UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA

Sistemas Distribuídos

Prof. Sergio T. Carvaho

sergio@inf.ufg.br

Programação Java RMI

Programação Java RMI

Conceitos

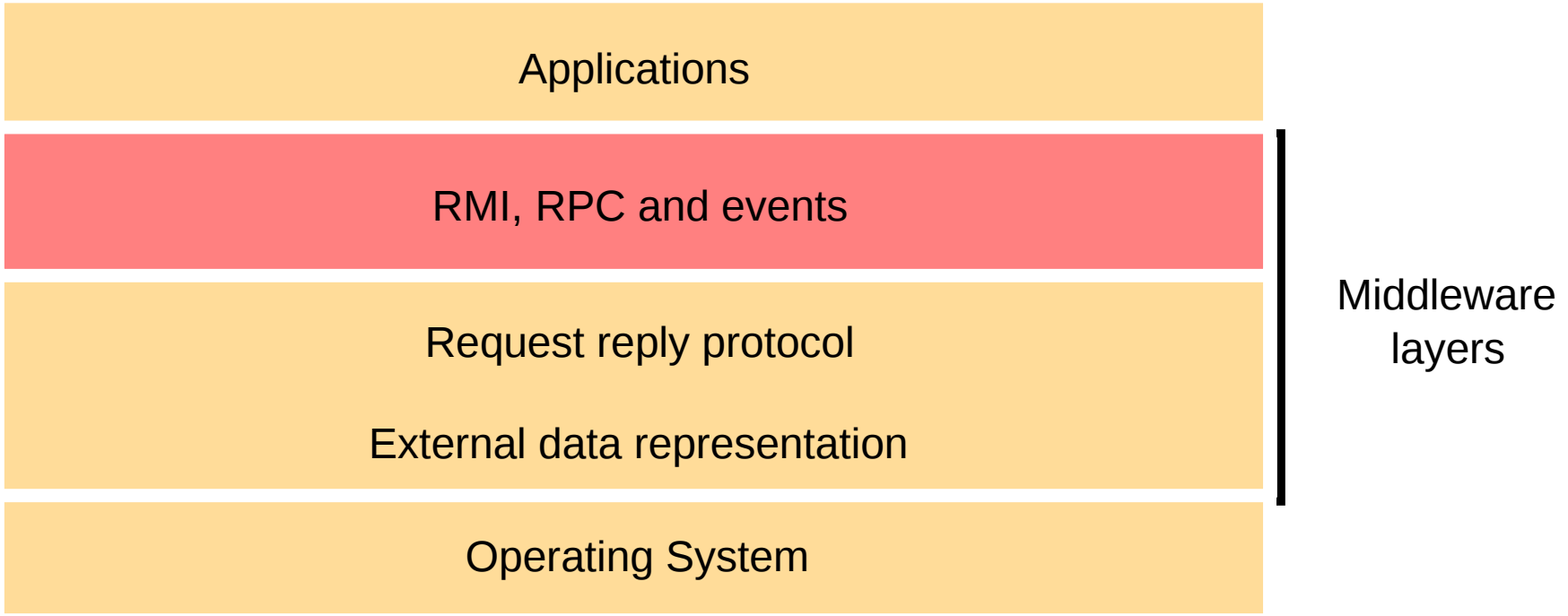
Mecanismo

Exemplo de Invocação Remota

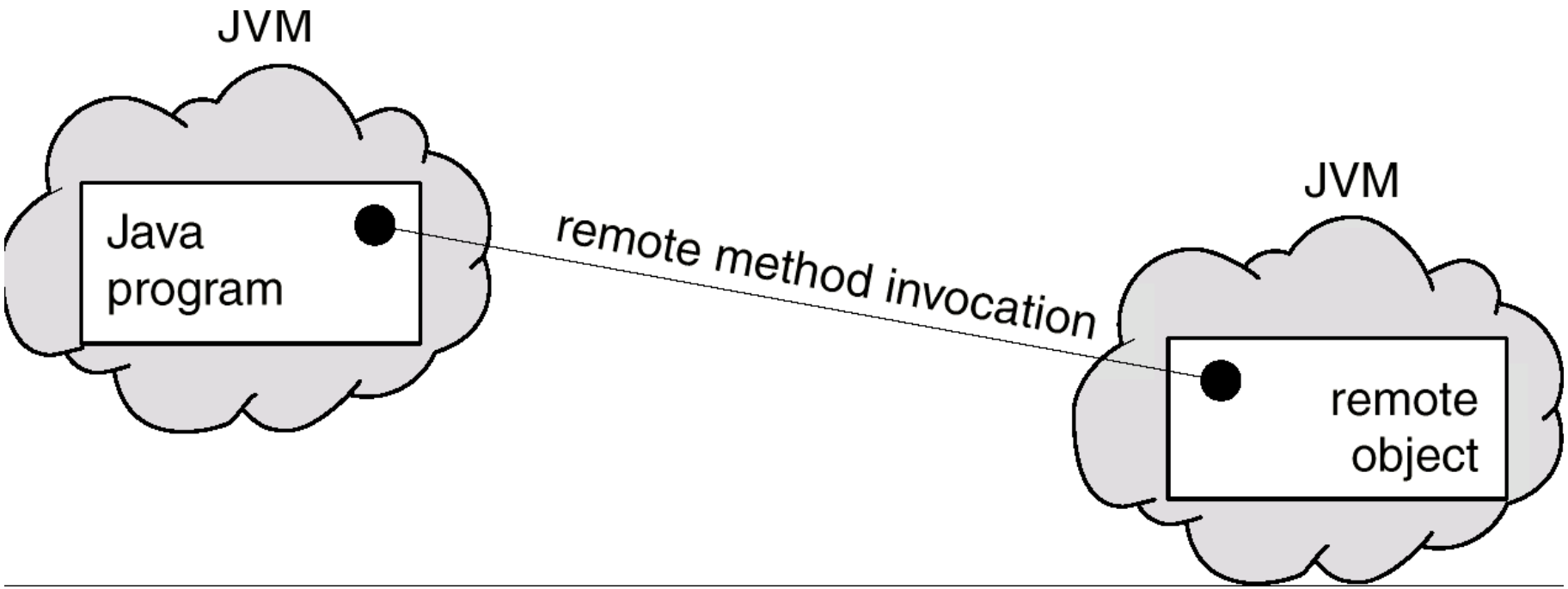
Request-Release com Monitor Remoto

Sumário

Camadas do Middleware



