Распределенные вычислительные системы

Лекция №4: Стандарты OMG, Архитектура CORBA, ORB

Алексей В. Бурдаков, к.т.н.
burdakov@usa.net

План лекций

№	Дата	Тема
1	13.09	Вводная лекция: распределенные системы
2	20.09	Эволюция распределенных технологий
3	27.09	Метамодель системы, принципы ПО среднего слоя
4	04.10	Стандарты OMG, Архитектура CORBA, ORB
5	11.10	Пример приложения на CORBA
6	18.10	РК1
7	25.10	COM, Java/RMI
8	01.11	Определение местонахождения распределенных объектов
9	15.11	Долговременное хранение распределенных объектов
10	22.11	Распределенные объектные транзакции
11	29.11	Безопасность
12	06.12	Расширенное взаимодействие
13	13.12	РК2
14	20.12	Зачет/Экзамен

План лекции

- OMG, Стандарты
- Object Management Architecture (OMA)
- Объектная модель OMA/CORBA и язык определения интерфейсов IDL
- Брокер объектных запросов

4-3

Реализации ORB

- Некоммерческие:
  - ACE ORB (TAO) – Washington University
  - ORBit
  - OmniORB2 – AT&T Laboratories Cambridge
  - ROBIN – Fermi National Accelerator Laboratory
- Коммерческие:
  - OAK
  - OrbixWeb
  - CorbaPlus for C++ - Expertsoft
  - Visibroker – Inprise (Borland + Visigenic)
  - Orbacus (OmniBroker)

http://adams.patriot.net/~tvalesky/freecorba.html

4-4

Object Management Group

- Факты о OMG:
  - Основана в апреле 1989
  - Открытое членство (более 1000 организаций: IBM, Oracle, Boeing, ...)
  - На сегодня самая большая организация по стандартизации
  - Неприбыльная организация
- Цель: **Создание и развитие стандартов в компьютерной индустрии**
- Интероперабельные приложения масштаба предприятия

4-5

Стандарты OMG

- Стандарты на ПО среднего слоя и его архитектуру
  - **OMA**: Object Management Architecture
  - **CORBA**: Common Object Request Broker Architecture
- Стандарты моделирования и определения метаданных
  - **MOF**: Meta Object Facility
  - **MDA**: Model Driven Architecture
  - **UML**: Unified Modeling Language
  - **XMI**: XML Metadata Interchange
  - **CWM**: Common Warehouse Metamodel

4-6

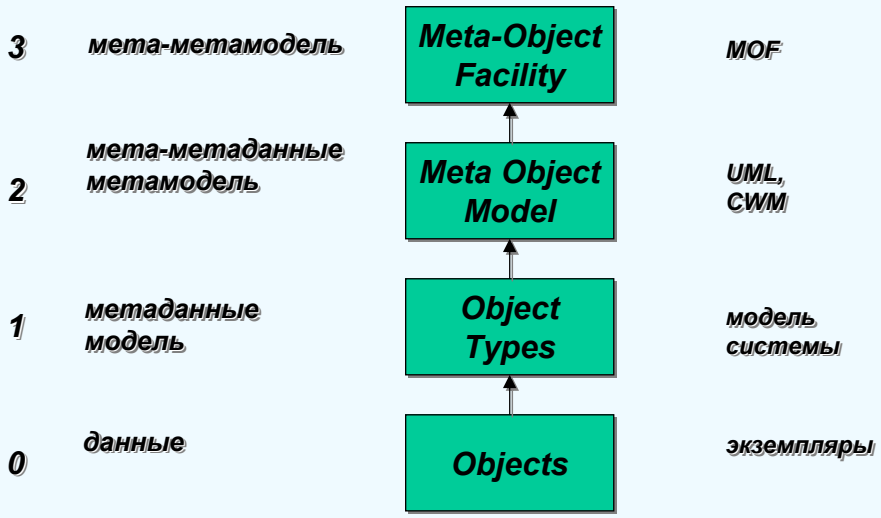
Статус стандартов OMG

- Стандарты OMG приняты International Standardization Organization / International Electrotechnical Commission

