ ко всей имеющейся на предприятии информации. Шлюз *INFORMIX-Enterprise Gateway* предлагает современную промышленную технологию интеграции, отвечающую потребностям корпоративного доступа к данным.

### **3.1. Технология и компоненты EDA/SQL**

Технология *EDA/SQL* фирмы *Information Builders, Inc.* позволяет осуществлять доступ средствами *SQL* не только к реляционным, но и к нереляционным источникам данных, таким как иерархические базы данных и файлы с определенной структурой записей (*record-oriented files*), характерные для мэйнфреймов. Ко всем данным, независимо от формата, обеспечивается унифицированный реляционный интерфейс. Технология *EDA/SQL* позволяет также производить соединения данных из разнородных источников.

Технология *EDA/SQL* основана на архитектуре клиент/сервер. Она включает четыре ключевых компоненты, необходимые для полного функционирования шлюза *Enterprise Gateway*.

#### **3.1.1. EDA API/SQL**

Продукт встроен в *Enterprise Gateway*.

*EDA API/SQL* — библиотека клиентской части, которая обеспечивает интерфейс уровня вызовов, определенный фирмой *Information Builders, Inc.*

Посредством этого интерфейса приложение клиента выполняет инструкции *SQL* или удаленные вызовы процедур.

#### **3.1.2. EDA/Link**

Продукт встроен в *Enterprise Gateway*.

*EDA/Link* — интерфейс обмена запросами между клиентами и серверами *EDA*. Интерфейсы *EDA/Link* поддерживают коммуникационные протоколы, формируют пакеты запросов и ответов, производят аутентификацию пользователей по паролям, преобразуют данные и выявляют ошибки передачи.

#### **3.1.3. EDA/SQL Server**

Независимый продукт, доступный от фирмы *Information Builders, Inc.*

*EDA/SQL Server* — многопоточковый сервер баз данных, который управляет выделением и соединением данных из реляционных и нереляционных источников. *EDA/SQL Server* управляет процессами на хост-машинах. Он управляет входным потоком запросов данных, инициализирует подпроцессы для интерпретации и трансляции запросов, вызывает и маршрутизирует хранимые процедуры, используя удаленные вызовы процедур, маршрутизирует вывод и осуществляет бюджетные функции и функции безопасности между сетевыми серверами.

*Enterprise Gateway* поддерживает *EDA/SQL Server* версии 2.2 и выше.

#### **3.1.4. EDA/Data Drivers**

Независимые продукты, доступные от фирмы *Information Builders, Inc.*

Драйверы *EDA/Data Drivers* отображают запросы *SQL* или *RPC*, сгенерированные приложением клиента, на тот язык, который используется на целевом источнике данных. Например, для *SQL*-запроса к базе данных *IMS* драйвер данных *IMS* сформирует последовательность вызовов на языке *DL/L*, и отошлет клиенту полученный ответ.

### **3.2. Возможности Enterprise Gateway**

*Enterprise Gateway* является процессом сервера баз данных

INFORMIX, который конвертирует запросы клиентов INFORMIX в запросы EDA/SQL.

Когда от клиентского приложения поступает инструкция SQL или удаленный вызов процедуры, предназначенный для Enterprise Gateway, то он просто перенаправляется на EDA/SQL Server, который затем обращается к соответствующим реляционным или нереляционным источникам данных. Ответы и данные, полученные от EDA/SQL Server, Enterprise Gateway возвращает приложению клиента.

### 3.2.1. Прозрачный доступ для чтения и записи

Enterprise Gateway представляет собой единый шлюз, который обеспечивает прозрачный доступ к данным в масштабах предприятия. Конечные пользователи обращаются к Enterprise Gateway так же, как к серверу баз данных INFORMIX. Доступ на чтение и запись осуществляется посредством стандартных инструкций SQL или удаленных вызовов процедур (RPC — Remote Procedure Call).