Remote Method Invocation (RMI)



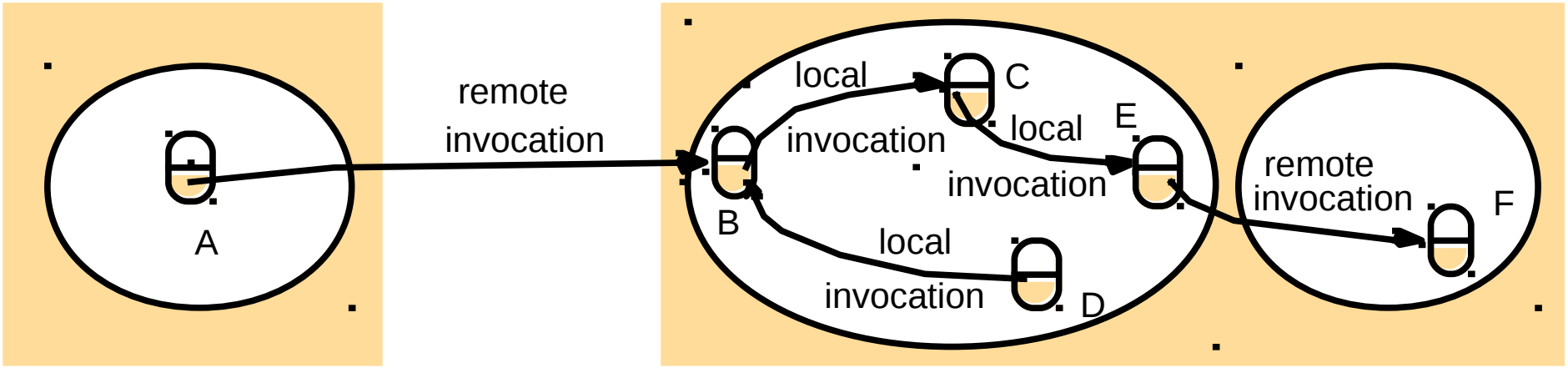
Java RMI – Conceitos

- **Modelo de objetos distribuídos c/ invocação de métodos como fundamento**
- **Semântica do modelo de objetos padrão**
- **Obtém referência do objeto remoto**
- **Invoca métodos diretamente ao objeto remoto**
- **Interface vs. Implementação**
- **Semelhança com o RPC (*Remote Procedure Call*)**