• CORBA:	ISO/IEC 19500-2
• OMG IDL:	ISO/IEC 14750:1999
• Object Trader Service	ISO/IEC 13235:1998
• MOF:	ISO/IEC 14769
• UML:	ISO/IEC DIS 19501-1

4-7

Различные уровни абстракции систем



4-8

MOF: Meta Object Facility

- Две части стандарта:
  - Обобщенное средство для представления метамоделей (средств описания различных моделей)
  - Репозитарий для моделей и метамоделей
- Преимущества от использования:
  - Модели разных метамоделей основанных на MOF могут храниться в одном репозитарии
  - UML диаграммы и CWM основаны на MOF
- Для определения MOF использован UML, OCL (Object Constraint Language) и вербальное описание
- Проходит стадию обсуждения на стандарт ISO/IEC 14769

4-9

CWM: Common Warehouse Metamodel

- Стандартизирует метамодели данных для баз и хранилищ данных
- Определяет метамодели для реляционных и многомерных данных, OLAP и data mining, и т.п.
- Позволяет решить проблему обмена данными между различными базами и хранилищами данных

4-10

UML: Unified Modelling Language

- Графический язык, стандартным образом выражающий анализ требований и проект прикладной системы

- История создания:
  - Множество методов представления объектно-оринетированного анализа и проектирования в начале 90-х
    - •Booch'es Grady Booch
    - •OOSE Ivar Jacobson
    - •OMT Jim Rambaugh
  - Компания Rational Software объединила различные языки проектирования в UML
- Будучи независимым от методологии, используется большим числом средств анализа и проектирования

4-11

UML: Unified Modelling Language

- Структурные диаграммы
  - Классов
  - Объектов
  - Компонентов
  - Распределения
- Поведенческие диаграммы
  - Варианты использования
  - Последовательности
  - Взаимодействий
  - Состояний
  - Действий
- Диаграммы управления моделями
  - Пакетов
  - Моделей
  - Подсистем

4-12

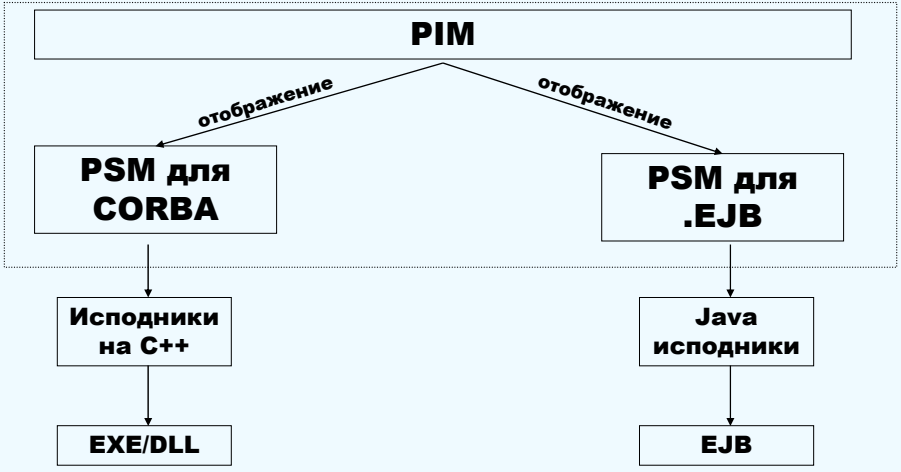
XMI: XML Metadata Interchange

- XMI – потоковый формат для обмена MOF-совместимыми метаданными включая UML-модели, создаваемые на этапе анализа и проектирования
- OMG использовал XML в качестве основы для языка обмена метамоделями
- XMI это отображение MOF в XML
  - Отображение UML в XML
  - Отображение CWM в XML