Для SQL поддерживаются оба стандарта синтаксиса — ANSI-92 SQL и ANSI-89 SQL; текущая версия EDA/SQL поддерживает синтаксис ANSI-89 SQL.

Доступ посредством RPC обеспечивается для инструментов разработки и приложений INFORMIX, а также третьих фирм. Удаленные вызовы процедур EDA/SQL выглядят как обращения к хранимым процедурам, поэтому для их использования в приложениях требуется внести лишь минимальные изменения. RPC позволяют выполнять операции чтения и записи и возвращать многострочные результаты.

Для обработки многострочных наборов данных, полученных в результате выполнения RPC или инструкции SQL, в Enterprise Gateway поддерживается механизм ролируемых курсоров (scroll cursors), который позволяет осуществлять прямой и обратный просмотр наборов данных.

### 3.2.2. Распределенные соединения

Enterprise Gateway может участвовать в распределенных соединениях,

координируемых сервером баз данных INFORMIX. Это позволяет импортировать в базы данных INFORMIX и/или интегрировать с ними данные из различных внешних источников.

### 3.2.3. Конфигурирование Enterprise Gateway

Enterprise Gateway прост в конфигурировании. Соединение клиента с Enterprise Gateway конфигурируется точно так же, как соединение между клиентской частью приложения INFORMIX и сервером INFORMIX-OnLine DS или INFORMIX-SE. Например, приложение под MS Windows, созданное инструментом разработки INFORMIX-NewEra, конфигурируется одинаково, независимо от того, обращается ли оно к серверу баз данных INFORMIX или к Enterprise Gateway.

Enterprise Gateway выполняется под управлением операционной системы UNIX и должен иметь доступ к EDA/SQL Server через сеть TCP/IP. Соединение Enterprise Gateway и EDA/SQL Server конфигурируется при помощи обычных конфигурационных файлов TCP/IP и конфигурационного файла EDA/Link.

### 3.2.4. Безопасность

Enterprise Gateway поддерживает централизованное управление пользовательскими идентификаторами (ID) и паролями, отображая их из среды INFORMIX в среду EDA/SQL. EDA/SQL Server обеспечивает безопасность путем взаимодействия с подсистемами безопасности соответствующих ОС. Например, в MVS осуществляется взаимодействие с подсистемами безопасности RACF, ACF2 и CA-Top Secret.

## 4. Библиотеки сопряжения сервера Informix-OnLine DS с менеджерами транзакций: Informix-TP/XA и Informix-TP/TOOLKIT

В состав инструментального продукта INFORMIX-ESQL/C входит библиотека C-программ INFORMIX-TP/XA. Эта библиотека обеспечивает для приложений, построенных при помощи INFORMIX-ESQL/C, сопряжение сервера

ра INFORMIX-OnLine DS с менеджерами транзакций, основанными на стандарте X/Open-XA, например, TUXEDO System/T. Аналогичную возможность обеспечивает библиотека 4GL-функций INFORMIX-TP/Toolkit для приложений на основе INFORMIX-4GL. Такое сопряжение позволяет организовать участие сервера INFORMIX в разнородных распределенных транзакциях с серверами баз данных других поставщиков, поддерживающих стандарт X/Open-XA, и использовать прочие преимущества, которые предоставляют современные менеджеры транзакций:

- Управление работой нескольких разнородных серверов от разных поставщиков. Менеджер транзакций выступает как координатор распределенных транзакций между подключенными к нему серверами, обеспечивая для их реализации протокол двухфазовой фиксации транзакций и механизмы восстановления.
- Перераспределение и баланс загрузки, максимально эффективное использование системных ресурсов.
- Масштабируемость и динамическое переконфигурирование прикладной среды с учетом изменяющихся потребностей.
- Обеспечение высокой доступности путем перенаправления запросов к дублирующим серверам в случае отказа.