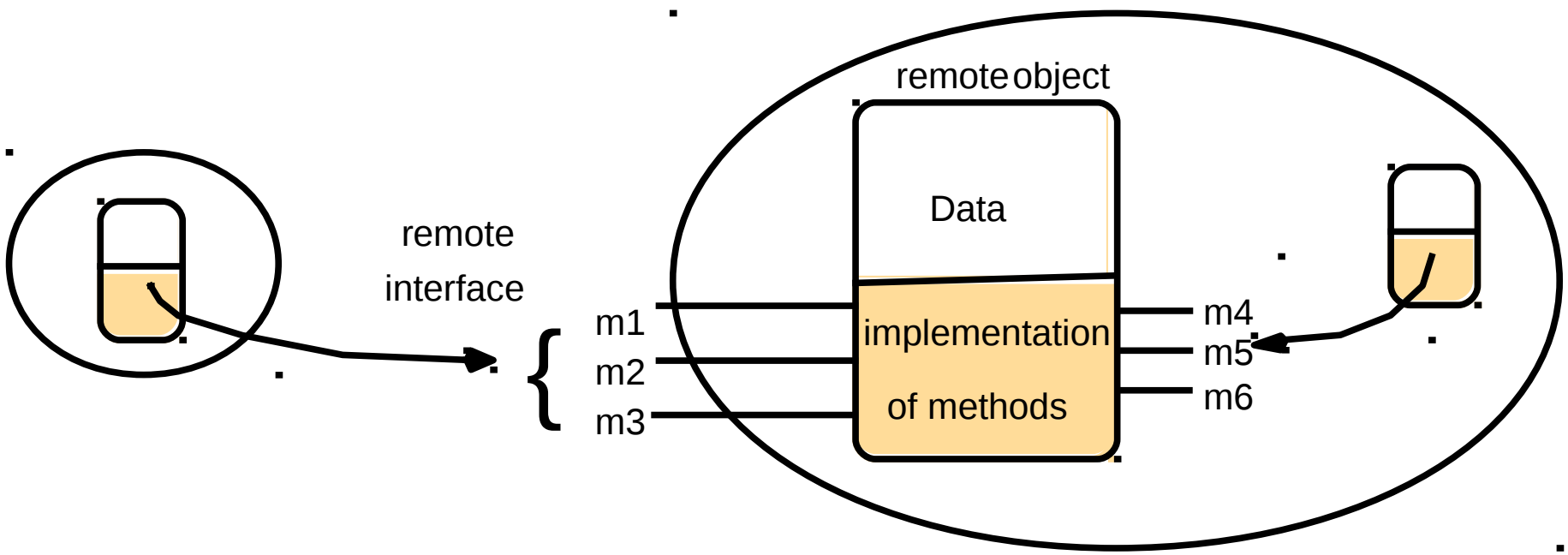
RPC x RMI

- **RPC – programação Procedural**
- **RMI – programação Orientada a Objetos**
- **Parâmetros no RPC são Estruturas de Dados**
- **Parâmetros no RMI são Objetos**