MDA: Model Driven Architecture

- Много альтернативных платформ распределенных систем:
  - J2EE (Java 2 Enterprise Edition)
  - Компонентная модель CORBA
  - .NET
- Недостатки палтформенной зависимости
  - Перспективы лидерства платформы не ясны
  - Опасность смешивания бизнес-логики и платформо-зависимого кода
  - Ведет к непереносимым компонентам
  - На серверной стороне: бизнес-логика «переживает» платформы

MDA: Model Driven Architecture



4-15

MDA: Model Driven Architecture

- Платформо-независимый способ описания спецификаций и разработки прикладных систем
- Платформо-независимая модель (PIM)
- Отображение в ряд платформо-зависимых моделей (PSM) (например, CORBA, .NET - XML/SOAP, …)
- MDA нельзя сравнивать с существующими Middleware, т.к. это модель, описывающая общие независимые от конкретных Middleware принципы построения распределенных систем

4-16

OMA и CORBA

- OMA (Object Management Architecture)
  - Определяет основные архитектурные компоненты распределенной неоднородной объектной среды, их интерфейсы и протоколы взаимодействия.
- CORBA (Common Request Broker Architecture)
  - Определение спецификации архитектуры и инфраструктуры для создания распределенных объектных систем, основанной на архитектурной концепции промежуточного слоя (брокера объектных заявок).

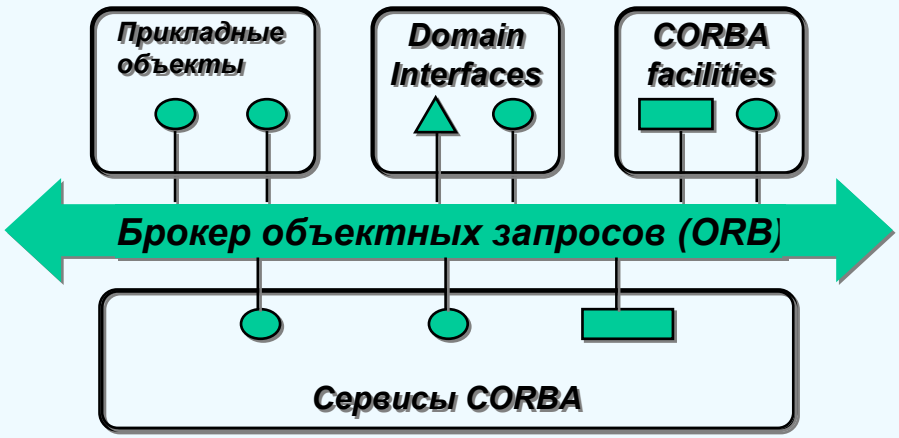
4-17

План лекции

- OMG, Стандарты
- Object Management Architecture (OMA)
- Объектная модель OMA/CORBA и язык определения интерфейсов IDL
- Брокер объектных запросов

4-18

Object Management Architecture



Спецификация OMA определяет эталонную модель распределенных объектов и делит из на различные категории

4-19

Брокер объектных запросов (ORB)

- Коммуникационное ядро, шина, обеспечивающая взаимодействие между объектами и клиентами
- Независимо от:
  - их расположения в сети
  - аппаратной платформы
  - программной платформы
  - языка программирования

4-20

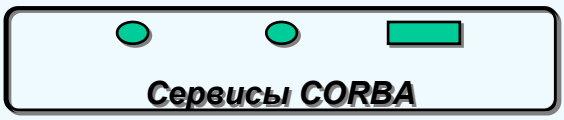
Прикладные объекты

- Объекты, разработанные под конкретные задачи
- Стандартизация не нужна

4-21

Сервисы CORBA (CORBA Services)

- Независимые от предметной области интерфейсы к сервисам, реализующим базовую функциональность и используемые большинством распределенных объектных приложений
- Включает функции:
  - именования объектов
  - управления доступом
  - поддержания непротиворечивых ссылок между объектами в распределенной среде
  - совместного доступа
  - транзакций
  - лицензирования
  - и др.



