

Распределенные вычислительные системы

Лекция №6: COM, Java/RMI

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	Пример приложения на CORBA
6	09.10	COM, Java/RMI
7	16.10	Лекция
8	23.10	Перенос (?)
9	30.10	Лекция + <u>Рейтинг по Л.1-7</u>
10	06.11	Лекция

План лекций (продолжение)

№	Дата	Тема
11	13.11	Лекция
12	20.11	Лекция
13	27.11	Лекция
14	04.12	Лекция
15	11.12	Лекция
16	18.12	Лекция
17	25.12	Зачет

План лекции

- Анонс рейтинга
- COM
- Java/RMI

Анонс рейтинга

- 30.10 планируется проведения рейтинга
- Рейтинг
 - 20-30 минут
 - Каждому по 3 вопроса
 - Случайная выборка из общего пула в 60-70 вопросов (пул формируется)
- 20% лучших получают преимущество на зачете

План лекции

- Анонс рейтинга
- COM
- Java/RMI

СОМ: Цели

- Компонентная модель, обеспечивающая двоичную инкапсуляцию и двоичную совместимость
 - Двоичная инкапсуляция: Клиенты не должны быть перекомпилированы если серверные объекты изменяются
 - Двоичная совместимость: Клиентские и серверные объекты могут быть разработаны с помощью различных средств и языков
- СОМ – собственный де-факто стандарт

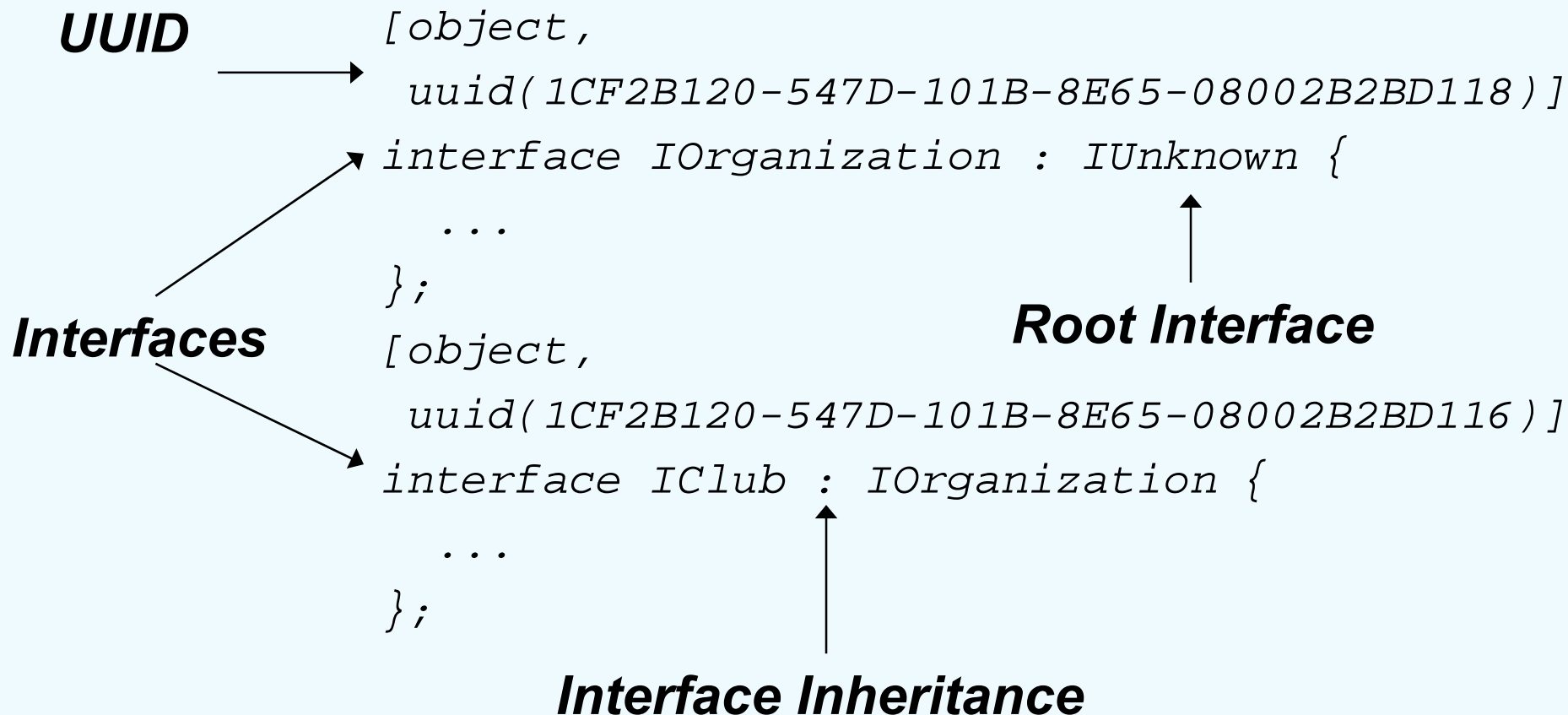
СOM: Объектная модель и определение интерфейса

