

Распределенные вычислительные системы

Лекция №4: Стандарты OMG, Архитектура CORBA, ORB

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

План лекций

№	Дата	Тема
1	04.09	Вводная лекция
2	11.09	Эволюция распределенных технологий
3	18.09	Принципы ПО среднего слоя
4	25.09	Стандарты OMG, CORBA и ORB
5	02.10	Перенос
6	09.10	
7	16.10	
8	23.10	
9	30.10	
10	06.11	

План лекции

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

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



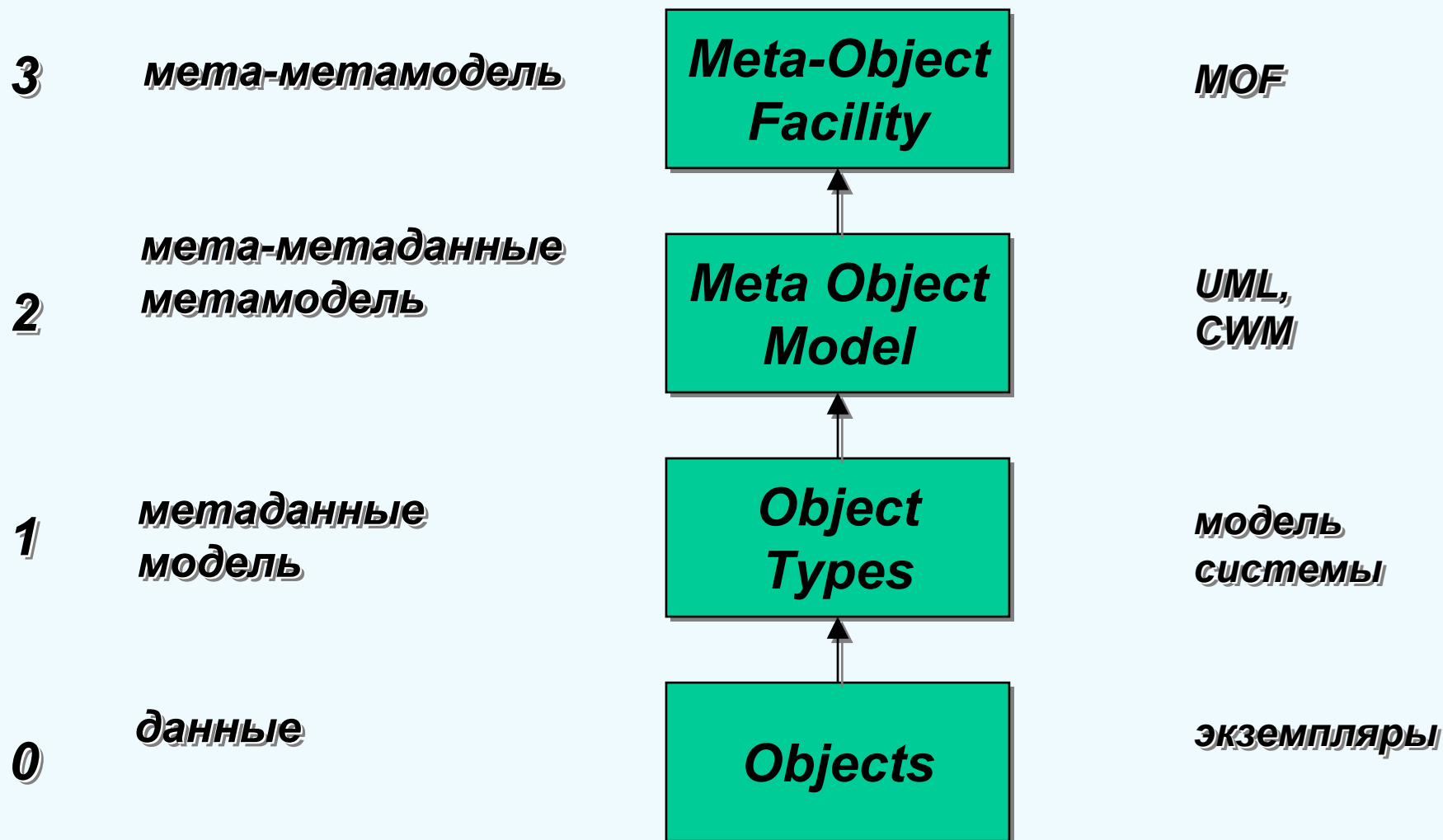
Стандарты OMG

- OMA: Object Management Architecture
- CORBA: Common Object Request Broker Architecture
- MOF: Meta Object Facility
- MDA: Model Driven Architecture
- UML: Unified Modeling Language
- XMI: XML Metadata Interchange
- CWM: Common Warehouse Metamodel

Принятие OMG-стандартов ISO/IEC

- Стандарты 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

Различные уровни абстракции систем



MOF: Meta Object Facility

- Две части:
 - Обобщенное средство для представления метамodelей (средств описания различных моделей)
 - Репозиторий для моделей и метамodelей в рамках архитектуры CORBA
- 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

UML: Unified Modeling Language

- Множество методов представления объектно-ориентированного анализа и проектирования в начале 90-х
 - Booch's Grady Booch
 - OOSE Ivar Jacobson
 - OMT Jim Rumbaugh
- Компания Rational Software объединила различные языки проектирования в UML
- UML:
 - Визуальная, унифицированная нотация моделирования
 - Стандартным образом описывает требования и проект объектно-ориентированной программной системы



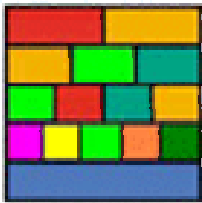
UML: Unified Modeling Language

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



CWM: Common Warehouse Metamodel

- Стандартизирует метамодели данных для баз данных и хранилищ
- Определяет метамодели для реляционных и многомерных данных, преобразований, OLAP и data mining, и т.п.