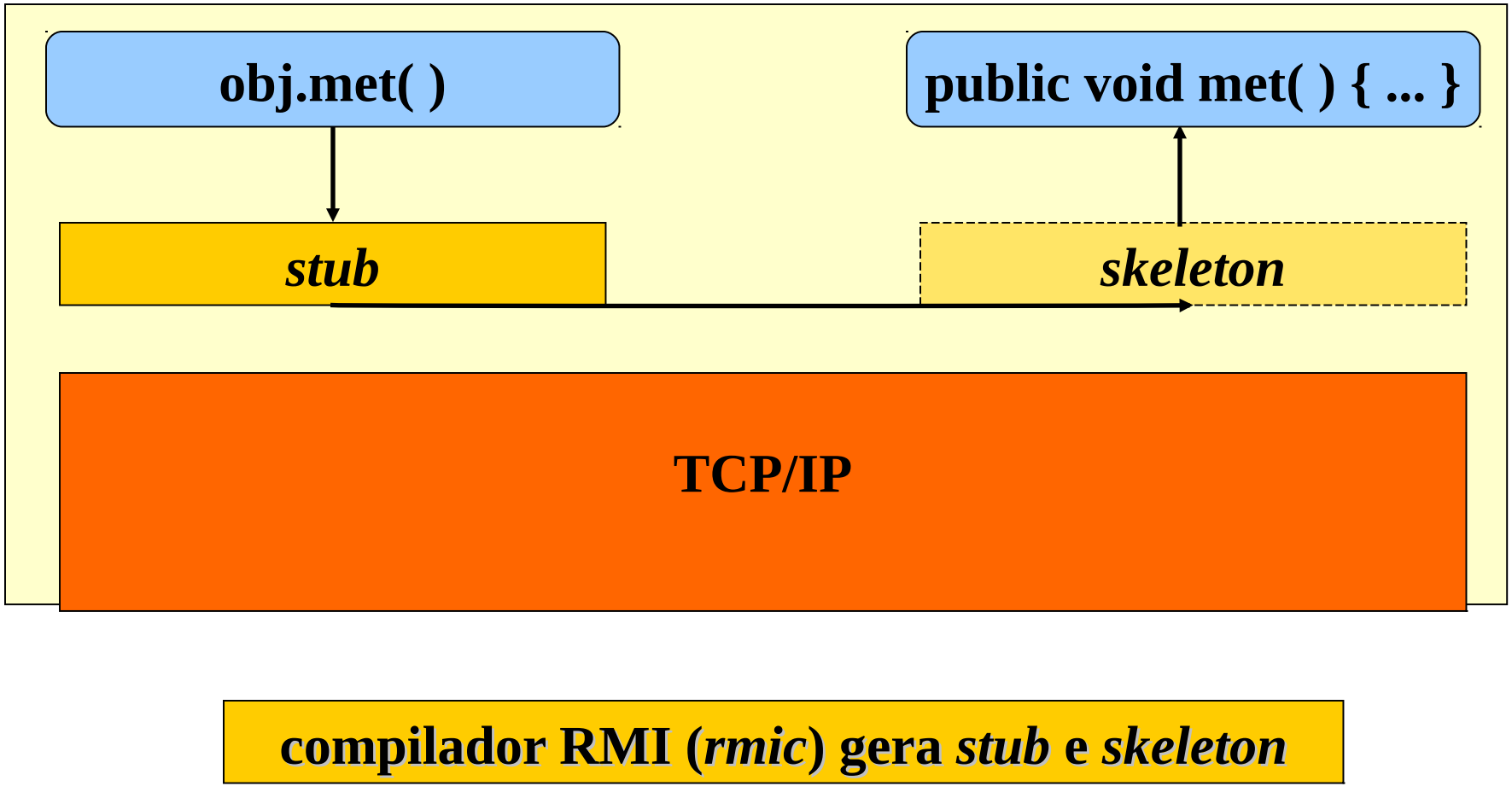
Chamadas locais e remotas



Um objeto com interfaces local e remota



Java RMI – Mecanismo

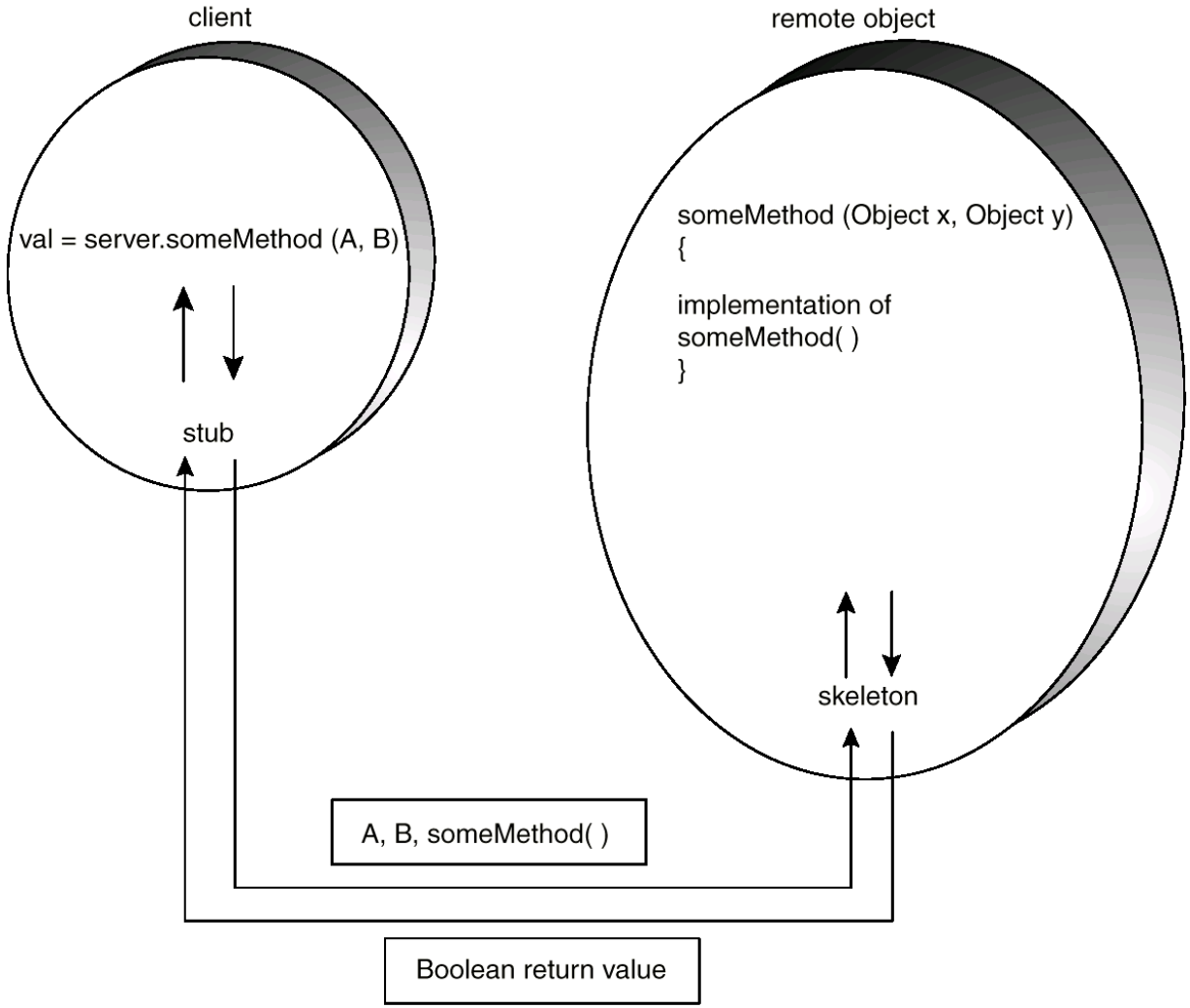


compilador RMI (*rmic*) gera *stub* e *skeleton*

Stubs e Skeletons

- **“Stub” é um Proxy para o Objeto Remoto – Reside no Cliente**
- **O Stub *“Marshalls”* os Parâmetros e os envia para o Servidor.**
- **“Skeleton” está do lado Servidor.**
- **Skeleton *“Unmarshalls”* os parâmetros e os entrega para o Servidor.**