- Интерфейсы
- Реализации и объекты
- Классы
- Атрибуты
- Операции
- Запросы
- HRESULTS
- Наследование

COM: Microsoft IDL (MIDL)

- Язык, выражающий концепции COM
- MIDL:
 - Язык независимый от языка программирования
 - В основе лежит OSF/RPC IDL
 - Вычислительно неполон
- Доступно связывание с различными языками

COM: Интерфейсы



COM: Реализации

- Реализует интерфейсы на языках программирования (C++)

```
#include "Soccer.h"
class Player : public IPlayer {
private:
    char* name;
    short Number;
protected:
    virtual ~TrainerPlayer(void);
public:
    TrainerPlayer(void);
    IMPLEMENT_UNKNOWN(TrainerPlayer)
    BEGIN_INTERFACE_TABLE(TrainerPlayer)
    IMPLEMENTS_INTERFACE(ITrainer)
    IMPLEMENTS_INTERFACE(IPlayer)
    END_INTERFACE_TABLE(TrainerPlayer)
    void book(); // IPlayer methods
};
```

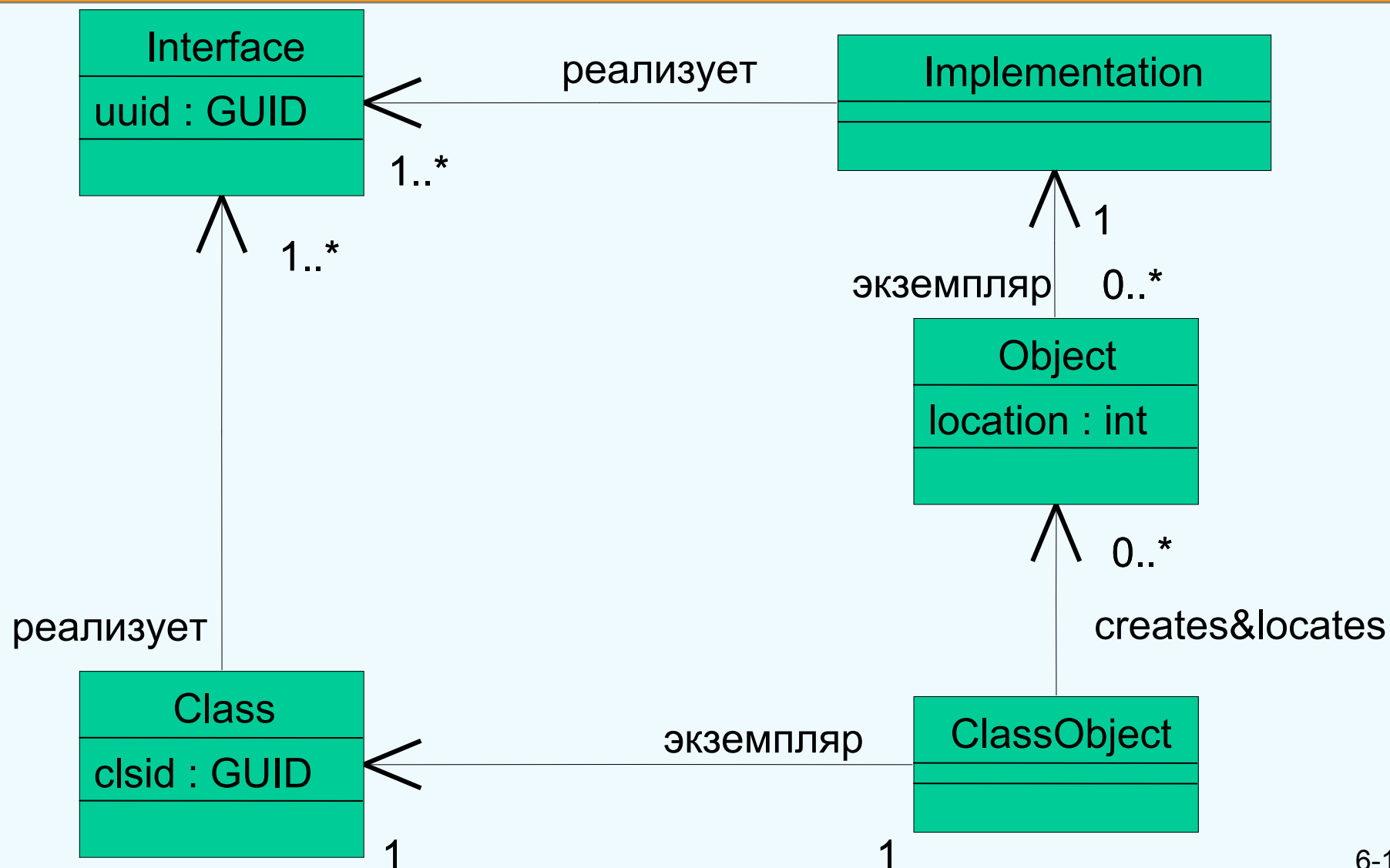
COM: Объекты

- Экземпляр реализации COM
- Ссылки на объекты COM называются указателями на интерфейсы
- Указатели на интерфейсы ссылаются на области в основной памяти
- Ссылки обеспечивают прозрачность расположения
- Объектные ссылки постоянны (persistent)

COM: Классы

- Именованные реализации
- Один или несколько интерфейсов
- Основной механизм для создания объектов COM
- Может вернуть указатель на интерфейс к объекту COM

COM: Объекты, интерфейсы и классы



COM: Атрибуты

- COM не поддерживает атрибуты
- Атрибуты должны быть представлены как операции установки и чтения (set / get)
- COM использует специальное ключевое слово для обозначения
- Пример:

```
interface IOrganization : IUnknown {  
    [propget] HRESULT Name([out] BSTR val);  
};
```

COM: Операции

Вид параметра **Список параметров**

```
interface IClub : IOrganization {  
    [propget] HRESULT NoOfMembers([out] short *val);  
    [propget] HRESULT Address([out] ADDRESS *val);  
    [propget] HRESULT Teams([in] long cMax, [out] long *pcAct,  
        [out,size_is(cMax),length_is(*pcAct)] ITeam *val);  
    [propput] HRESULT Teams([in] long cElems,  
        [in,size_is(cElems)] ITeam *val);  
    [propget] HRESULT Trainers([out] ITrainer *val[3]);  
    [propput] HRESULT Trainers([in] ITrainer *val[3]);  
    HRESULT transfer([in] IPlayer *p);  
};
```

Возвращает значение, указывающее успех/неудачу **Имя операции** **Параметры, например указатель на интерфейс**