CWMTM
common warehouse metamodel

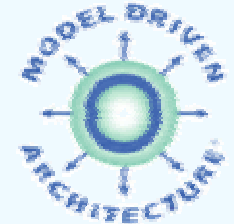
XMI: XML Metadata Interchange

- XMI – потоковый формат для обмена MOF-совместимыми метаданными включая UML-модели
- OMG использовал XML в качестве основы для языка обмена метамоделями
- XMI это отображение MOF в XML
 - Отображение UML в XML
 - Отображение CWM в XML

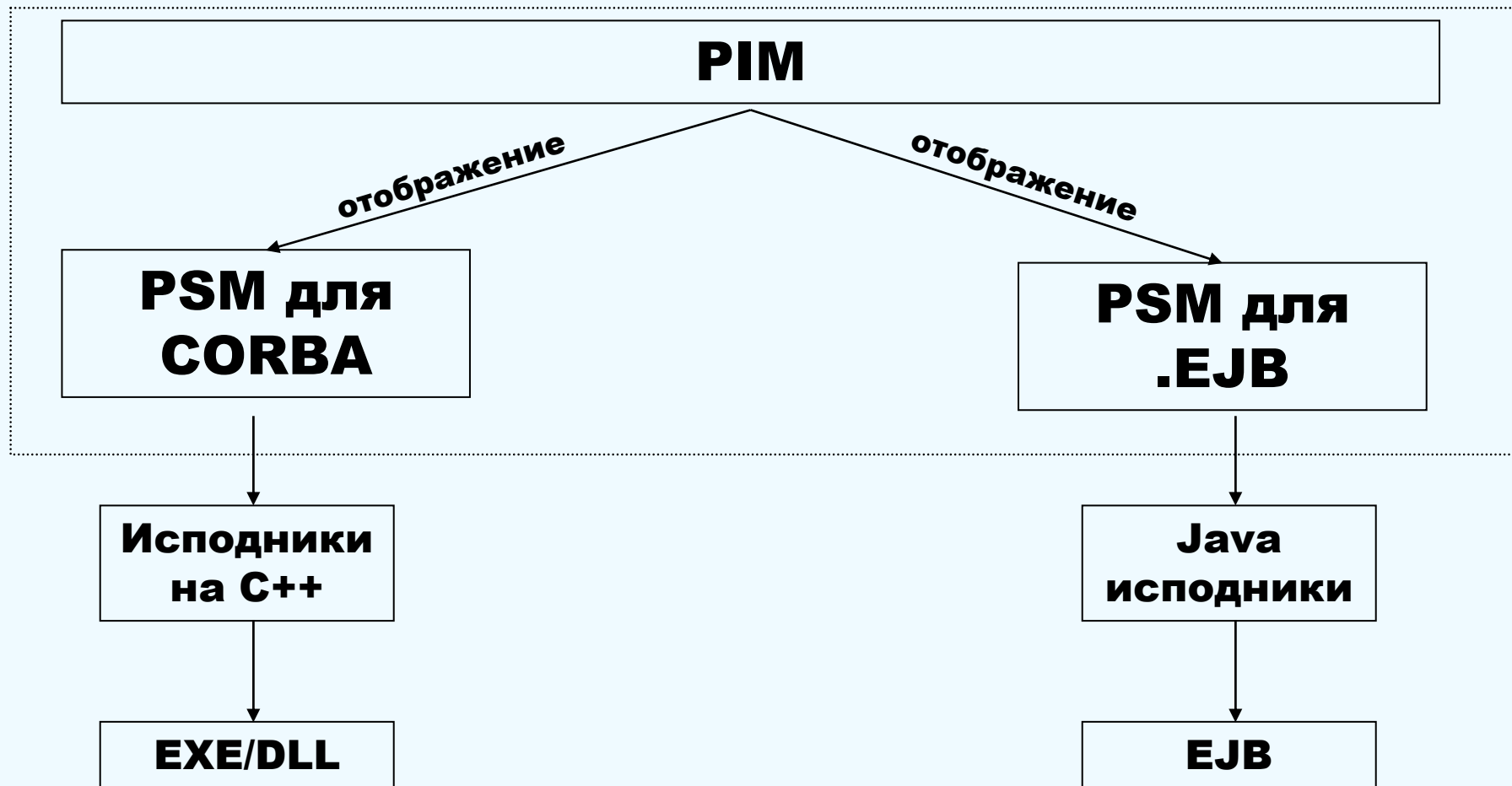


MDA: Model Driven Architecture

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

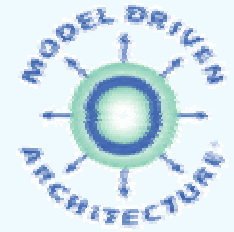