4-22

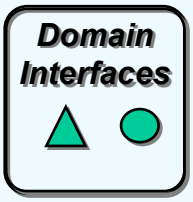
Общие (горизонтальные) средства (Horizontal/Common Facilities)

- Расположены между сервисами и прикладными объектами
- Являются потенциально применимыми вне зависимости от прикладной области
- В отличии от Сервисов CORBA ориентированны на приложения конечных пользователей
  - Distributed Document Component Facility - средства компонентов для распределенных документов
  - интернационализация

4-23

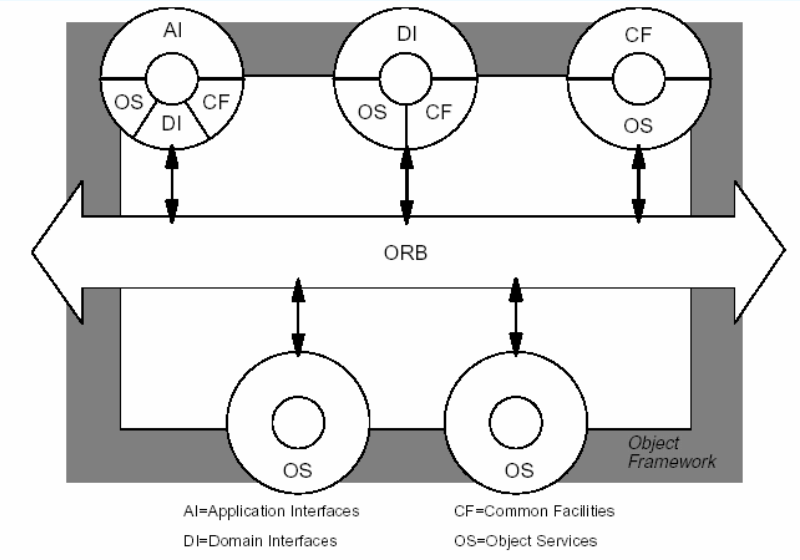
Интерфейсы предметной области (Vertical/Domain Interfaces)

- Domain Interaces = Vertical Facilities
- Компоненты, реализующие решения проблем бизнеса в определенных «вертикальных» сегментах рынка:
  - финансы
  - здравоохранение
  - производство
  - и т.д.



4-24

OMA: Использование интерфейсов



4-25

План лекции

- OMG, Стандарты
- Object Management Architecture (OMA)
- Объектная модель OMA/CORBA и язык определения интерфейсов IDL
- Брокер объектных запросов

4-26

Язык определения интерфейсов OMG IDL

- Описывает интерфейсы объектов CORBA
- Язык для выражения всех концепций объектной модели CORBA
- OMG/IDL:
  - независим от языков программирования
  - ориентирован на концепции C++
  - вычислительно не полон (декларативный язык)
- Определены связывания с различными языками
- Для совместимости с другими языками IDL развивается по принципу «чем проще, тем лучше»

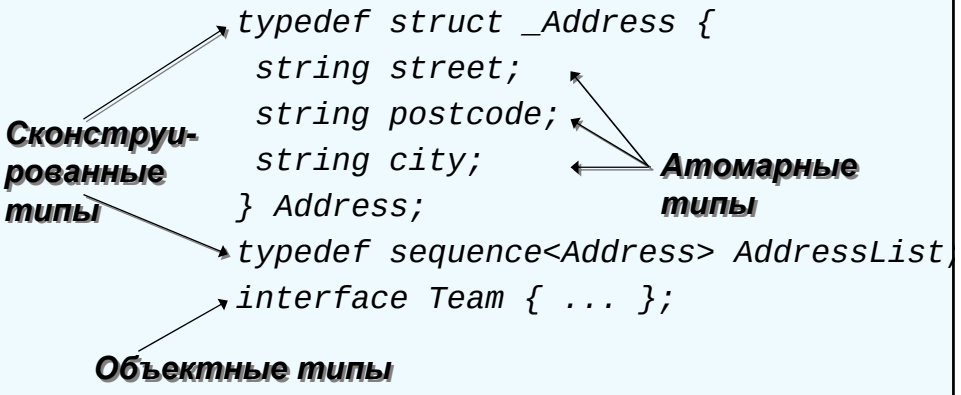
4-27

Объекты (Objects)