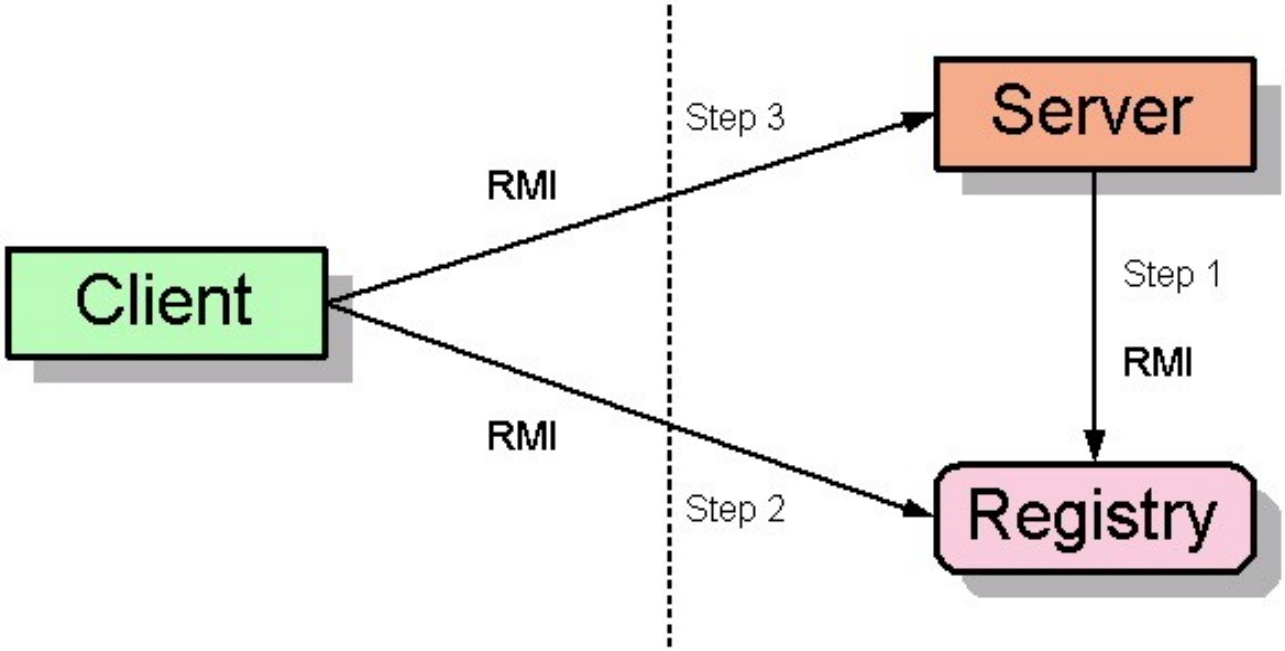
Marshalling Parameters



Java RMI – Mecanismo

- **`Transporte sobre TCP/IP`**
  - **Usa sockets TCP/IP**
  - **Objetos acessados através de portas específicas**

RMI Registry



Java RMI – Mecanismo

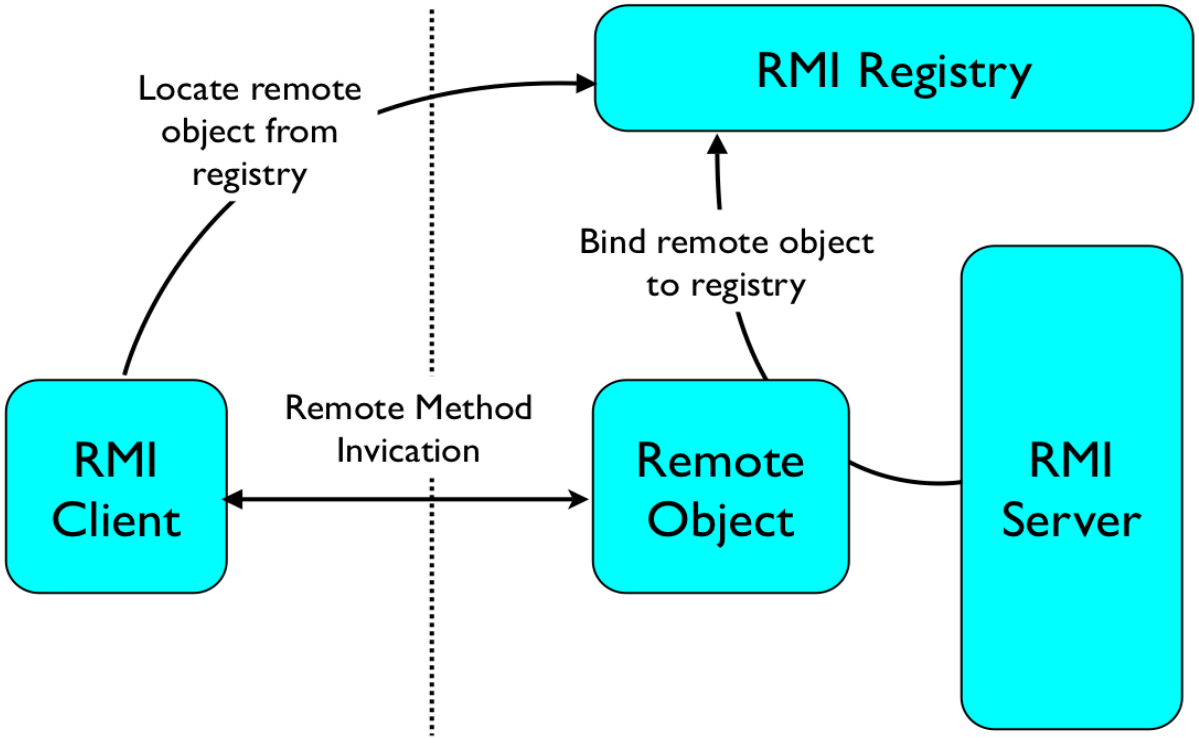
Classe UnicastRemoteObject

- **objetos remotos herdam métodos desta classe**
- **permite que objetos sejam “exportados”**