## 5. Заключение

Если рассматривать создание и развитие информационной системы (ИС) как исторический процесс, то оценка СУБД как базиса для создания или развития ИС может проводиться по трем направлениям:

- Каковы перспективы ее использования в будущем?
- Допускает ли СУБД взаимодействие с унаследованными базами данных и компьютерными платформами?
- Каковы потребительские качества существующей версии СУБД?

Взаимодействие с унаследованными базами данных обеспечивает шлюз INFORMIX-Enterprise Gateway.

Продукты последней версии INFORMIX обладают высокими потребительскими качествами. Перечислим основные из них.

### Высокая производительность

Ее увеличению способствуют следующие свойства и оптимизирующие механизмы сервера INFORMIX-OnLine DS:

- Многопоточная архитектура
- Параллельная обработка
- Фрагментация таблиц и индексов
- Оптимизация выполнения запросов
- Разделяемая память
- Кэши словарей данных и хранимых процедур
- Собственное управление дисковой памятью
- Асинхронный ввод-вывод
- Опережающее чтение

Высокая производительность на приложениях OLTP, DSS, пакетных заданиях и их сочетаниях подтверждается тестами TPC (Transaction processing Performance Council), особенно на многопроцессорных платформах.

### Масштабируемость

Этим термином обозначается такое свойство сервера, которое обеспечивает при увеличении доступных вычислительных ресурсов (количества или быстродействия процессоров, числа дисков) соответствующее улучшение системных характеристик. Под улучшением системных характеристик понимается, например,

- рост числа обслуживаемых пользователей с сохранением среднего времени отклика;
- ускорение обработки одного запроса;
- сохранение того же времени обработки запроса при увеличении объема участвующих в нем таблиц.

Перечислим свойства и механизмы сервера, обеспечивающие масштабируемость:

- Многопоточная архитектура с поддержкой многопроцессорной обработки. Обслуживание клиентов равномерно распределяется между всеми наличными процессорами.
- Технология PDQ. Выполнение сложного запроса распределяется между всеми наличными процессорами. Результаты тестов показывают линейное ускорение обработки при увеличении числа процессоров.
- Фрагментация таблиц. Обработка больших таблиц ускоряется пропорционально числу фрагментов, располагаемых на разных дисковых устройствах.
- Гибкие средства наблюдения и настройки. Допускается динамическое изменение объема и конфигурации ресурсов, используемых сервером — числа виртуальных процессоров, дисковых пространств баз данных. В соответствии с наличием ресурсов и потребностями можно оперативно регулировать интенсивность параллельной обработки, изменять правила фрагментации таблиц.
- Поддержка распределенных транзакций. Производительность ИС может наращиваться путем распределения данных и их обработки между несколькими серверами, связанными сетью.

**Универсальность сервера**

Возможность смешанной загрузки его приложениями OLTP, DSS и пакетными заданиями, обеспечивается средствами параллельной обработки сложных запросов и средствами оперативной настройки, которые позволяют управлять балансом системных ресурсов между разными типами приложений.

Практическая осуществимость смешанной загрузки поддерживается также всеми механизмами, направленными на эффективное разделение ресурсов и повышение производительности, поскольку без этого невозможно проводить обработку трудоемких запросов, сохраняя приемлемое время отклика для приложений OLTP.

**Высокая доступность данных**

Данные становятся недоступны пользователям, если произошел программный или аппаратный сбой, а также если сервер остановлен с целью выполнения определенных административных действий. Сервер INFORMIX-OnLine DS обладает рядом возможностей, которые позволяют повысить надежность ИС и практически отказаться от плановых простоев:

- Зеркалирование дисковых областей
- Полное тиражирование данных сервера
- Развитые средства сохранения данных
- Восстановление не критичных для работы сервера данных в оперативном режиме
- Инструменты слежения за состоянием сервера
- Выполнение большинства административных задач, в том числе, настройки, в оперативном режиме
- Фрагментация таблиц (при отказе одного диска сохраняется частичная доступность таблицы)

