

Распределенные информационные системы

1

Тема 1: Введение

МАКУНИН АЛЕКСЕЙ АНАТОЛЬЕВИЧ,
ДОЦЕНТ КАФЕДРЫ АОИ, К.Т.Н.

Структура курса. Лекции



- Распределенные системы: задачи, терминология принципы функционирования.
- Архитектура клиент-сервер. Типовые задачи. Области применения.
- Пример информационной системы (типичное приложение в архитектуре клиент-сервер).
- Многозвенная архитектура. Области применения. Краткий обзор современных технологий. XML, CGI/JSP, Servlets, DCOM, CORBA, RMI (.NET).
- Выделение слоев в многозвенной архитектуре (типичная архитектура). «Тонкие» и «Толстые» клиенты. Сервер приложения (Application server). Сервер базы данных (Database Server). Миграция объектов (вопросы распределения вычислительной нагрузки). Развертывание системы.
- Основы CORBA. CORBA и ООП. Язык определения интерфейсов IDL. Отображение IDL на C++. Отображение IDL на Java. ORB. Динамическое взаимодействие клиентов и серверов. Сервисы именования CORBA.
- Пример информационной системы, выполненной в многозвенной архитектуре.

Структура курса. Практика



Лабораторная работа 1

Система обслуживания дисконтных карт

Необходимый инструментарий: сервер - Oracle 8.1.7 (MSSQL Server 2000 sp3), клиент – Java (jdk, VisualCafe, MS J++, ...)

Лабораторная работа 2

WMS (Warehouse Management System) Тонкий клиент (Web, HandHeld, сотовый телефон, ...). Сервер приложения.

Взаимодействие клиент – сервер приложений. Сервер бизнес-логики. Вопросы распределения вычислительной нагрузки. Обеспечение отказоустойчивости.

Необходимый инструментарий: сервер - Oracle 8.1.7 (MSSQL Server 2000 sp3), Приложение/бизнес-логика – Java (jdk, VisualCafe, MS J++, ...)

Лекция 1



- Что такое распределенная информационная система?
- Зачем нужны распределенные информационные системы
- Какими они должны быть?
- Какие проблемы существуют при построении распределенных информационных систем?

Распределенные системы: определения



- “...система нескольких автономных вычислительных узлов, взаимодействующих для выполнения общей цели.”
- Автономные? А тонкие клиенты?
- Общая цель? А интернет?

Распределенные системы: определения



- Система, чьи компоненты размещены на различных узлах взаимодействующие и управляемые только посредством передачи сообщений.

Существуют системы с разделяемой памятью (или с разделяемым временем)

Распределенные системы: определения



- ”Система, состоящая из набора двух или более **независимых** узлов которые координируют свою работу посредством синхронного или асинхронного обмена **сообщениями**.”

Распределенные системы: определения



- “распределенная система это набор независимых узлов (компьютеров), которые представляются пользователю как один компьютер.” [Tanenbaum]
- “распределенная система это собрание независимых компьютеров соединенных сетью с программным обеспечением, обеспечивающим их совместное функционирование.”

Последствия...



● Параллельность

- Независимые процессы
 - Синхронизация
- Необходимость разделения ресурсов
 - Данные
 - Сервисы
 - Устройства
- Типичные проблемы
 - Deadlocks
 - Ненадежные коммуникации (проблема освобождения ресурсов)

Последствия...



- Нет “глобального” времени
 - Асинхронная передача сообщений -
 - Ограниченная точность синхронизации часов
- Нет состояния системы
 - Нет ни одного процесса в распределенной системе, который бы знал текущее глобальное состояние системы
 - Следствие параллелизма и механизма передачи данных

Последствия...