COM: HRESULTS

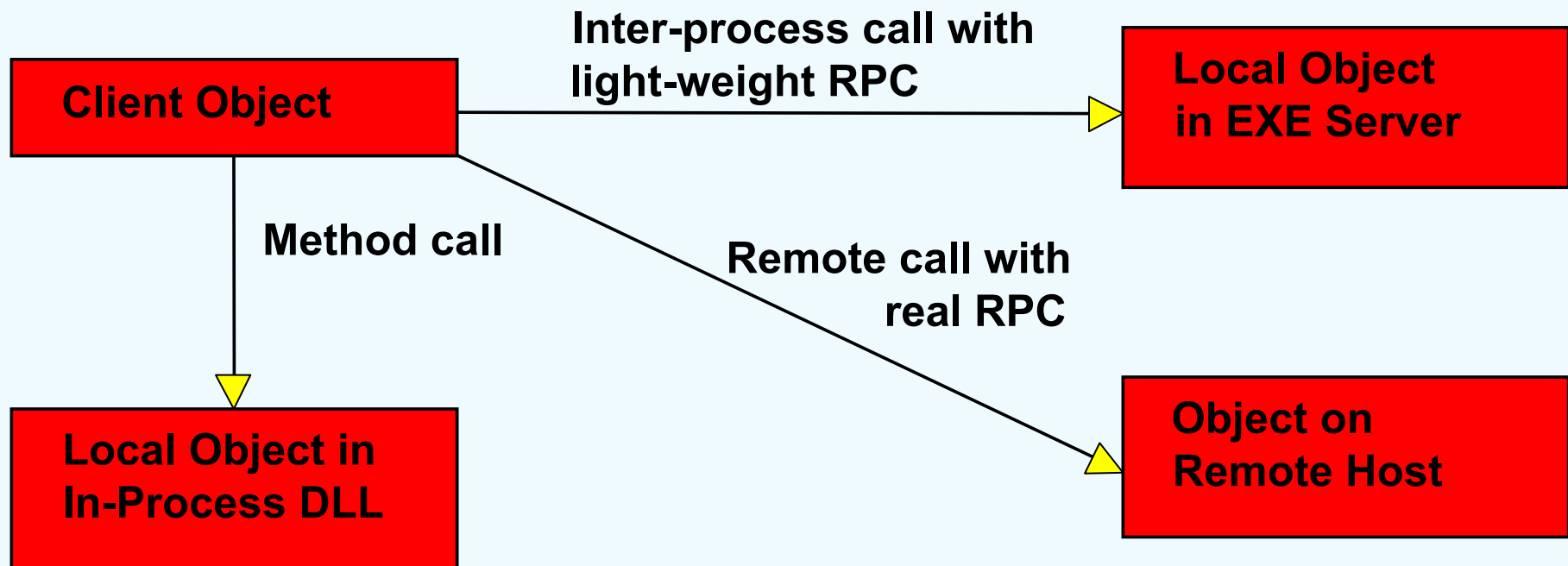
- HRESULTS – 32-битные целые
- Структурированы в виде 4-х полей