MDA: Model Driven Architecture



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



ОМА и CORBA

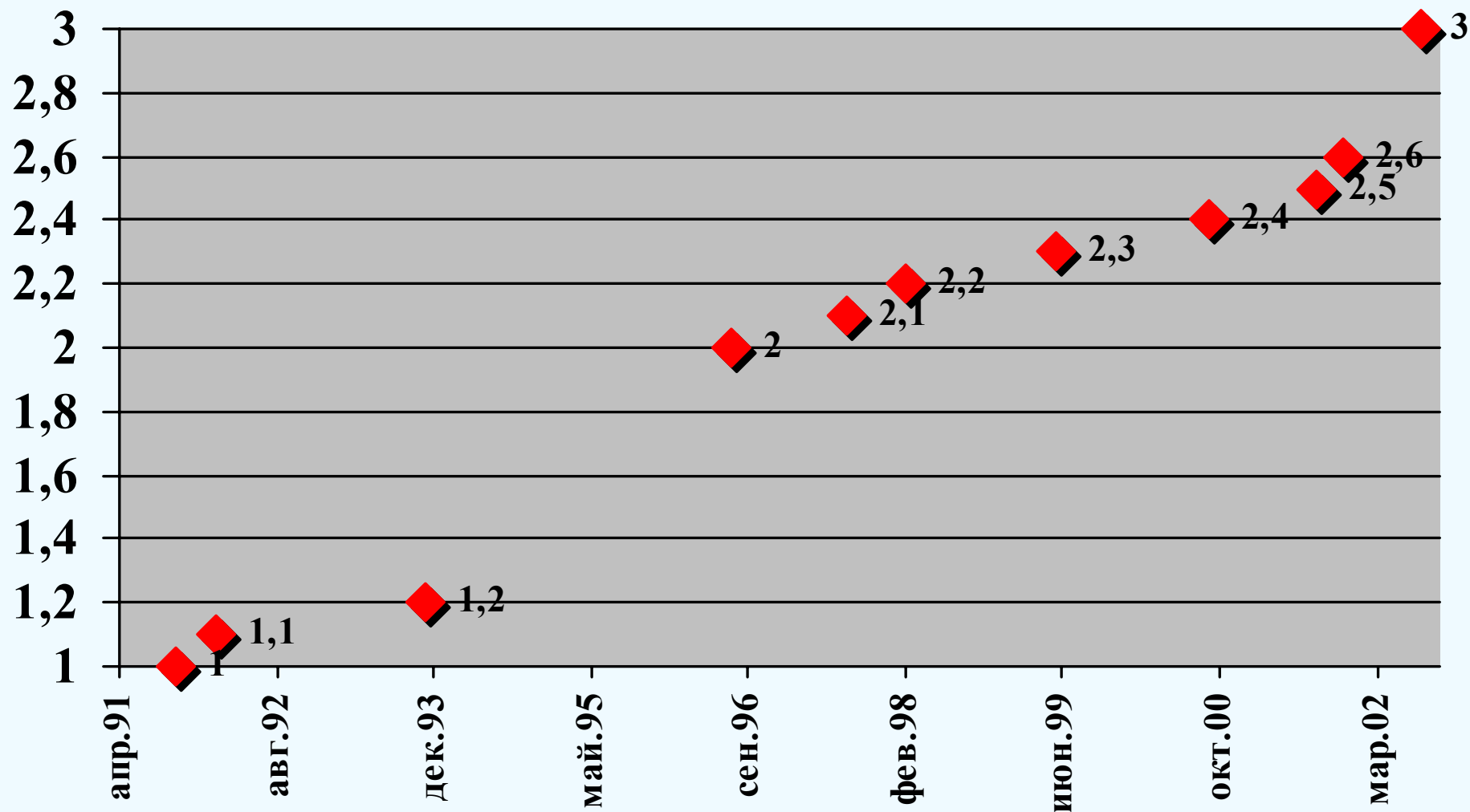
- OMA (Object Management Architecture)
 - Определяет основные архитектурные компоненты распределенной неоднородной объектной среды, их интерфейсы и протоколы взаимодействия.
- CORBA (Common Request Broker Architecture)
 - Определение архитектурной платформы для создания распределенных объектных систем,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