● Сбои

- Процессы выполняют автономно, изолированно
- Неудачи отдельных процессов могут остаться необнаруженными
- Отдельные процессы могут не подозревать о общесистемном сбое
- Сбои происходят чаще чем в централизованной системе
- Новые причины сбоев (которых не было в монолитных системах)
- Сетевые сбои изолируют процессы и фрагментируют систему

Принципы разделения



- **Функциональное разделение:** узлы выполняют различные задачи
 - Клиент / сервер
 - Хост / Терминал
 - Сборка данных/ обработка данных
 - Решение - создание разделяемых сервисов
- **Естественное разделение** (определяемое задачей)
 - Система обслуживания сети супермаркетов
 - Сеть для поддержки коллективной работы

Принципы разделения



- **Распределение нагрузки/балансировка:** назначение задачи на процессора так, чтобы оптимизировать общую загрузку системы.
- **Усиление мощности:** различные узлы работают над одной задачей
 - Распределенные системы содержащие набор микропроцессоров, по мощности могут приближаться к суперкомпьютеру

Принципы разделения



- **Физическое разделение:** система строится в предположении, что узлы физически разделены (требования к надежности, устойчивости к сбоям).
- **Экономические:** набор дешевых чипов может обеспечить лучшие показатели отношения цена/производительность, чем мэйнфрэйм
 - Мэйнфрэйм: 10 раз быстрее, 1000 раз дороже