- Предоставляет один или более сервисов (операций)
- Каждый объект имеет один уникальный в пределах ORB идентификатор
- Множество ссылок на объект
- Ссылки:
  - поддерживают прозрачность расположения
  - создаются и управляются ORB
  - постоянны (persistent)
- Объекты создаются и уничтожаются вследствие запросов

4-28

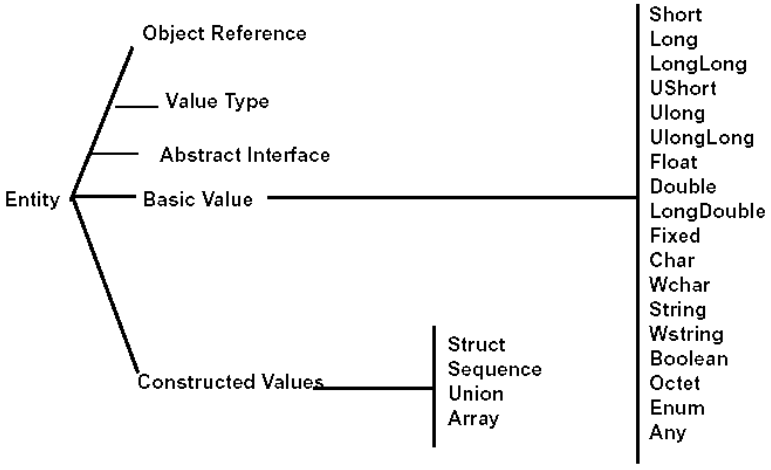
Типы (Types)



Тип (type) – идентифицируемая сущность с ассоциированным предикатом (математическая функция одного аргумента с булевым результатом), определенным на множестве объектов. Сущность соответствует типу если предикат для данной сущности определяет значение TRUE.

4-29

Типы (Types) продолжение



4-30

Типы (Types) прод.

- Value Types:
  - введены в версии 2.3
  - технология заимствована из RMI
  - позволяет передавать по значению любой объект
  - Value Types названы как «объекты с состоянием» (stateful)
- Абстрактный интерфейс
  - концепция схожа с абстрактным классом в C++
  - на его основе могут быть созданы другие интерфейсы или типы-значения

4-31

Модули (Modules)

```
module Soccer {
 typedef struct _Address {
  string street;
  string postcode;
  string city;
 } Address;                    <-- Soccer::Address
};
module People {
 typedef struct _Address {
  string flat_number;
  string street;
  string postcode;
  string city;
  string country;
 } Address;                    <-- People::Address
};
```

Модули

- Области видимости

4-32

Атрибуты (Attributes)

```
interface Player;
typedef sequence<Player> PlayerList;
interface Trainer;
typedef sequence<Trainer> TrainerList;
interface Team {
  readonly attribute string name;
  attribute TrainerList coached_by;
  attribute Club belongs_to;
  attribute PlayerList players;
  ...
};
```

Не изменяемое значение

Изменяемое

Тип атрибута

Имя атрибута

4-33

Операции (Operations)

```
interface Team {
  ...
  void bookGoalies(in Date d);
  string print();
};
```

