RMI – вызов удаленных методов

RMI – вызов удаленных методов

RMI (Remote method invocation) – технология распределенного объектного взаимодействия, позволяющая объекту, расположенному на стороне клиента, вызывать методы объектов, расположенных на стороне сервера (удаленных объектов). Для программиста вызов удаленного метода осуществляется так же, как и локального.

Определения

Удаленный объект — объект, методы которого могут быть вызваны из другой виртуальной Java-машины, возможно расположенной на другой вычислительной системе.

Удаленный интерфейс — интерфейс, который реализуют удаленные объекты.

RMI – вызов удаленных методов

Вызов удаленного метода — действие по вызову метода и удаленного интерфейса, реализованного в удаленном объекте. Вызов такого метода имеет такой же синтаксис, как и вызов локального.

Сервер объектов — программа, предоставляющая удаленные методы для вызова.

Клиент — программа, осуществляющая вызов удаленных методов.

Каталог удаленных объектов (RMI Registry) — служебная программа, работающая на той же вычислительной системе, что и сервер объектов. Позволяет определять объекты, доступные для удаленных вызовов с данного сервера.

RMI – вызов удаленных методов

Объект-заглушка (Stub) - посредник удаленного объекта со стороны клиента. Предназначен для обработки аргументов и вызова транспортного уровня.

RMI – вызов удаленных методов

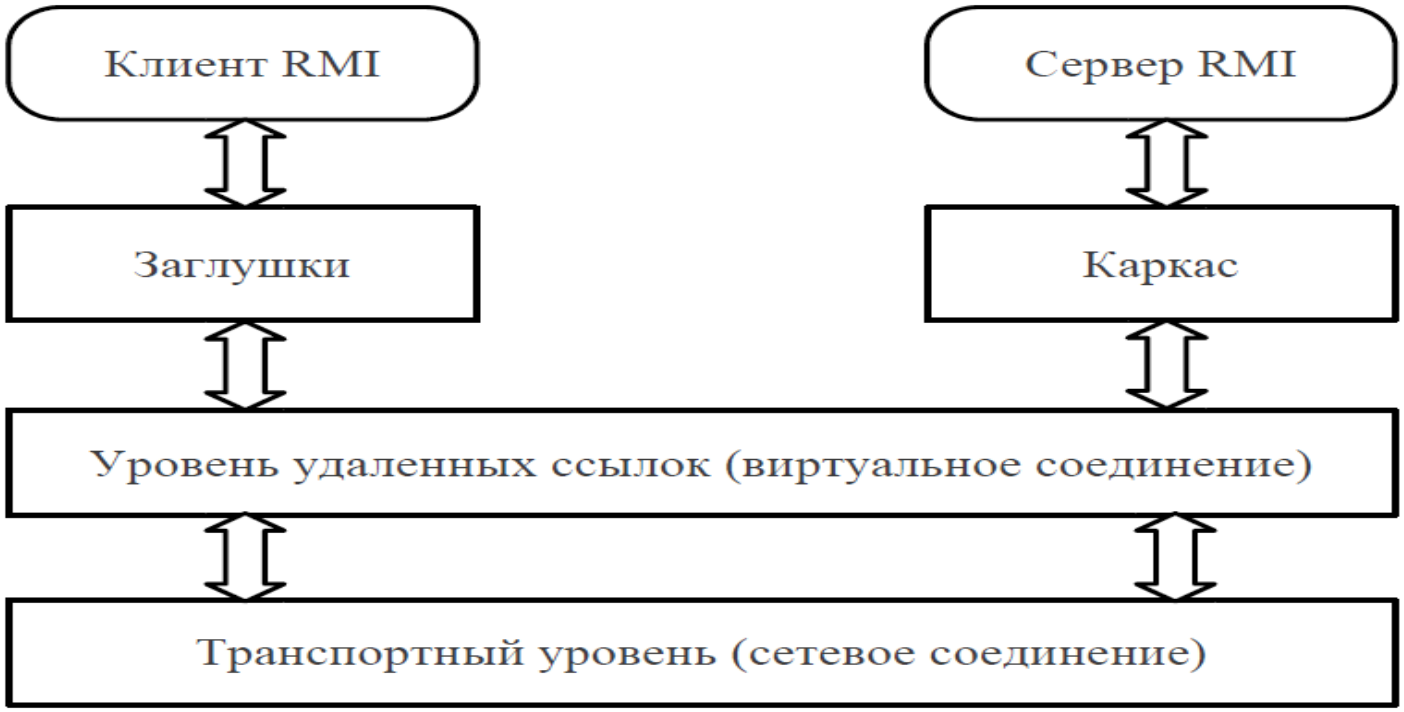

Рис. Структура RMI

RMI – вызов удаленных методов

Алгоритм работы с RMI

1. Определение удаленных интерфейсов.
2. Создание сервера.
3. Создание клиента.
4. Запуск каталога удаленных объектов, сервера и клиента.

Определение удаленных интерфейсов

Требования к удаленному интерфейсу:

- должен иметь модификатор `public`;
- должен наследоваться от `java.rmi.Remote;`
- каждый метод интерфейса должен объявлять, что он выбрасывает `java.rmi.RemoteException;`
- аргументы и значения методов должны иметь примитивный или сериализуемый тип, либо тип удаленного интерфейса.

RMI – вызов удаленных методов

Пример

```
package hello;
import java.rmi.*;
public interface Hello extends Remote {
   String sayHello() throws RemoteException;
}
```

RMI – вызов удаленных методов

Создание сервера

1. Объявление класса, реализующего удаленный интерфейс:

```
package hello;
public class Server implements Hello {
   public Server() {
   }
   public String sayHello() {
       return “Hello, World!”;
   }
}
```

RMI – вызов удаленных методов

2. Создание и экспорт удаленного объекта.

Метод `exportObject` класса `java.rmi.server.UnicastRemoteObject` экпортирует удаленный объект, позволяя ему принимать удаленные вызовы на анонимный порт. Метод возвращает объект-заглушку, которая передается клиентам. Можно задать определенный порт, указав его в качестве второго аргумента.

```
Server obj = new Server();
Hello stub = (Hello)
  UnicastRemoteObject.exportObject(obj,0);
```

RMI – вызов удаленных методов

3. Зарегистрировать удаленный объект в каталоге RMI registry:

```
Registry reg = LocateRegistry.getRegistry();
reg.bind(“Hello”, stub);
```

Создание клиентов

1. Получение ссылки на удаленный метод из каталога RMI registry:

```
Registry reg = LocateRegistry.getRegistry(hostname);
Hello stub = (Hello) registry.lookup("Hello");
```

RMI – вызов удаленных методов

2. Вызов удаленного метода

```
String response = stub.sayHello();
System.out.println(response);
```

Запуск каталога, сервера и клиентов

Для UNIX/Linux:

```
rmiregistry &
java -classpath path -Djava.rmi.server.codebase=file:path/ Server &
java -classpath path Client
```

RMI – вызов удаленных методов

Для Windows:

```
start rmiregistry
start java -classpath path -Djava.rmi.server.codebase=file:path/ Server
java -classpath path Client
```