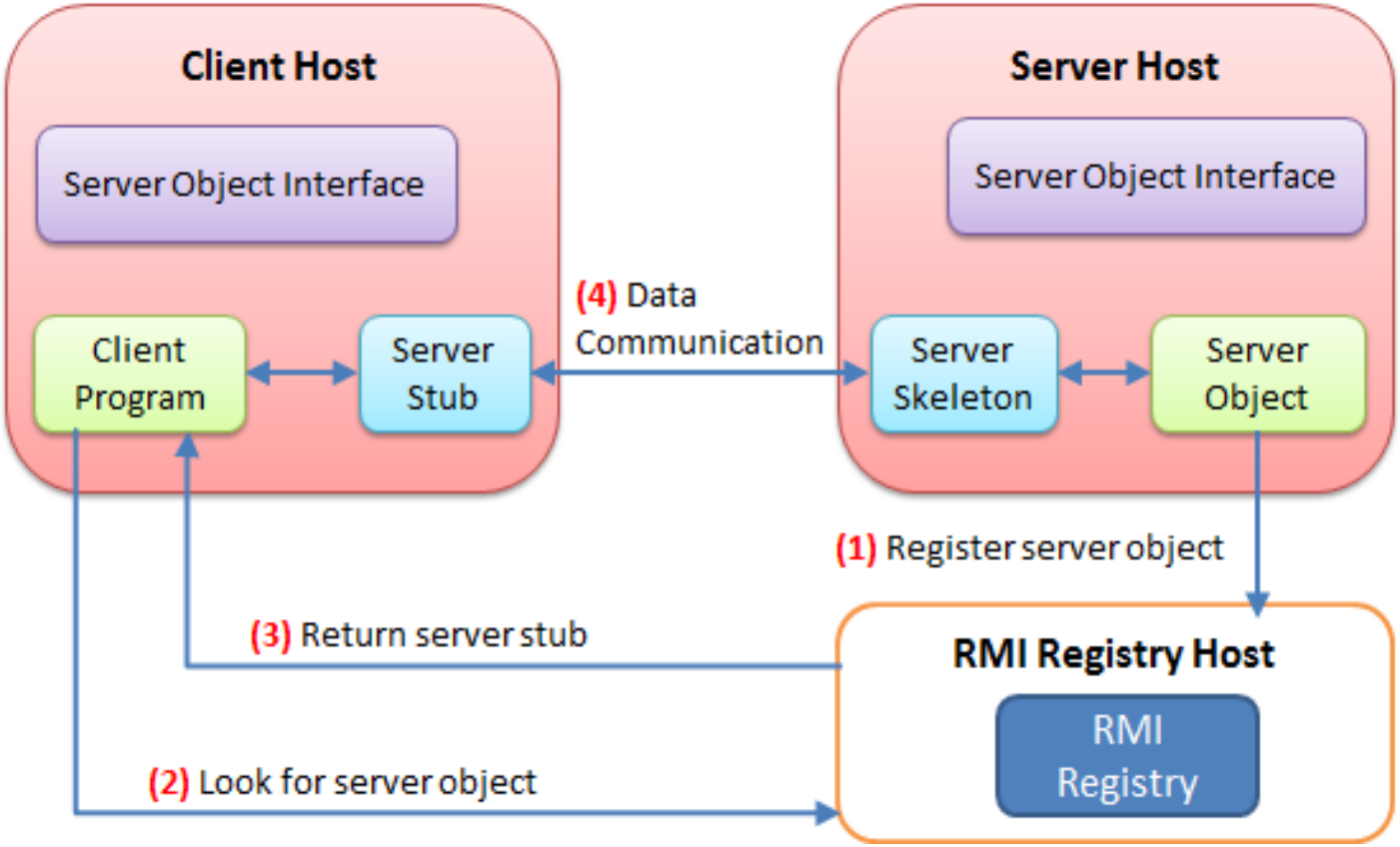
RMI Registry

- **serviço que registra e recupera referências a objetos através do nome**

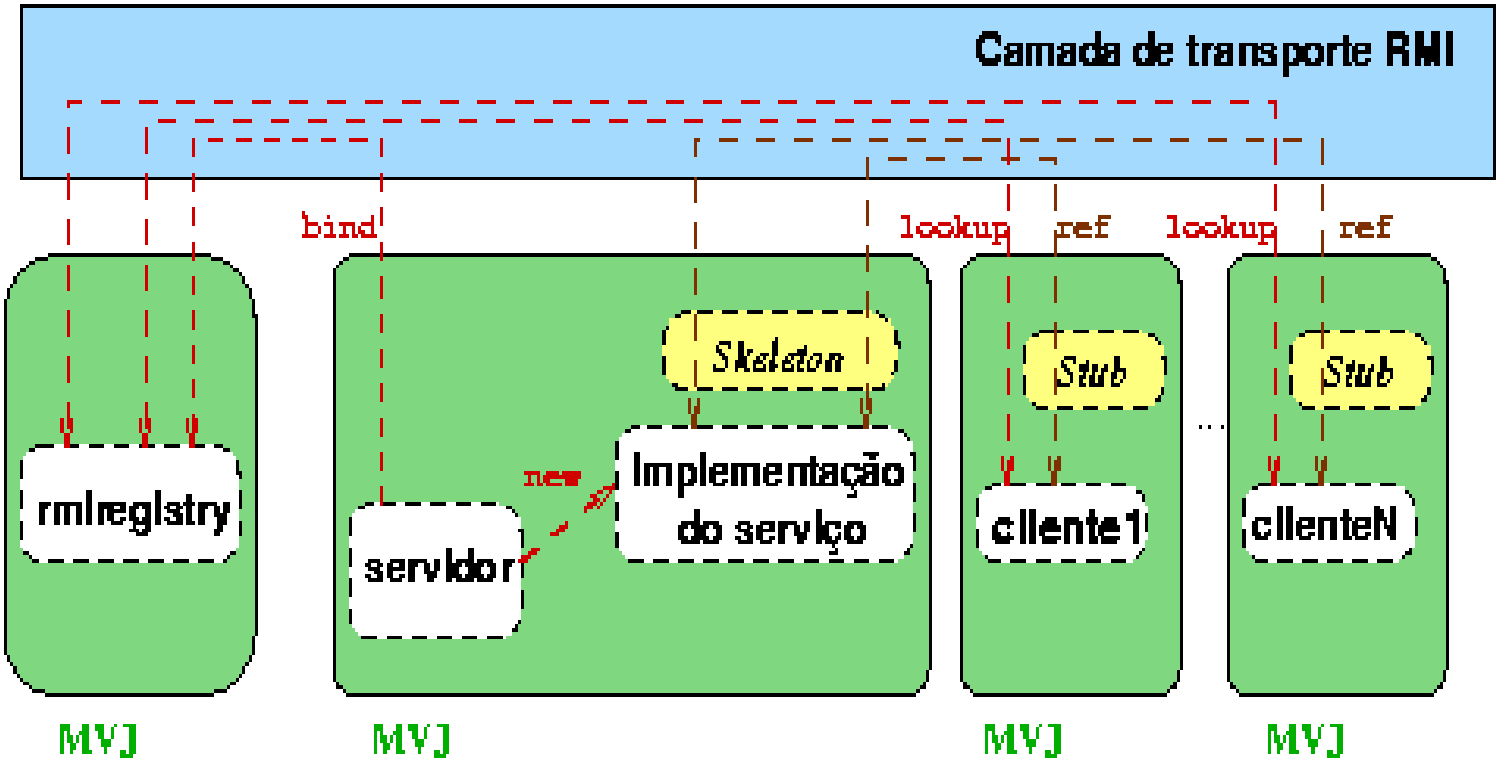
RMI Registry



RMI Registry

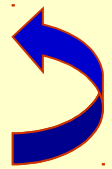


RMI Registry



URL

- **rmi://host:porta/nome**
  - **host default: localhost**
  - **porta default: 1099**
  - **nome: nome dado ao objeto remoto**
  - **Exemplos:**
    - **rmi://grumari:1238/objremoto**
    - **rmi://camboinhas/objremoto**
    - **rmi://localhost:1099/msgrcv**
    - **monitor**



Exemplo: Definição da Interface Remota

```
import java.rmi.Remote;
import java.rmi.RemoteException;

public interface MsgIntf extends Remote {
        String mostraMsg() throws RemoteException;
}
```

Ex: Implementação de Método Remoto

```
import java.rmi.Naming;
import java.rmi.RemoteException;
import java.rmi.server.UnicastRemoteObject;
import java.rmi.registry.Registry;
import java.rmi.registry.LocateRegistry;

public class MsgImpl extends UnicastRemoteObject
                     implements MsgIntf {

    public MsgImpl() throws RemoteException {
           super();
    }

    public String mostraMsg () {
         return ("Ok!");
    }
       .
       .
       .
```

Exemplo: Registro do Objeto Remoto

```
        .
        .
        .
public static void main(String args[]) {
   try {
     LocateRegistry.createRegistry(Registry.REGISTRY_PORT);
     Registry reg = LocateRegistry.getRegistry(“localhost");

     MsgIntf msg = new MsgImpl();
     Naming.rebind("MsgSrv", msg);

     System.out.println("Msg ativo...");

    } catch (Exception e) { }
  }
}
```