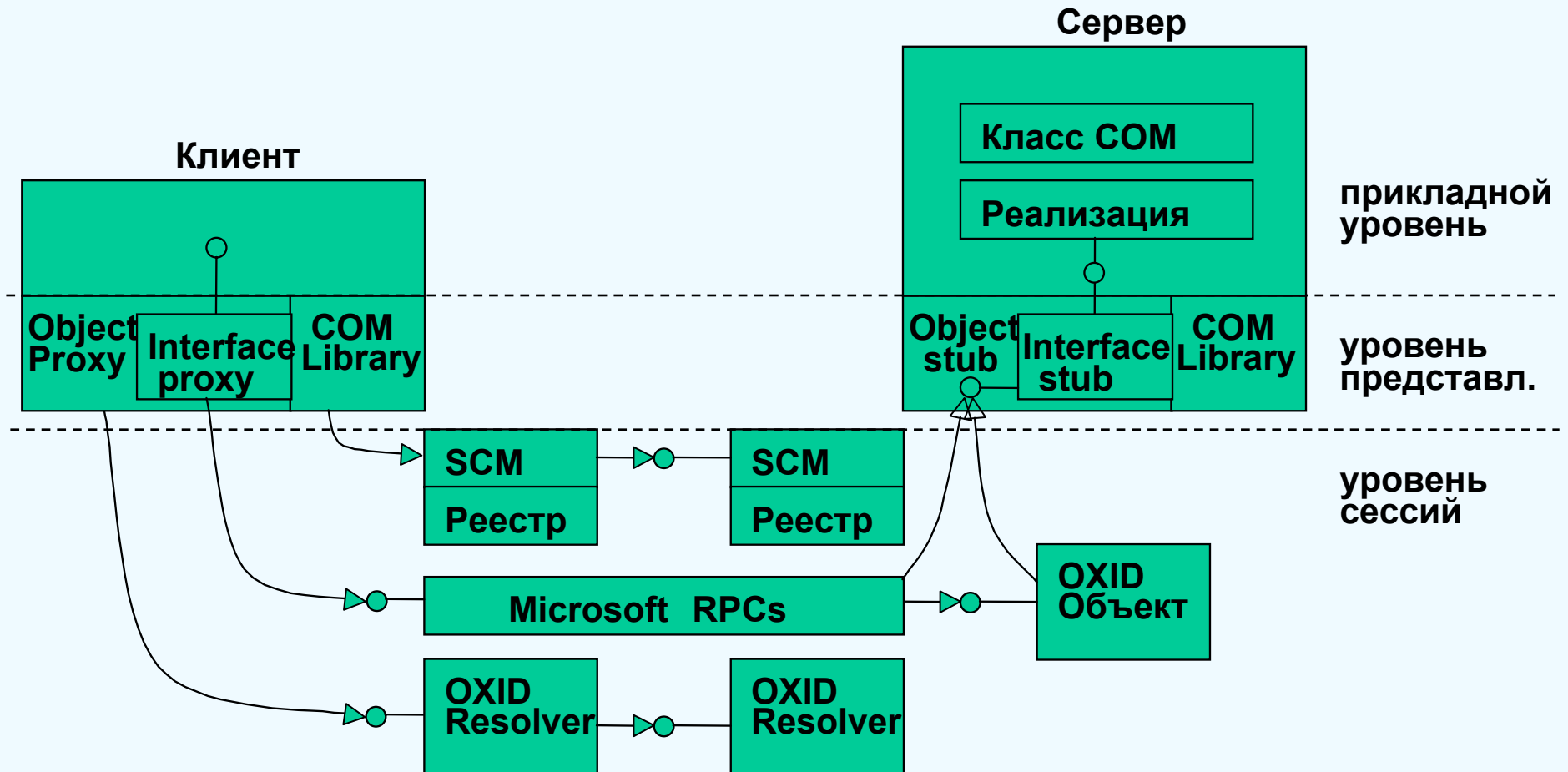
Выполнение операции COM

- Выполнение определяется клиентскими объектами
 - Выполнение определяет:
 - Указатель на интерфейс серверного объекта
 - Имя вызываемой операции
- Актуальные параметры
- Выполнение проводится синхронно
- Выполнение может быть:
 - Статическим
 - Динамическим
- Клиенты должны проверять HRESULTS

Три варианта реализации запроса



Архитектура



План лекции

- Анонс рейтинга
- COM
- Java/RMI

Язык Java

- Java – объектно-ориентированный язык программирования, созданный Sun Microsystems
- Гибридный язык программирования, который и компилируется и интерпретируется
- Компилятор создает байт-код, который после интерпретируется виртуальной машиной

Цели RMI

- В Java 1.0 взаимодействие между объектами ограничено одной виртуальной машиной
- RMI (Remote Method Invocation – Удаленный вызов методов) поддерживает взаимодействие между различными виртуальными машинами (потенциально между различными узлами)
- Обеспечивает тесную интеграцию с Java
- Минимизация изменений в язык Java
- Работает в гетерогенных средах

Объектная модель Java

- Интерфейсы и удаленные объекты
- Классы
- Атрибуты
- Операции
- Исключения
- Наследование

Интерфейсы Java и удаленные объекты

- Java уже включает концепцию интерфейсов в язык
- RMI не имеет специального языка определения интерфейсов
- Предопределенный интерфейс Remote
- Удаленные интерфейсы расширяют интерфейс Remote
- Удаленные классы реализуют удаленные интерфейсы
- Удаленные объекты являются экземплярами удаленных классов