Вид параметра in, out, inout

Список параметров

Типы возвращаемых значений

Тип параметра

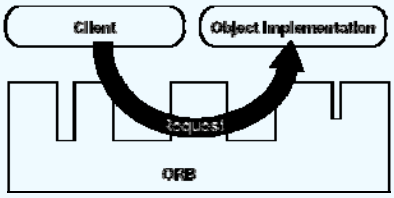
Имя параметра

[oneway] <op_type_spec><identifier>(param1,…,paramL)
[raises(except1,…exceptN)][context(name1,…nameM)]

4-34

Запросы (Requests)

- Запросы определяются клиентскими объектами
- Состав запроса:
  - Ссылка на серверный объект
  - Имя запрашиваемой операции
  - Запрошенные параметры
  - Информация о контексте (по транзакции, по безопасности)
- Запрос выполняется синхронно
- Запрос может быть определен:
  - статически
  - динамически



4-35

Исключения (Exceptions)

- Общие исключения (разрыв связи, неправильная ссылка, отсутствие свободной памяти, и т.п. - около 25)
- Определенные пользователем

Имя исключения *Дата*

```
exception PlayerBooked{sequence<Date> free;};
interface Team {
 ...
 void bookGoalies(in Date d) raises(PlayerBooked)
};
```

Операции декларируют исключения

4-36

Подтипы, наследование (Inheritance)

Неявный супертип: `Object`

Наследуется типом `Club`

Супертип

```
interface Organization {
  readonly attribute string name;
 };
interface Club : Organization {
  exception NotInClub{};
  readonly attribute short noOfMembers;
  readonly attribute Address location;
  attribute TeamList teams;
  attribute TrainerList trainers;
  void transfer(in Player p) raises NotInClub;
 };
```

4-37

Подтипы, наследование (Inheritance)

- Интерфейсы могут наследоваться (поддержка множественных интерфейсов)
- Множественное наследование
- Противоречия должны разрешаться компилятором IDL
- Общий корень - Object
- Наследование делает систему открытой для расширения, но закрытой для модификаций (так называемый принцип «Открытости-закрытости»)
- Поддержка полиморфизма на уровне объектов

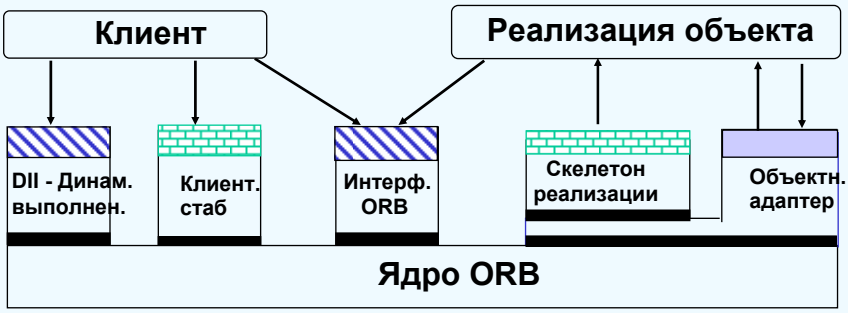
4-38

План лекции

- OMG, Стандарты
- Object Management Architecture (OMA)
- Объектная модель OMA/CORBA и язык определения интерфейсов IDL
- Основные элементы архитектуры CORBA

4-39

Основные элементы архитектуры CORBA



Стандартный интерфейс
Один интерфейс на каждую операцию
Один интерфейс на каждый адаптер
Интерфейс зависимый от ORB

4-40

Ядро ORB

- Обеспечивает доставку запроса к серверу и ответа клиенту
- Обеспечиваемая прозрачность:
  - расположения объекта
  - доступа
  - реализации объекта (язык, программно-аппаратная платформа)
  - состояния объекта (активен/не активен)
  - коммуникации (скрытие протокола TCP/IP, разделенная область памяти, локальный вызов, …)
- Создание как можно более простого ядра ORB, передача основной функциональности другим компонентам OMA (Сервисы, общие средства,…)

4-41

Интерфейс ORB

- ORB может быть реализован различными способами
- Для разделения приложения от деталей реализации CORBA определяет абстрактный интерфейс для ORB
  - объектные ссылки: *бинарная <> строковая формы*
  - создание списка аргументов для динамического вызова