Cliente Invocando Método Remoto

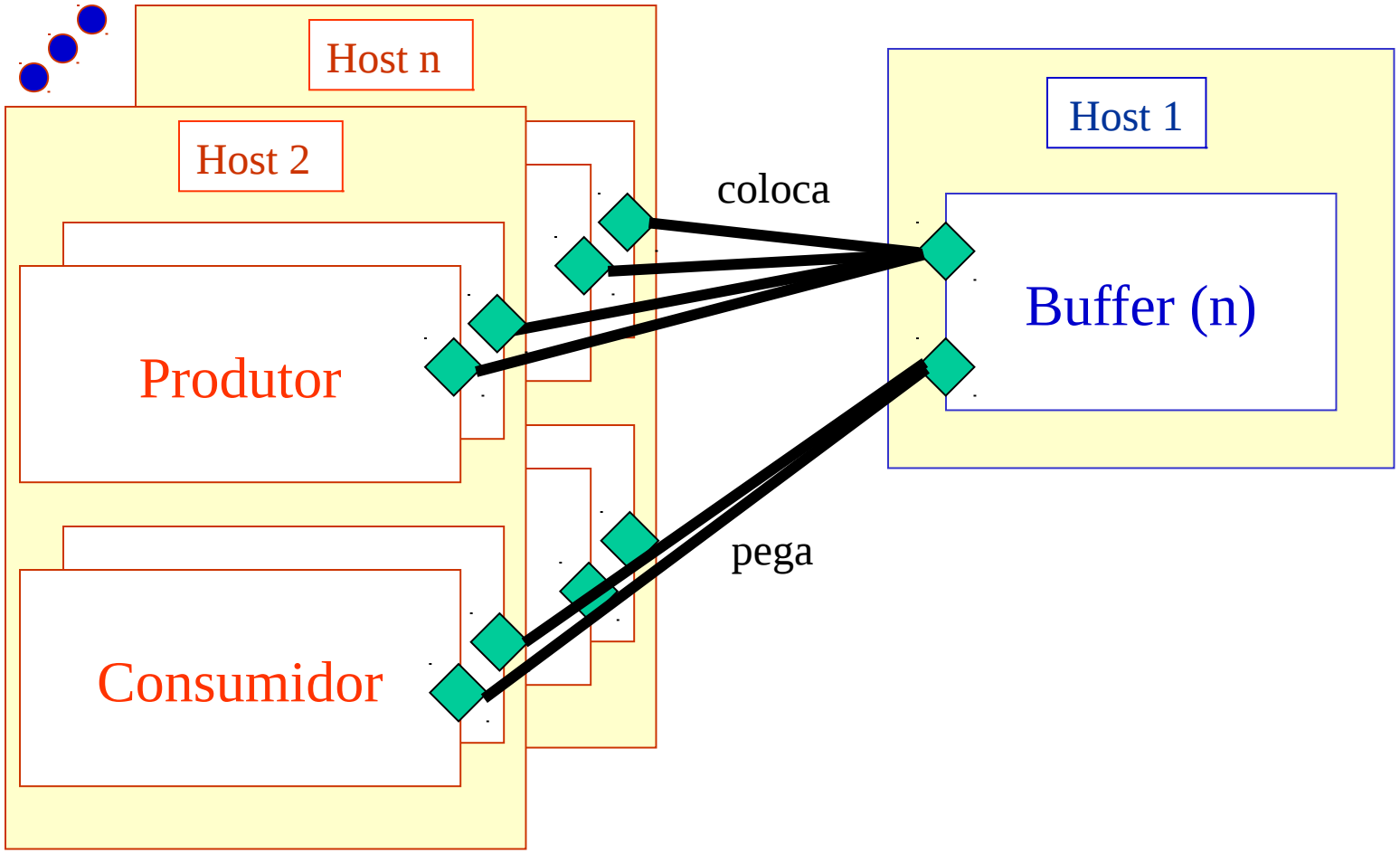
```
import java.rmi.Naming;
import java.rmi.RemoteException;

public class MsgCliente {

    public static void main (String[] args) {
    String mensagem;

        MsgIntf msg = (MsgIntf) Naming.lookup
                                ("rmi://grumari/MsgSrv");
    mensagem = msg.mostraMsg();
        System.out.println (mensagem);
    }
}
```

Utilização do Exemplo

- **Compilar classes e stubs**
  - **javac *.java**
  - **rmic MsgImpl**
- **Inicializar servidor (objeto remoto)**
  - **java MsgImpl**
- **Executar cliente**
  - **java MsgCliente**

Trabalho Prático

- **Problema do Buffer Limitado**



Interface Remota do Monitor

```
import java.rmi.Remote;
import java.rmi.RemoteException;

public interface MonitorIntf extends Remote {
   boolean ocupado = false;

   public void request() throws RemoteException;
   public void release() throws RemoteException;
}
```

Implementação do Monitor Remoto

```
import java.rmi.Naming;
import java.rmi.RemoteException;
import java.rmi.server.UnicastRemoteObject;

public class MonitorImpl extends UnicastRemoteObject

                         implements MonitorIntf {

       private boolean ocupado = false;

       public MonitorImpl() throws RemoteException {
              super();
             }

   .
   .
   .
```

Implementação do Monitor Remoto

```
.
.
.
    public synchronized void request() {
           while (ocupado) {
                  try {
                      wait();
                     } catch (InterruptedException e)
                            { }
                }
           ocupado = true;
          }

    public synchronized void release() {
           ocupado = false;
           notifyAll();
         }      .
.
.
```

Registro do Monitor Remoto

```
                    .
                    .
                    .

    public static void main(String args[]) {
      try {
         MonitorIntf monitor = new MonitorImpl();

         Naming.rebind("rmi://grumari/Monitor",
                       monitor);
         System.out.println("Monitor ativo... ");

        }
      catch (Exception e) { }
     }
}
```

Criação das Threads

```
import java.rmi.Naming;
import java.rmi.RemoteException;

public class RequestRelease {
  public static void main(String[] args) {
    MonitorIntf m = null;

    try {
      m = (MonitorIntf) Naming.lookup(("rmi://grumari/Monitor");
    } catch (Exception e) { }

    Usuario us_1 = new Usuario(m);
    Usuario us_2 = new Usuario(m);

    us_2.start();
    us_1.start();
   }
}
```

Definição da Classe Usuário (*Thread*)

```
import java.rmi.RemoteException;

public class Usuario extends Thread {
   private MonitorIntf monitor;

   public Usuario (MonitorIntf m) {
      monitor = m;
   }

   public void run() {
      try {
          monitor.request();
              ... usa recurso ...
          monitor.release();
         } catch (RemoteException e) {}
     }
  }
```

Utilização do Exemplo

- **Inicializar serviço de nomes**
  - **rmiregistry**
- **Compilar classes e stubs**
  - **javac *.java**
  - **rmic MonitorImpl**
- **Inicializar monitor remoto**
  - **java MonitorImpl**
- **Executar usuários**
  - **java RequestRelease**

Créditos

Prof. Sérgio T. Carvalho

sergio@inf.ufg.br