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

Развитие стандарта CORBA



CORBA 1.0

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

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

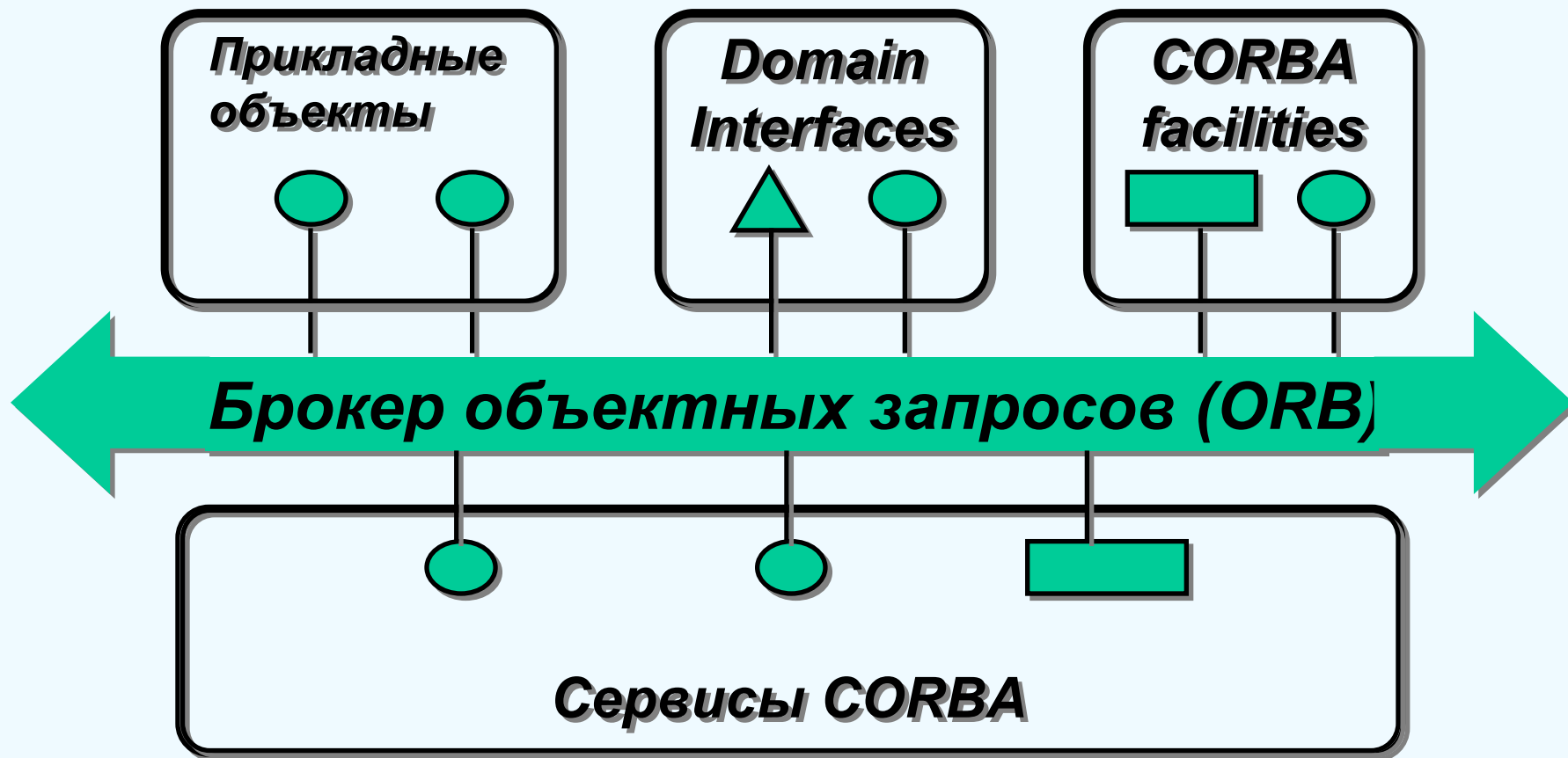
CORBA 3.0

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

План лекции

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

Object Management Architecture



Спецификация ОМА определяет эталонную модель распределенных объектов и делит их на различные категории

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



Брокер объектных запросов (ORB)

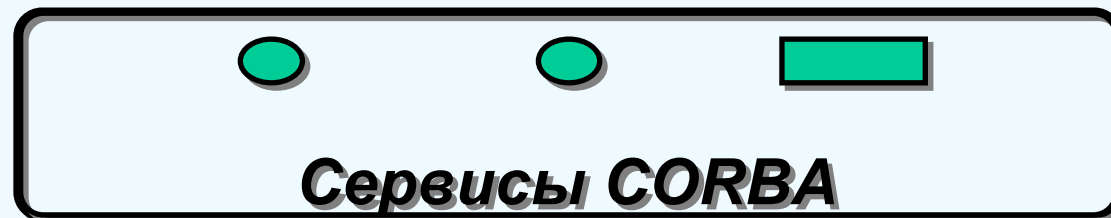
Прикладные объекты

- Объекты, разработанные под конкретные задачи
- Стандартизация вряд ли возможна



Сервисы CORBA (CORBA Services)

- Независимые от предметной области интерфейсы к сервисам, 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.