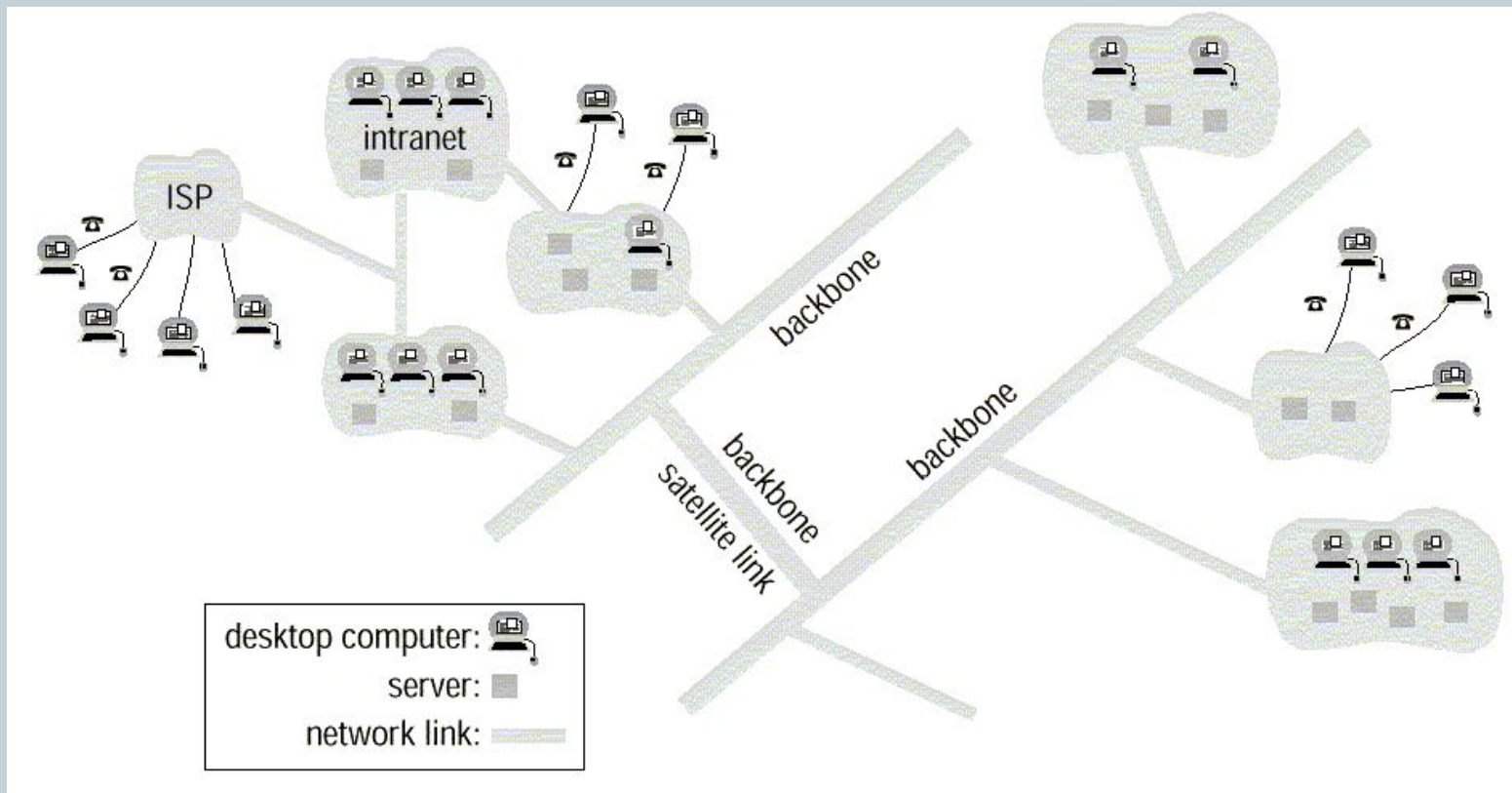
Примеры распределенных систем



- Internet (?)
- Intranet
- Вычислительные кластеры
- ...

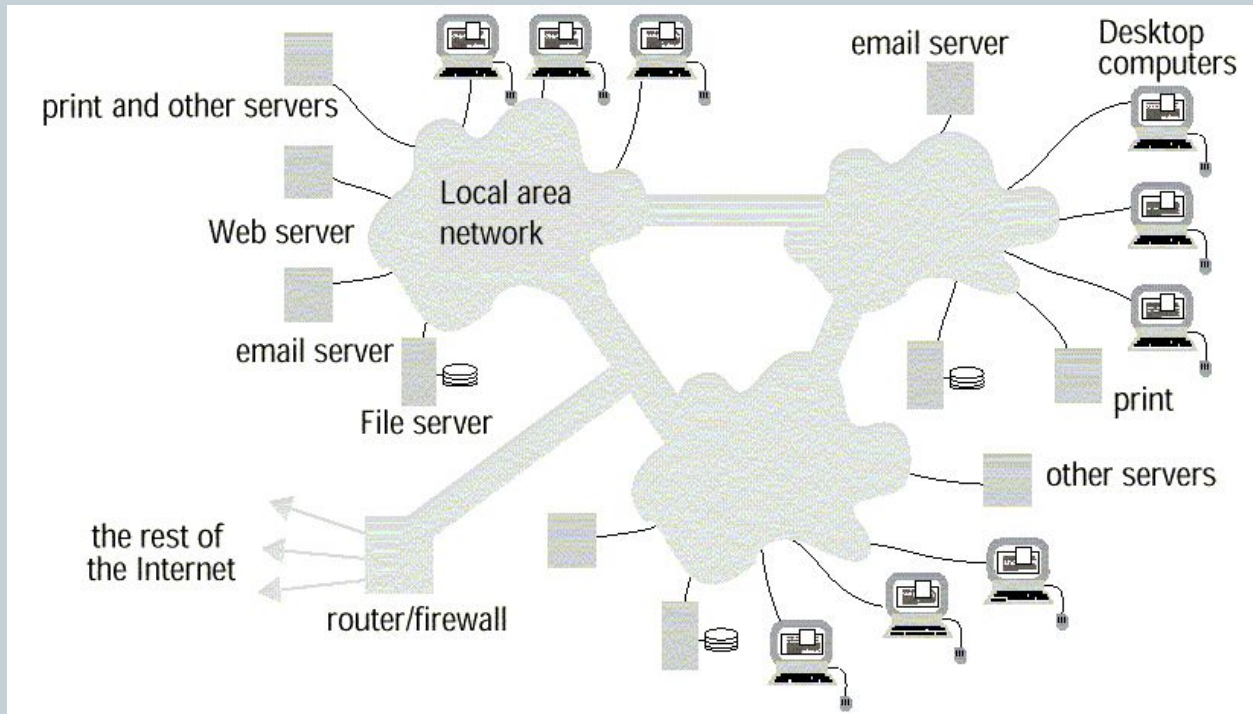
Пример: Internet

- Гетерогенная сеть компьютеров и приложений
- Реализация взаимодействия - IP стек