4-42

IDL Стабы (Stubs)

- Механизм, создающий и осуществляющий запросы от лица клиента
- Зависит от конкретного языка реализации
- «Статические» запросы
- Представляет CORBA объект как обычную сущность языка
- Совместно с ORB выполняет *marshaling*

```
// C++
Factory_var factory_objref;
// Initialize factory_objref using Naming or
// Trading Service (not shown), then issue request
Object_var objref = factory_objref->create();
```

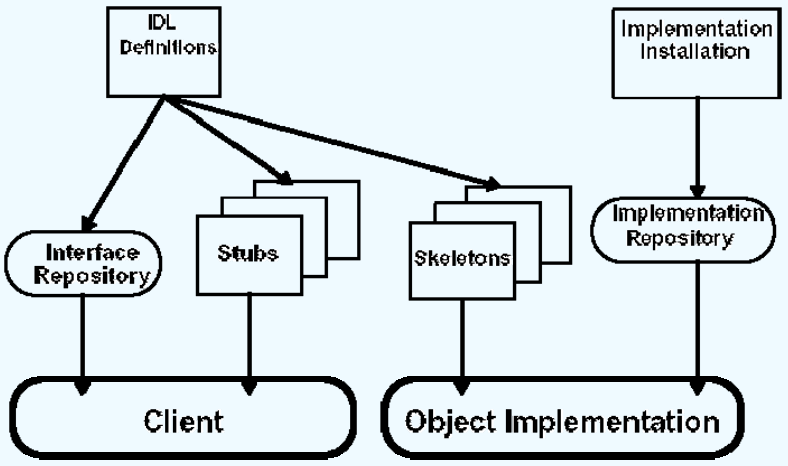
4-43

IDL Скелетоны (Skeletons)

- Сущность языка программирования, соединяющая OA с его реализацией (servant)
- Реализация:
  - язык C: коллекция указателей на функции
  - язык C++: базовый класс от которого наследуются классы реализаций (servants)
- Зависит от конкретного языка реализации
- «Статические» запросы
- Совместно с ORB выполняет *unmarshaling*
- Для динамических поддерживается специальный интерфейс (DSI)

4-44

Компиляция определений интерфейсов IDL



4-45

Объектный адаптер (Object Adapter)

- «Объектный адаптер - компонент CORBA, отвечающий за включение концепции объектов CORBA в концепцию реализаций объектов, принятую в языках программирования» [3]
- Объекты CORBA *абстрактны*, а их реализации - *конкретны*
- Отвечает за *передачу вызова* и *инициализацию*
- Плюсы концепции объектного адаптера:
  - поддержка различных стилей реализаций (servants), например объекты в БД, real-time OA и т.п.
  - облегчение ядра ORB

4-46

Объект (Object)

- Абстрактная или «виртуальная» сущность
- Обладает *интерфейсом*
- Для него определена соответствующая *реализация (Servant)*
- Находится, адресуется и вызывается по объектной ссылке (object reference)

4-47

Клиент (Client)

- Программная сущность, вызывающая операции реализации объекта
- Прозрачность вызова объекта

```
// C++
object->op(arguments)
```

4-48

Реализация объекта (Servant)

- Сущность на языке программирования существующая в контексте сервера и реализующая объект CORBA
- Реализация:
  - Не ООЯ (например, C): набор функций, манипулирующих данными
  - ООЯ (например, C++, Java): экземпляр класса
- Не обязательно содержит состояние (хранение в БД)

4-49

Динамический интерфейс вызова (DII)

- Прямой доступ клиента к механизму вызовов ORB
- Динамический вызов без *a-priori* существующих IDL стабов
- DII позволяет также осуществлять следующий типы вызовов:
  - отложенный синхронный (deferred synchronous)
  - oneway (только посылка)