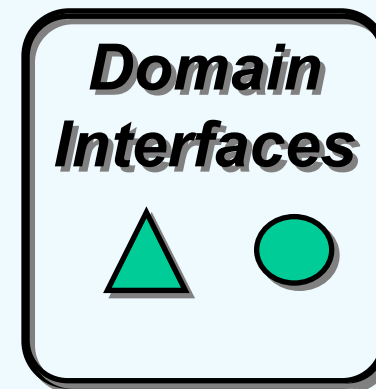
Общие (горизонтальные) средства (Common Facilities)

- Также как и Сервисы CORBA горизонтально ориентированы
- В отличии от Сервисов CORBA ориентированны на приложения конечных пользователей
 - Distributed Document Component Facility - средства компонентов для распределенных документов
 - интернационализация

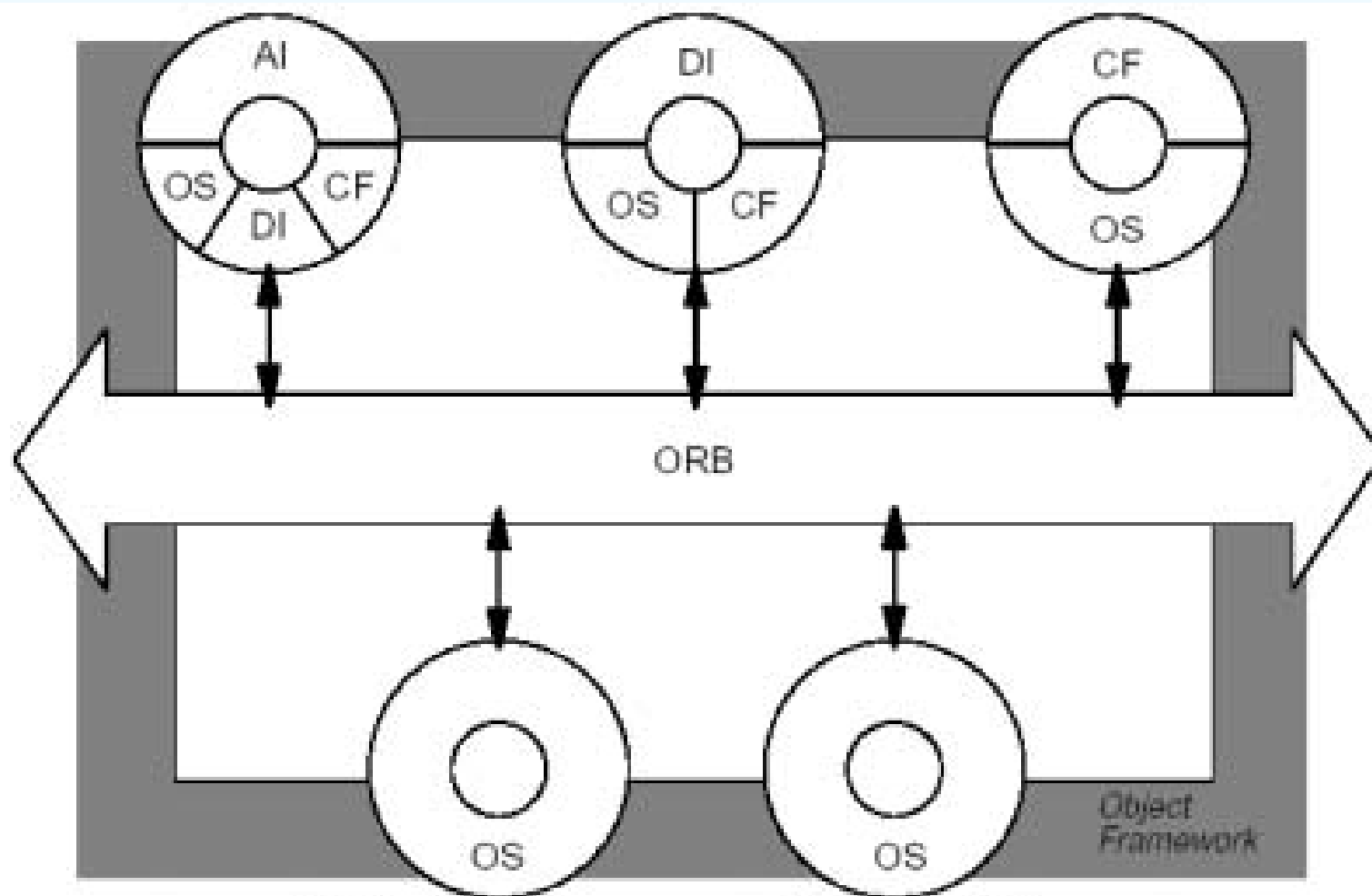


Интерфейсы предметной области (Domain Interfaces)

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



ОМА: Использование интерфейсов



AI=Application Interfaces

DI=Domain Interfaces

CF=Common Facilities

OS=Object Services

План лекции

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

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)