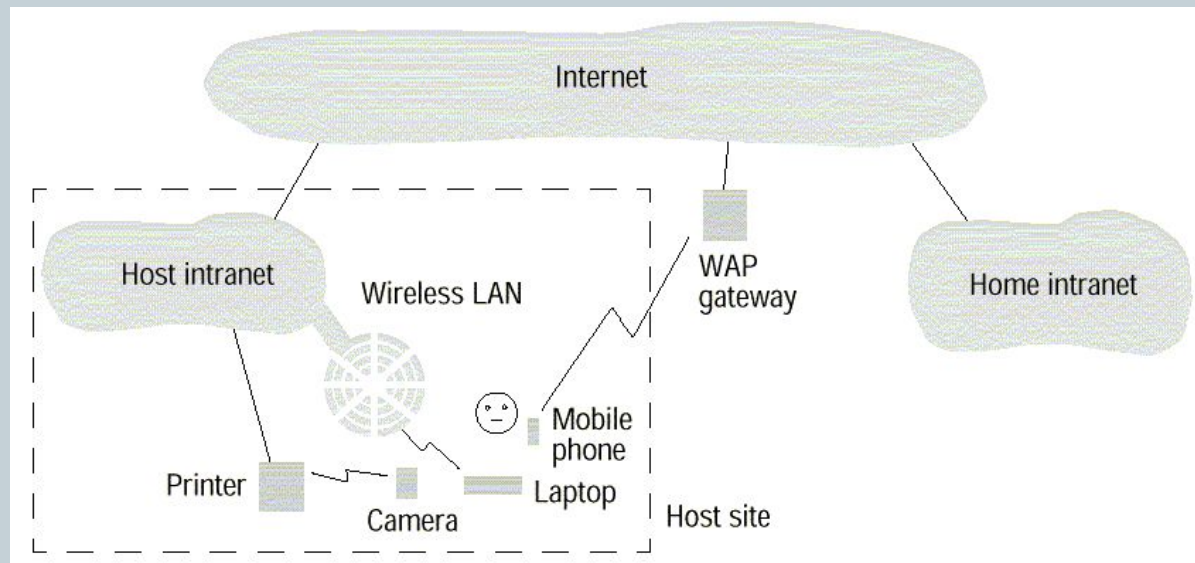
Пример: Intranet

- Администрируется локально
- Взаимодействие с Internet
- Обеспечивает сервисами (внутренних и внешних пользователей)



Пример: Wireless Information Devices

- Система сотовой связи (GSM)
 - Ресурсы разделяемы (радио частота, время передачи на частоте,...)
- Laptop (подключаются к Wireless LAN)
- Handheld, PDAs etc.



Другие примеры



- Системы управления аэропортом
- Автомобильные управляющие системы
Mercedes S класса сегодня имеет более 50 автономных встроенных процессоров соединенных общей шиной



Примеры...



- Телефонные системы
- Сложные сети предприятий
- Сетевые файловые системы
- WWW
- и многое другое...