**Функциональные возможности сервера**

Соответствуют входному уровню стандарта ANSI-92 SQL и включают, помимо рассмотренных выше, следующие средства:

- Хранимые процедуры
- Триггеры
- Курсоры
- Каскадные удаления данных
- Поддержка целостности, в том числе, целостности по ссылкам
- Уровни изоляции чтения:
  - грязное чтение
  - подтвержденное чтение
  - стабильное чтение
  - повторяемое чтение
- Поддержка больших бинарных объектов (BLOB)
- Поддержка оптических дисков

**Средства безопасности**

В сервере INFORMIX-OnLine DS эти средства соответствуют стандарту класса C2.

### Открытость

Это сложное понятие, включающее оценки по многим направлениям. Степень открытости определяет степень интегрируемости СУБД и продуктов, созданных на ее основе, в разнообразных аппаратных, программных, административных, национальных и др. средах, что чрезвычайно важно как для построения ИС в настоящем, так и для ее развития в будущем. Перечислим некоторые свойства, характеризующие открытость INFORMIX:

- доступность на множестве платформ, включая Sequent, HP, Sun, IBM, Siemens Nixdorf, NCR;
- поддержка, помимо UNIX, операционных систем Windows NT и NetWare;
- переносимость прикладных систем между платформами;
- возможность включения баз данных INFORMIX в распределенные разнородные ИС, построенные на основе аппаратно-программных платформ и СУБД разных производителей;
- интегрируемость INFORMIX с системами централизованного управления и администрирования, такими как Tivoli Management Environment (TME), HP OpenView, IBM NetView;
- поддержка национальных языков.

### Средства разработки

Средства разработки и средства доступа для конечного пользователя, в особенности, объектно-ориентированный инструмент групповой разработки прикладных систем с графическим интерфейсом INFORMIX-NewEra, оцениваются экспертами как высокоразвитые инструменты, отвечающие современным требованиям. Помимо этого INFORMIX поддерживается многими инструментальными системами независимых производителей.

С точки зрения развития информационной системы в будущем важны такие характеристики, как перспективность СУБД по применяемым методам и планируемые направления развития, поскольку от этого зависят возможности

развития ИС. Архитектурные и технологические решения сервера отвечают современным представлениям в этой области и постоянно совершенствуются. В ближайших версиях планируется:

- Развитие архитектуры сервера для обеспечения поддержки платформ MPP, а также слабосвязанных систем.
- Реализация дискретного тиражирования на уровне отдельных таблиц и других подмножеств данных.
- Создание интегрированных средств удаленного управления и администрирования совокупности серверов INFORMIX-OnLine DS на основе графического интерфейса, обладающих более изощренными возможностями наблюдения, обработки событий, планирования, управления базами данных и приложениями. Интеграция существующих административных утилит с инструментами системного и сетевого управления других производителей, основанных на промышленных и фактических стандартах.

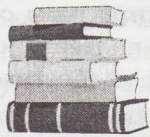
Существенное соображение при выборе продукта — стабильность, подтверждаемая общим стажем и "запасом лидерства" компании, т. е. общей долей рынка. Доля INFORMIX на мировом рынке СУБД — примерно 20%, в последние годы имеет тенденцию к росту.

Все это позволяет рассматривать INFORMIX как перспективную СУБД, которая может служить основой для построения развитых ИС.

### 6. Литература

1. Г. Г. Барон. Параллельные архитектуры серверов баз данных. Jet Info, Вып. 1, 1995.
2. INFORMIX-OnLine Dynamic Server. Administrator's Guide, Vol. 1,2. Version 7.1. 1994, INFORMIX SoftWare Inc.
3. INFORMIX-OnLine Dynamic Server 6.0, 7.1. Training Course, 1993, 1994, INFORMIX SoftWare Inc.
4. INFORMIX-OnLine Enterprise Gateway Version 7.1. Technical Brief. 1994, INFORMIX SoftWare Inc.