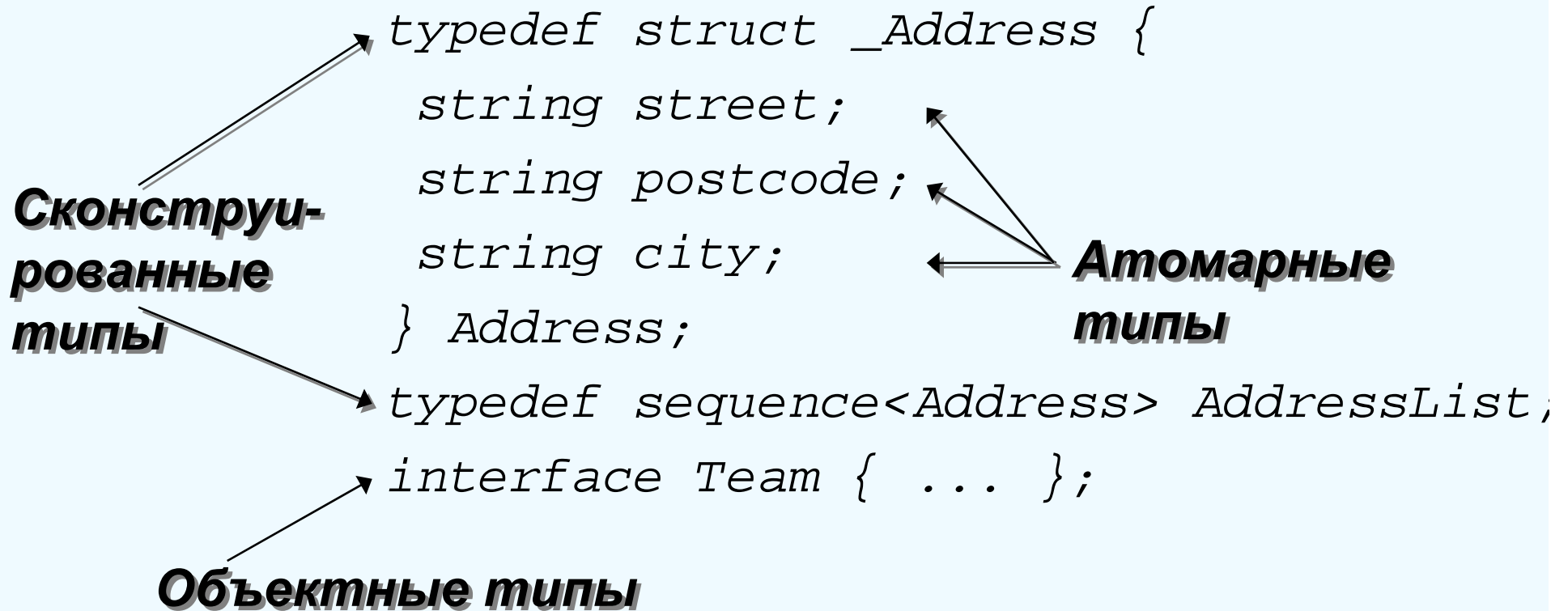
Язык определения интерфейсов OMG IDL

- 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»

Объекты (Objects)

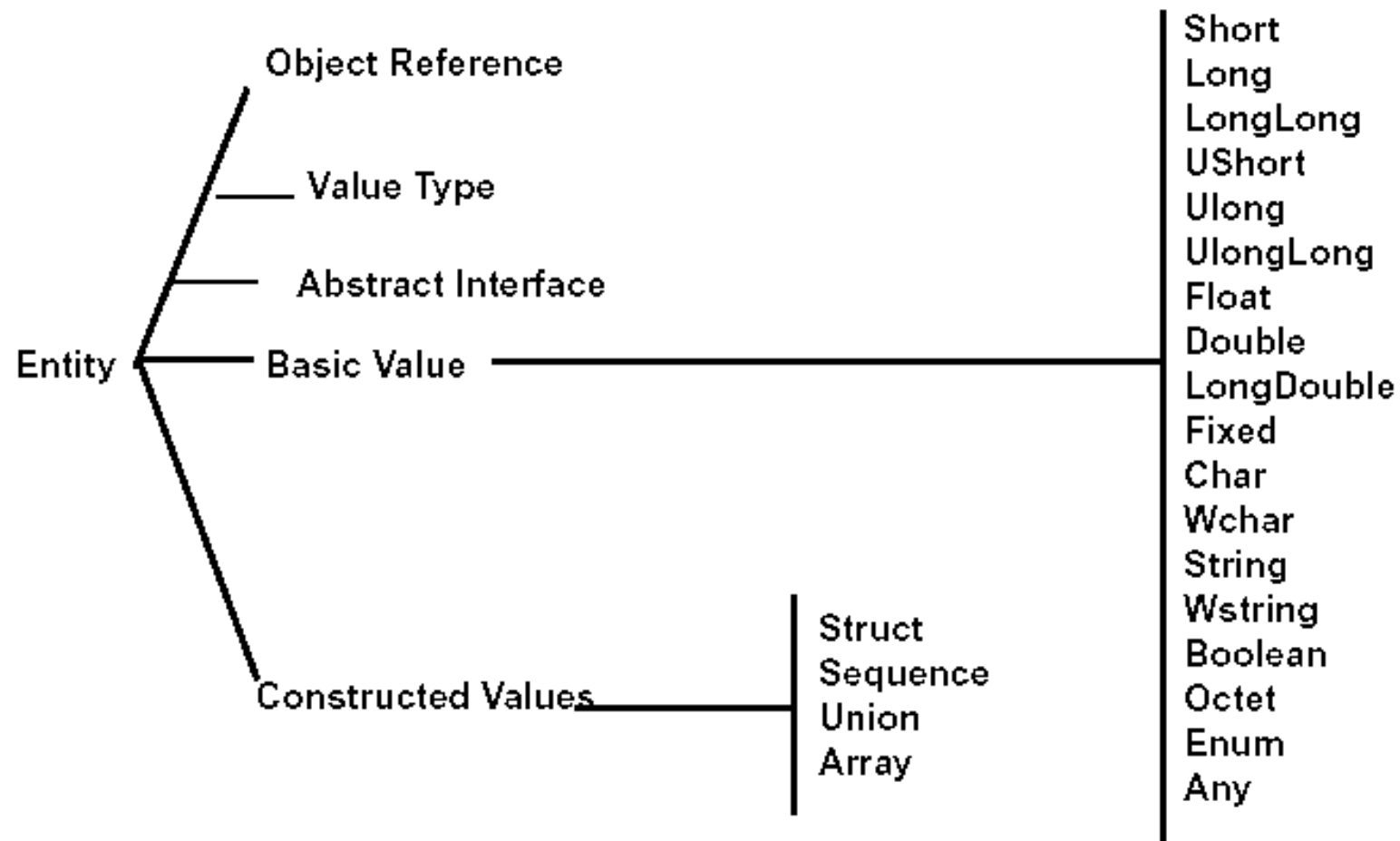
- Предоставляет один или более сервисов (операций)
- Каждый объект имеет один уникальный в пределах ORB идентификатор
- Множество ссылок на объект
- Ссылки поддерживают прозрачность расположения
- Объектные ссылки постоянны (persistent)
- Объекты создаются и уничтожаются вследствие запросов