Разделение ресурсов



- Разделение ресурсов часто является одной из причин разработки распределенной системы
 - Уменьшается стоимость, (file и print сервера)
 - Разделение данных между пользователями (совместная работа над проектом)
- Сервисы
 - Управляют набором **ресурсов**
 - Представляют **услуги** пользователям

Разделение ресурсов

- **Сервер** используется для предоставления сервисов
 - Принимает запросы на обслуживание от клиентов
 - вызов операции
 - Прием сообщения/ответ на сообщение
 - полная реализация - удаленный вызов
 - Роли клиента и сервера меняются от вызова к вызову
 - один и тот же процесс может быть как клиентом, так и сервером
 - Терминология Клиент/Сервер применяется к процессам, а не к узлам!!!

Проблемы

- Распространение приложения
- Гетерогенность
- Открытость
- Безопасность
- Масштабируемость
- Обработка ошибок и восстановление после сбоев
- Параллелизм
- Прозрачность
- Управляемость

Распространение приложения



- **Фрагментация**
 - разделение приложения на модули для распространения
- **Конфигурация**
 - Связь модулей друг с другом (зависимости)
- **Размещение**
 - выгрузка модулей на целевую систему
 - Распределение вычислительных модулей между узлами (статическое или динамическое)

Гетерогенность



- Гетерогенные = разные
- Различные
 - сетевые инфраструктуры,
 - hardware&software (пример Intel & Motorola, UNIX sockets & Winsock calls),
 - языки программирования (и представления данных!!!)
- Различия должны быть скрыты

Гетерогенность



- Интерфейсы и реализация могут быть разными
 - Базовые концепции обычно неизменны
- Необходимы стандарты

Гетерогенность



- **Middleware:** промежуточный программный слой
 - позволяет гетерогенным узлам взаимодействовать
 - Определяет однородную вычислительную модель
 - Поддерживает один или несколько языков программирования
 - Обеспечивает поддержку распределенных приложений
 - Вызов удаленных объектов
 - Удаленный вызов SQL
 - Распределенная обработка транзакций
- Примеры: CORBA, Java RMI, Microsoft DCOM

Гетерогенность



- Мобильный код: код разработан для миграции между узлами
 - Необходимо преодолевать аппаратные различия (разные наборы инструкций)
- Виртуальные машины
 - Компилятор «изготавливает» байт-код для VM
 - VM реализована для всех аппаратных платформ (Java)
- Методы грубой силы
 - Портируем код под каждую платформу...

Открытость

- Гарантирует расширяемость
- Возможность повторного использования
- Важные факторы:
 - Наличие четких спецификаций
 - Наличие полной документации
 - Опубликованные интерфейсы
 - Тестирование и проверка на многих платформах



Безопасность

- Три компонента:
 - Защищенность
 - Целостность
 - Доступность
- Задача: посылка значимой информации по сети безопасно и эффективно

Безопасность

- **Сценарий 1: Доступ к результатам тестирования по NFS**
 - Откуда мы знаем, что пользователь - преподаватель, имеющий доступ к данным?
 - Авторизация
- **Сценарий 2: Посылка номера кредитной карты в интернет-магазин**
 - Никто кроме получателя не должен прочитать данные
 - Криптография

Безопасность

- **Нерешенные проблемы**
 - Атаки типа DoS (отказы в обслуживании)
 - Безопасность мобильного кода
 - ▢ Непредсказуемые эффекты
 - ▢ Может вести себя подобно троянскому коню...

Масштабируемость

- Распределенная система масштабируема, если она остается эффективной при увеличении числа обслуживаемых пользователей или ресурсов
- Проблемы:
 - Контроль стоимости ресурсов
 - Контроль потерь производительности

Масштабируемость

- Стоимость физических ресурсов
 - Растет, при увеличении числа пользователей
 - Не должна расти быстрее, чем $O(n)$, где n = количеству пользователей
- Потери производительности
 - Увеличиваются с ростом размера данных (и количества пользователей)
 - Время поиска не должно расти быстрее, чем $O(\log n)$, где n = размер данных

Масштабируемость

- Существуют естественные ограничения
 - Некоторые определяются легко
 - Другие труднее
- Обход узких мест
 - Децентрализация алгоритмов
 - ▣ Пример - Domain Name Service
 - Тиражирование и кэширование данных

Обработка сбоев



- Сбои более частые, чем в централизованных системах, но обычно локальные
- Обработка сбоев включает
 - Определение факта сбоя (может быть невозможно)
 - Маскирование
 - Восстановление

Обработка сбоев



- **Диагностика**

- Может быть возможна (ошибки передачи - контрольная сумма)
- Может быть невозможна (удаленный сервер не работает или просто очень загружен?)