Пример удаленного интерфейса Java

Имя пакета *Имя интерфейса* *Декларация как удаленного*

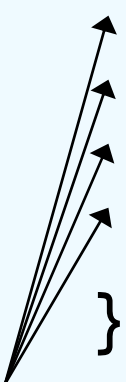
```
package soccer;  
interface Team extends Remote {  
public:  
    String name() throws RemoteException;  
    Trainer[] coached_by() throws RemoteException;  
    Club belongs_to() throws RemoteException;  
    Players[] players() throws RemoteException;  
    void bookGoalies(Date d) throws RemoteException;  
    void print() throws RemoteException;  
};
```

Удаленная операция

Атрибуты

- RMI не поддерживает атрибуты
- Атрибуты должны быть представлены как пары Set и Get операций разработчиком
- Пример:

```
interface Club extends Organization, Remote {  
    public:  
    int noOfMembers() throws RemoteException;  
    Address location() throws RemoteException;  
    Team[] teams() throws RemoteException;  
    Trainer[] trainers() throws RemoteException;  
    ...  
};
```



Операция get для получения значения атрибута

Комбинация классов и удаленных интерфейсов

```
interface Organization {  
    private:  
        String name() RemoteException;  
};  
class Address {  
    public:  
        String street;  
        String postcode;  
        String city;  
};  
interface Club extends Organization, Remote {  
    public:  
        int noOfMembers() throws RemoteException;  
        Address location() throws RemoteException;  
        Team[] teams() throws RemoteException;  
        Trainer[] trainers() throws RemoteException;  
        void transfer(Player p) throws RemoteException;  
};
```