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.

Типы (Types) прод.



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

Модули (Modules)

Модули

```
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  
};
```

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;  
    ...  
};
```

↑ ↑

Тип атрибута **Имя атрибута**

**Не изменяемое
значение**

Изменяемое

Операции (Operations)

Типы возвращаемых значений → **Вид параметра**

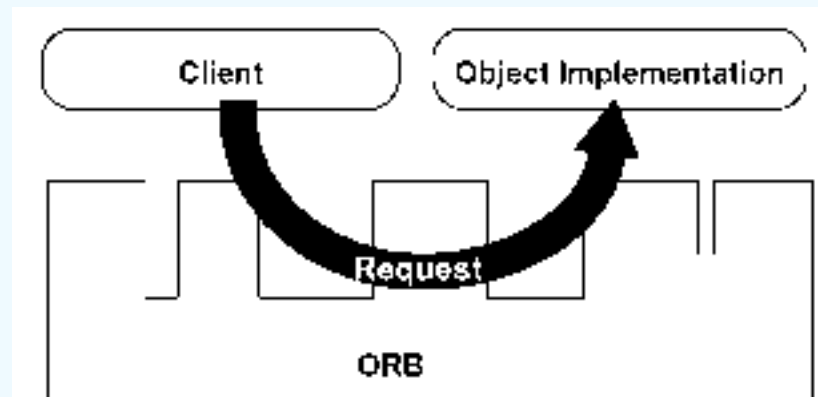
```
interface Team {  
    ...  
    void bookGoalies(in Date d);  
    string print();  
};
```

Список параметров → **Тип параметра** → **Имя параметра**

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

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



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*

Подтипы, наследование (Inheritance)

Неявный супертип:

Object

Наследуется типом Club

```
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;  
};
```

Супертип

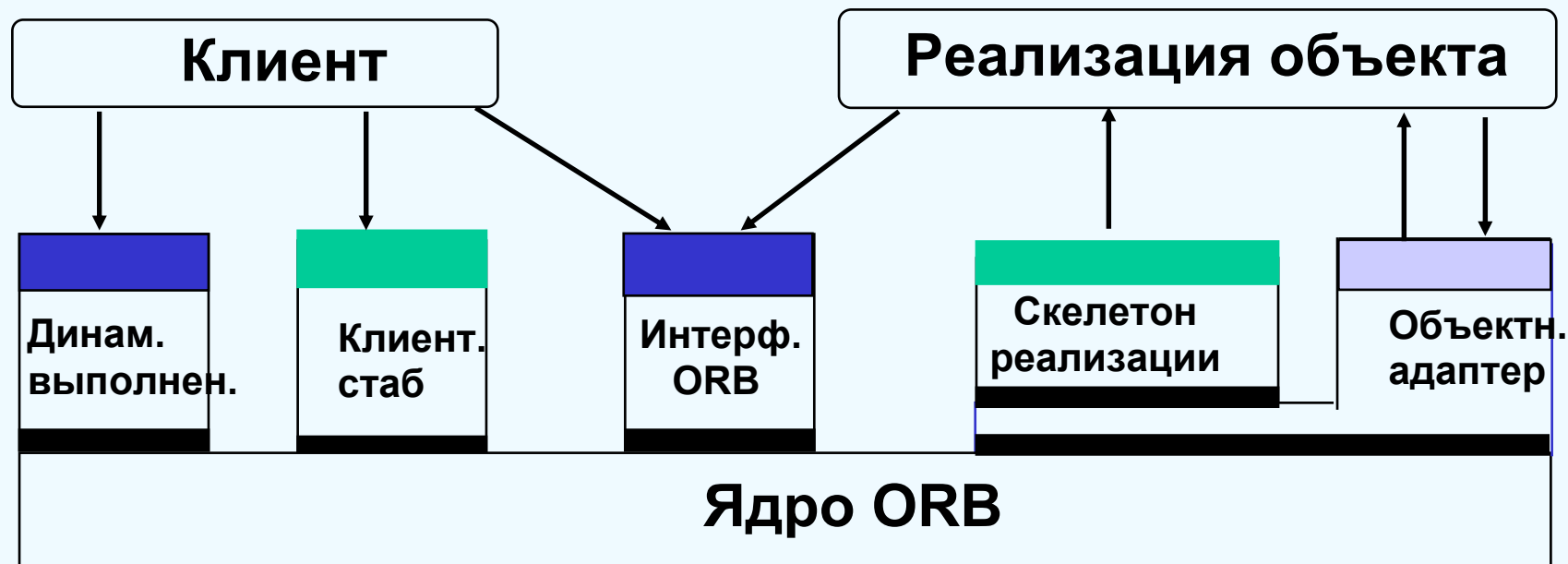
Подтипы, наследование (Inheritance)





- 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»)

План лекции

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

Основные элементы архитектуры CORBA



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

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 ОМА (Сервисы, общие средства,...)

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

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
```