## ЧТО ЕЩЕ ПОЧИТАТЬ



### 1. Хендерсон Т. Подстелил бы Ethernet. — "LAN", русское издание, апрель 1995, том 1, номер 1, с. 32-37.

В статье сопоставляются две современные сетевые технологии — быстрый Ethernet и 100VG-AnyLAN, анализируются внутренне присущие им недостатки, способные затормозить их дальнейшее развитие. Подобный анализ всегда интересен, даже если приводимые автором аргументы вызывают возражения. Можно надеяться, что статья позволит читателям осуществить осознанный выбор между быстрым Ethernet и 100VG-AnyLAN.

### 2. Штейнке С. Ethernet-коммутиция. — "LAN", русское издание, апрель 1995, том 1, номер 1, с. 106-108.

Ethernet-коммутиция буквально окутана домыслами. В статье Стива Штейнке просто и доходчиво излагаются основные понятия, помогающие со знанием дела читать информацию различных фирм о предлагаемых ими коммутаторах. Важно и то, что читатель сможет понять, в каких случаях Ethernet-коммутиция является наиболее практичным способом расшивки узких мест локальной сети.

### 3. Розенблатт Б. UNIX RDBMS: следующее поколение. — "СУБД", 1, 1995, с. 6-23.

В статье рассматриваются тенденции развития реляционных СУБД на платформе UNIX — тема необычайно актуальная. Статья состоит из двух частей. В первой части анализируются

способы повышения производительности СУБД — сокращение числа конфликтов на сервере, использование параллелизма. Описываются средства построения распределенных баз данных. Внимание уделяется также такому ключевому вопросу современных информационных технологий, как интероперабельность. Во второй части автор акцентирует внимание на таких вопросах, как масштабируемость, типы данных, определенные пользователем, средства пользователя и разработчика. Чтение статьи позволит руководителям и техническим специалистам лучше понять, что предлагают и что предложат ведущие производители СУБД и на кого можно делать ставку.

### 4. Волков А. Тесты ТРС. — "СУБД", 2, 1995, с. 70-78.

Почти все любят ссылаться на результаты тестов (бенчмарков), но мало кто может внятно объяснить, что же собственно стоит за этими числами. В статье рассматриваются три теста — ТРС-А, ТРС-В и ТРС-С, служащие для измерения производительности СУБД на задачах оперативной обработки транзакций. Чтение статьи позволит со знанием дела сопоставлять информацию из рекламных и технических материалов различных фирм, осознанно принимать решение о выборе той или иной СУБД.

### 5. Меллинг В. Корпоративные информационные архитектуры: и все-таки они меняются. — "СУБД", 2, 1995, с. 45-59.

Автор — аналитик крупнейшей консультационной компании Gartner Group, рассматривает переход от мейнфреймов к открытым системам и связанные с этим тенденции развития информационных технологий. Формулируются опорные положения стратегического планирования на ближайшие пять лет. Представляется, что к мнению Gartner Group разумно прислушаться.

Статья весьма полезна для стратегически мыслящих руководителей.



### ЗАО "ДиалИТ"

Россия, 105023, г. Москва, ул. Краснобогатyrская, 44, офис 628

тел.: (095) 913-5169, факс: (095) 269-1955

e-mail: lena@dialit.msk.su

Храмцов Александр Владимирович

Пархомук Елена Ивановна

Решения в области системной интеграции и телекоммуникаций на базе продуктов Sun Microsystems, 3Com, Informix, Network General, услуг сети Спринт и др.