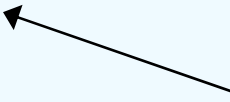
Club возвращает объект Адрес

*Club делает операцию name()
Доступной удаленно*

Операции и передача параметров

- Удаленные вызовы:
 - Атомарные типы передаются по значению
 - Удаленные объекты передаются по ссылке
 - Не-удаленные объекты передаются по значению

```
class Address {  
    public:  
        String street;  
        String postcode;  
        String city;  
};  
interface Club extends Organization, Remote {  
    public:  
        Address location() throws RemoteException;  
        ...  
};
```

 ***Возвращает копию address!***

Заявки

- Семантика однократного выполнения, т.е. если нет ошибки, то метод выполнен успешно
- В отличие от CORBA и COM, поддерживается только статические вызовы (т.е. до компиляции приложения необходимо знать с кем будет взаимодействовать клиентский объект)

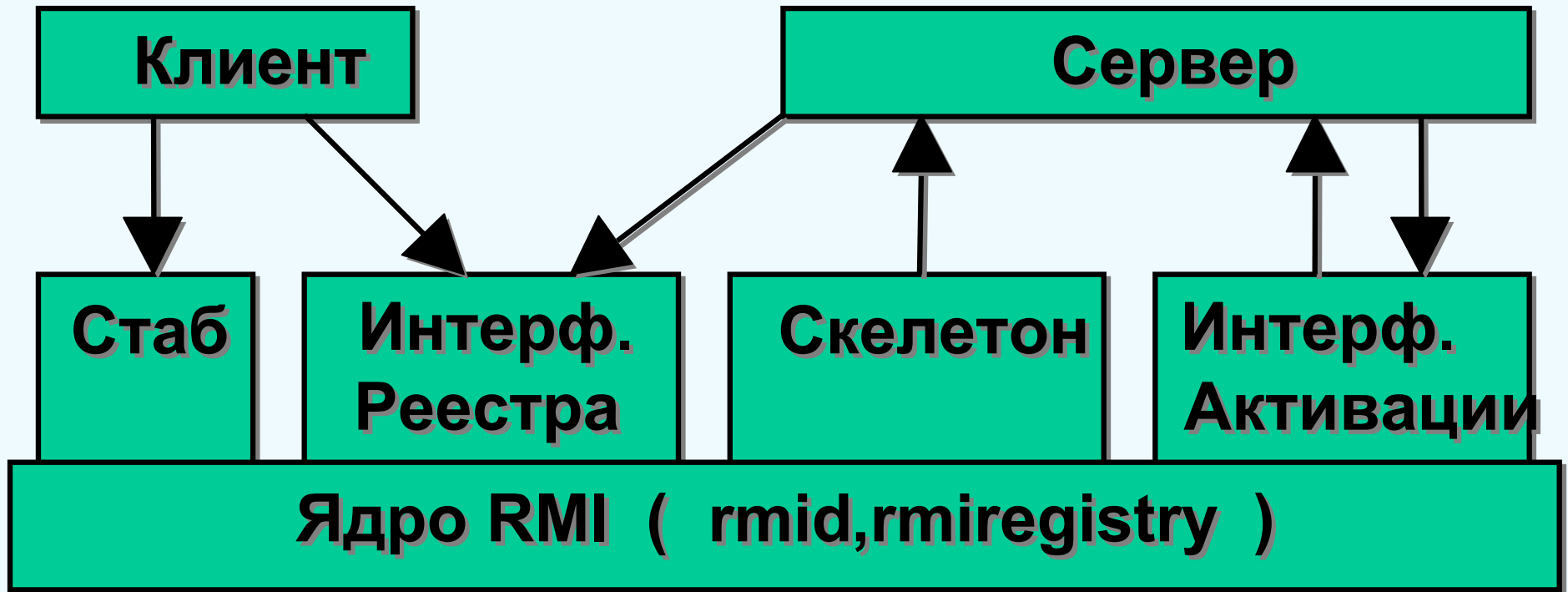
Исключения

- Предопределенные исключения RemoteException
- Типо-зависимые исключения
- Пример:

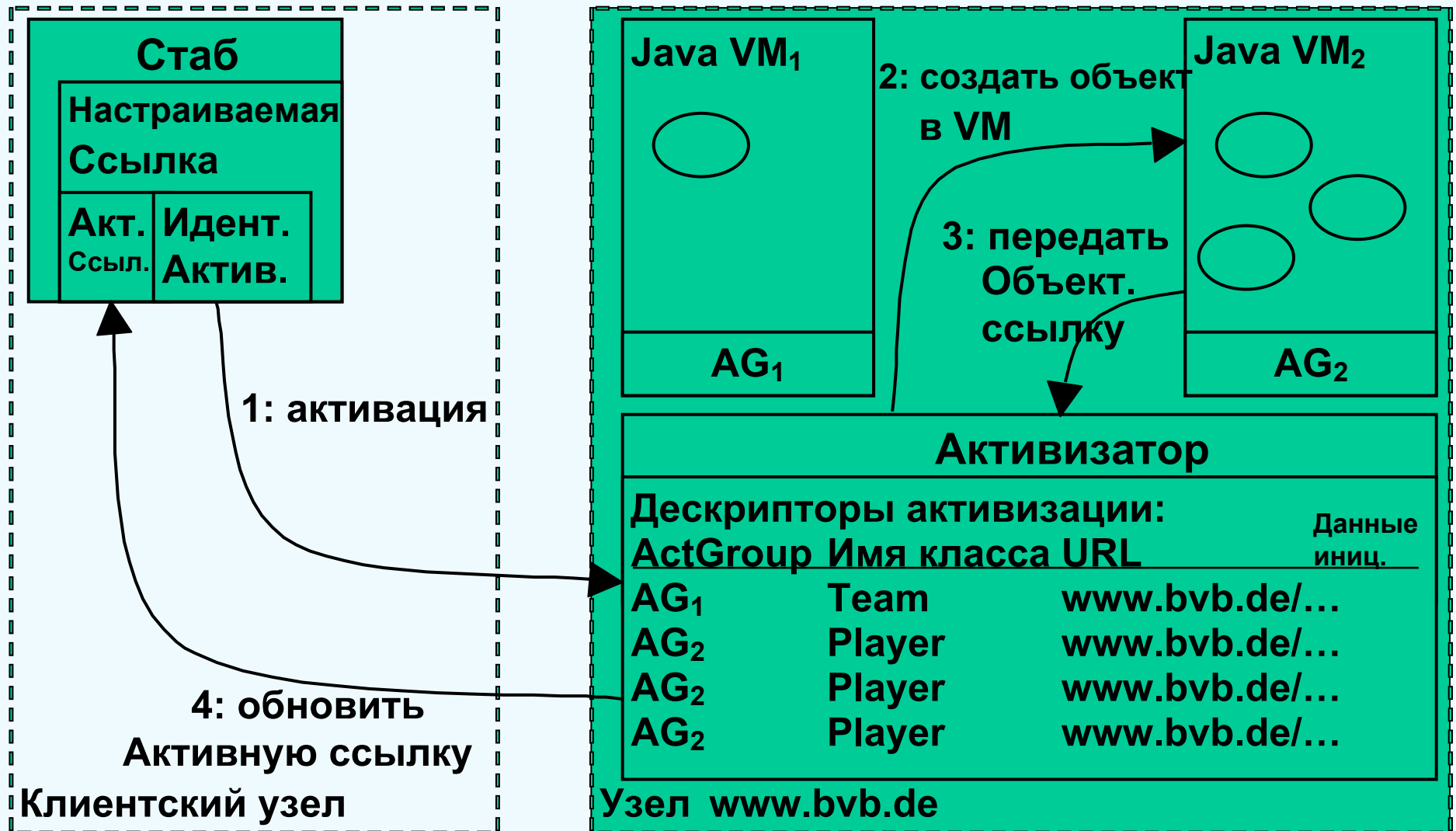
Типо-зависимые исключения

```
class PlayerBooked extends Exception {};  
interface Team extends Remote {  
    public:                Операция декларирует, что она может  
                           вызвать это исключение  
        ...  
        void bookGoalies(Date d) throws  
            RemoteException, PlayerBooked;  
        ...  
};
```

Архитектура



Активация в Java



Обсуждение

- Использование различной семантики вызова для удаленных и локальных объектов означает неполную прозрачность доступа
- Стандартный механизм ссылок в Java не обеспечивает прозрачности местонахождения
- В этом отношении Java/RMI стоит ниже нежели COM & CORBA

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