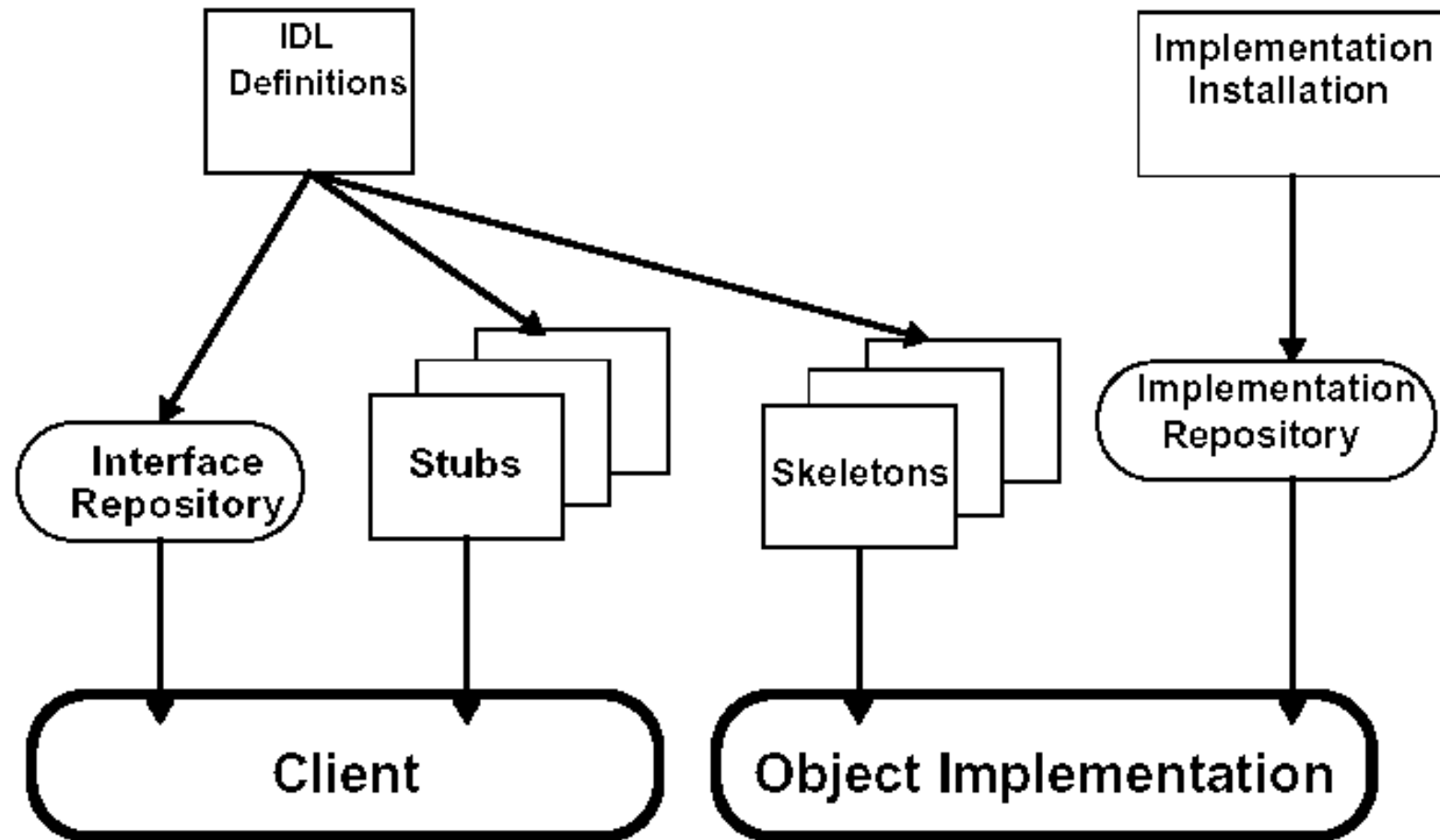
```
// Trading Service (not shown), then issue request
```

```
Object_var objref = factory_objref->create();
```


IDL Скелетоны (Skeletons)

- Сущность языка программирования, соединяющая ОА с его реализацией (servant)
- Реализация:
 - язык С: 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)

Компиляция определений интерфейсов IDL



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

Объект (Object)

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

Клиент (Client)

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

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

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)

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)

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

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