4-50

Динамический интерфейс скелетона (DSI)

- Аналог DII на сервере
- Вызов реализаций (Servants) не имеющих на момент компиляции знания о типе реализуемого объекта
- Клиент не имеет представления о том, что сервер использует DSI или скелетон IDL

4-51

Литература / Internet источники

- В. Эммерих ***Конструирование распределенных объектов***. - М.:Мир. - 2002.
- Vinoksi S. ***CORBA: Integrating Diverse Applications Within Distributed Heterogeneous Environments****, IEEE'96.*
- Schmidt D.C. And Vinoski S. ***Object Adapters: Concepts and Technology***, SIGS C++ Report, Vol. 9, No. 11, Nov-Dec 1997.
- Ю.А. Григорьев, А.Д. Плутенко. ***Жизненный цикл проектирования распределенных баз данных***. - Благовещенск. - 1999.
- www.omg.org

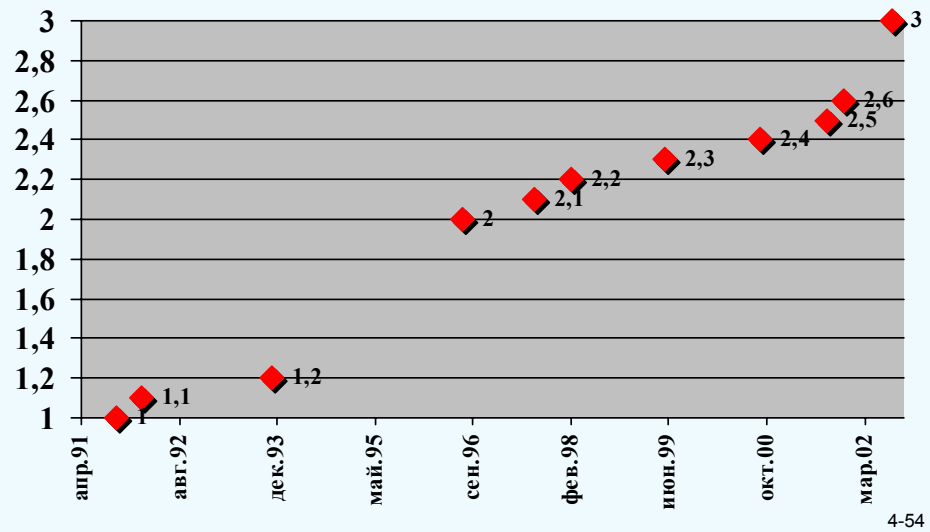
4-52

CORBA: цели

- Поддержка распределенных и гетерогенных запросов прозрачным для пользователей и программистов способом
- Обеспечение интеграции новых компонентов со старыми
- Открытый и свободно распространяемый стандарт
- Основой является консенсус большинства «игроков» индустрии

4-53

Развитие стандарта CORBA



4-54

CORBA 1.0

- Октябрь 1991
- Объектная модель CORBA
- Язык определения интерфейсов IDL
- Ядро API для DII и Interface Repository
- Отображение только на язык C

4-55

CORBA 2.0

- Август 1996
- Расширения стандарта:
  - Интерфейс динамического скелетона
  - Начальное разрешение ссылки
  - Расширения к Interface Repository
  - Интероперабельность (GIOP, IIOP, DCE CIOP)
  - Поддержка сервисов транзакций и безопасности
  - Взаимодействие с OLE2/COM

4-56

CORBA 3.0

- Сентябрь 2002
- Интеграция с Java и Internet
- Управление качеством обслуживания
- Компонентная архитектура CORBA

4-57

Объектная модель

- Определена в OMA Guide и CORBA
  - Объекты (objects)
  - Типы (types)
  - Модули (modules)
  - Атрибуты (attributes)
  - Операции (operations)
  - Запросы (requests)
  - Исключительные ситуации (exceptions)
  - Подтипы (subtypes)

4-58