Обработка сбоев



- Маскирование

- Многие сбои могут быть скрыты
- Может быть невозможно (все диски повреждены)
- Не всегда хорошо

Параллелизм



- **Контроль параллелизма**
 - Обращение нескольких потоков к ресурсу
 - ▣ Правильное планирование доступа в параллельных потоках (устранение взаимных исключений, транзакции)
 - Синхронизация (семафоры)
 - ▣ Безопасно, но уменьшают производительность
 - Разделяемые объекты(ресурсы) должны работать корректно в многопоточной среде

Прозрачность



- Соккрытие гетерогенной и распределенной структуры системы так, чтобы пользователю система представлялась монолитной

Прозрачность

Прозрачность доступа: доступ к локальным и удаленным ресурсам посредством одинаковых вызовов

Прозрачность расположения: доступ к ресурсам вне зависимости от их физического расположения

Прозрачность параллелизма: возможность нескольким процессам параллельно работать с ресурсами, не оказывая влияния друг на друга

Прозрачность репликации: возможность нескольким экземплярам одного ресурса использоваться без знания физических особенностей репликации.

Прозрачность обработки ошибок: Защита программных компонентов от сбоев, произошедших в других программных компонентах.

Восстановление после сбоев

Прозрачность мобильности: Возможность переноса приложения между платформами, без его переделки

Прозрачность производительности: возможность конфигурации системы с целью увеличения производительности при изменении состава платформы выполнения

Прозрачность масштабируемости: возможность увеличения производительности без изменения структуры программной системы и используемых алгоритмов

Прозрачность



- Очень важна для распределенных систем
 - Прозрачность доступа и физического расположения
 - Имеет критическое значения для должного использования распределенных ресурсов

Управляемость



- Распределенные ресурсы не имеют центральной точки управления
- Локальная оптимизация не всегда означает глобальную оптимизацию
 - Нужен глобальный взгляд на проблему
 - Не всегда возможен (есть системы, никому конкретно не принадлежащие)

Итоги

- Распределенная система:
 - Автономные (но соединенные средой передачи данных) узлы
 - Взаимодействие посредством передачи сообщений
- Много примеров того, что распределенные системы нужны и их нужно уметь строить
- Распределенные системы существуют и их нужно уметь развивать и поддерживать

Модели архитектуры



- Модель архитектуры распределенной системы должна содержать решение двух проблем:
 - Физическое размещение компонентов между узлами
 - Взаимодействие между компонентами.

Уровни



- Приложения, сервисы
- Middleware
- Операционная система
- Аппаратура

Возможные архитектуры



- Клиент - сервер
- Модель предоставления услуг пулом серверов
- Модель прокси – и кэш - серверов
- Модель равных процессов

Вариации на тему Клиент-сервер



- Мобильный код
- Мобильные агенты
- Network computers
- Тонкие клиенты
- X - window

Требования к дизайну



- Требования, накладываемые обеспечением требуемой производительности
- Использование кэширования и репликации
- Требование надежности

Требования к производительности



- Время отклика
- Производительность.
- Балансировка нагрузки

Использование кэширования и репликации



- Очень многие проблемы производительности системы могут быть решены путем кэширования данных .

Модели



- Модель взаимодействия
- Модель защиты от сбоев
- Модель безопасности

Модель взаимодействия



- Производительность линий связи
- Время и события
- Асинхронный и синхронный обмен

Спасибо за внимание!