### Компания "Диасофт"

Россия, 129347, г. Москва, ул. Проходчиков, 9

тел.: (095) 183-0221, 182-2104, факс: (095) 183-3325

e-mail: root@diasoft.msk.su

Михалев Олег Владимирович

Глазков Александр Валерьевич

Генцис Александр Юрьевич

Любинский Александр Валерьевич

Комплексные системы автоматизации банковской деятельности: разработка, консалтинг, внедрение и сопровождение. Комплексные решения для автоматизации работы с корпоративными и частными клиентами. Технологии для автоматизации front- и back-офисов, решения для автоматизации учетных и управленческих функций.

### Фирма "ИнтерСвязь"

Россия, 344007, г. Ростов-на-Дону, ул. Обороны 107/25

т. (8632) 63-1360, 64-3088, 62-0562

e-mail: mike@icomm.rnd.su

Монастырский Михаил Иосифович

Иванов Александр Александрович

Шнурченко Юрий Анатольевич

Сопровождение регионального узла сети EUnet/Relcom. Системная интеграция. Сложные коммуникационные проекты. Разработка программного обеспечения.

### НПФ "ИнфоСистем"

Россия, 445042, г. Тольятти, ул. Ленина, 84

т. (8469) 22-3403

ф. (8469) 34-6740

e-mail: anor@is.tlt.ru

Орлов Анатолий Николаевич

#### Информационный бюллетень *Jet Info*

Зарегистрирован Комитетом РФ по печати 20 июля 1995 года, № 013951

Индекс по каталогу РОСПЕЧАТИ - 32555

**Главный редактор**

В.А.Галатенко

**Технический редактор**

С.И.Демочкин

103006, Москва

Краснопролетарская ул., 6

Тел.: (095) 972 11 82, 972 13 32

Факс:(095) 972 07 91

E-mail: info@jet.msk.su

© Jet Infosystems 1995

Полное или частичное воспроизведение материалов, содержащихся в настоящем издании, допускается только по согласованию с Jet Infosystems.

Перед Вами один из номеров бесплатного периодического информационно-технического бюллетеня "Jet Info", издаваемого компанией "Инфосистемы Джет" - ведущим российским системным интегратором в области UNIX-систем.

Если Вы еще не являетесь подписчиком "Jet Info", но хотели бы получать его регулярно, просим Вас отправить в наш адрес заполненный купон этого объявления. Отправьте этот купон и в том случае, если Вы сменили свой почтовый адрес или решили изменить форму доставки Вам бюллетеня.

Информация для подписчиков: не забудьте в течение каждого декабря и июля обязательно отправлять нам этот отрывной купон для перерегистрации.



1. Организация \_\_\_\_\_

2. Фамилия, имя, отчество и должность лица, в чей адрес будет осуществляться рассылка \_\_\_\_\_

3. Индекс и почтовый адрес \_\_\_\_\_

4. Телефон и/или факс \_\_\_\_\_

5. Адрес электронной почты \_\_\_\_\_

6. В какой форме Вы хотели бы получать бюллетень

печатный вариант по почте

электронную версию по электронной почте

7. Прежняя форма и адрес подписки (для подписчиков, меняющих адрес и форму подписки, либо тех, кто проходит перерегистрацию) \_\_\_\_\_

8. Общее число компьютеров в организации \_\_\_\_\_

9. Ваша организация использует

Вычислительную технику

- Apple
- DEC Alpha
- DEC VAX
- IBM PC
- HP 9000
- Silicon Graphics
- Sun Microsystems
- другое \_\_\_\_\_

Операционные системы

- MS-DOS (Windows)
- Novell NetWare
- SCO Unix
- Solaris / SunOS
- UnixWare
- VAX VMS
- другое \_\_\_\_\_

10. Материалы на какую тему заинтересовали бы Вас в первую очередь \_\_\_\_\_

Мы будем рады, если кто-то из Вас захочет принять участие в подготовке выпусков "Jet Info" в качестве автора. Будем также признательны за любые конструктивные замечания и предложения по всем аспектам подготовки, выпуска и распространения нашего бюллетеня.

Редакция "Jet Info"