# RMI

Veerle Ongenae

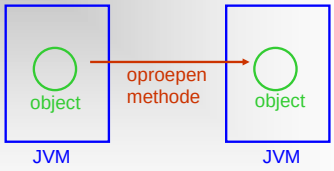
## Inhoud

- Wat is RMI?
- Remote Service maken
- Client
- Voorbeeld



## RMI

- Remote Method Invocation
- Java

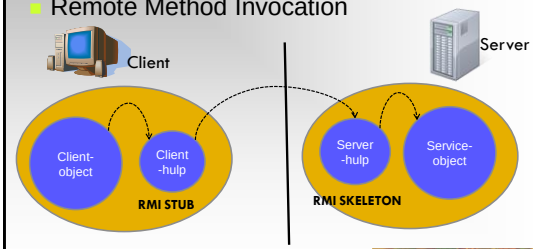



- Communicatie tussen Java-programma's op afstand



## RMI

- Remote Method Invocation






## RMI-applicatie

- Gedistribueerde object applicatie
  - Server
    - Maakt objecten aan
    - Biedt referentie naar die objecten aan
    - Wacht op clients
      - Oproepen methodes objecten
  - Client
    - Verkrijgt referentie naar objecten van de server
    - Roept methode op
- RMI
  - Bepaalt communicatie tussen client en server



## Stub

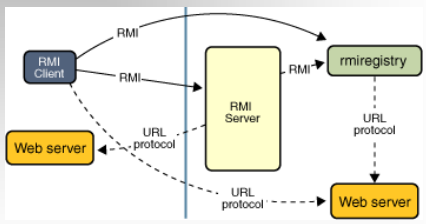
- RMI bezorgt de oproepende JVM een stub voor het remote object
  - GEEN KOPIE
- Stub
  - Lokale voorstelling van remote object (proxy)
  - Remote reference
- Client
  - Oproepen methode stub
    - Roept methode remote object op

## RMI-client

- Taken
  - Object op afstand lokaliseren
  - Communicatie met object op afstand
  - class-definities laden van objecten die doorgestuurd worden
    - Argumenten
    - Resultaat



## Structuur RMI-applicatie




Bron: java.sun.com

Web server: laden class-definities



## Remote objects

- Remote objects
  - Objecten met methodes
    - Kunnen uitgevoerd worden vanuit een andere JVM
- Implementeren een remote interface
  - Interface afgeleid van java.rmi.Remote
  - Elke methode van de interface
    - throws java.rmi.RemoteException



## Inhoud

- Wat is RMI?
- Structuur RMI-applicatie
- Ontwikkelen RMI-applicatie



## Inhoud

- Wat is RMI?
- Remote Service maken
- Client
- Voorbeeld



## Remote service maken

- Remote interface maken
  - Methodes die client kan aanroepen
- Implementatie van de remote interface
  - Object waarvan de client methodes wil oproepen
- Hulpobjecten genereren
  - Stubs en skeletons
- RMI registry starten
  - Hier haalt de client zijn proxy
- Remote Service starten
  - Service-object instantiëren en registreren

## Remote Interface maken

- Interface java.rmi.Remote uitbreiden
  - Geen methode
  - Wel speciale betekenis
- Methoden voor client declareren
  - Gooien RemoteException
    - Door I/O- of netwerkfouten
- Alle argumenten en teruggeefwaarden zijn primitief of serialiseerbaar
  - Verstuurd over het netwerk

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

serialiseerbaar



## Transport objecten

- Mogelijke argumenten/resultaten van remote methods
  - Primitieve types
  - Remote objecten
  - Serialiseerbare objecten
- Doorgeven objecten aan remote method
  - Remote objecten: passed by reference
  - Lokale objecten: passed by copy



## Remote service maken

- Remote interface maken
  - Methodes die client kan aanroepen
- Implementatie van de remote interface
  - Object waarvan de client methodes wil oproepen
- Hulpobjecten genereren
  - Stubs en skeletons
- RMI registry starten
  - Hier haalt de client zijn proxy
- Remote Service starten
  - Service-object instantiëren en registreren



## Implementatie van de remote interface

- Remote interface implementeren
  - Object dat eigenlijke werk doet maken

```
import java.rmi.*;
import java.rmi.server.*;
public class MyRemoteImpl implements
    MyRemote {
    public MyRemoteImpl() { }
    public String sayHello() {
        return "server zegt: 'hallo'";
    }
    public static void main (String[] args) {
        try {
            MyRemote service =
                new MyRemoteImpl();
            …
        } catch (Exception e) {…}
    }
}
```



## Remote service maken

- Remote interface maken
  - Methodes die client kan aanroepen
- Implementatie van de remote interface
  - Object waarvan de client methodes wil oproepen
- Hulpobjecten genereren
  - Stubs en skeletons
- RMI registry starten
  - Hier haalt de client zijn proxy
- Remote Service starten
  - Service-object instantiëren en registreren



## Hulpobjecten genereren

- Stub aanmaken
- Aanmelden bij RMI registry
  - Beschikbaar stellen voor clients
  - Stub in registry

```
import java.rmi.*;
import java.rmi.server.*;
public class MyRemoteImpl implements MyRemote  {
    public MyRemoteImpl() { }
    public String sayHello() { return "server zegt: 'hallo'";}
    public static void main (String[] args) {
        try {
            MyRemote service = new MyRemoteImpl();
            MyRemote stub = (MyRemote )
                UnicastRemoteObject.exportObject(obj, 0);

            Registry registry = LocateRegistry.getRegistry();
            registry.bind("Hello", stub);
        } catch (Exception e) {…}
    }
}
```

## Remote object beschikbaar maken

- Exporteren → binnenkomende aanvragen ontvangen
  - Methode van java.rmi.server.UnicastRemoteObject

**public static Remote exportObject(Remote obj, int port) throws RemoteException**

- Tweede argument: poort
  - 0: anoniem, RMI of systeem kiest
- Resultaat: stub (vb. Compute)
  - Enkel remote methods



## Registratie remote objects

- java.rmi.registry.LocateRegistry
  - Registratie object vinden (standaardpoort)

```
Registry registry = LocateRegistry.getRegistry();
```

  - Registratie object creëren
- java.rmi.registry.Registry
  - Interface
  - Bewaren en vinden referenties naar remote objects

```
void bind(String name,Remote obj) throws RemoteException,
    AlreadyBoundException, AccessException
void rebind(String name, Remote obj) throws
    RemoteException, AccessException
```

  - Referentie naar een remote object koppelen aan een naam in het register



## Remote service maken

- Remote interface maken
  - Methodes die client kan aanroepen
- Implementatie van de remote interface
  - Object waarvan de client methodes wil oproepen
- Hulpobjecten genereren
  - Stubs en skeletons
- RMI registry starten
  - Hier haalt de client zijn proxy
- Remote Service starten
  - Service-object instantiëren en registreren



## RMI registry starten

- Commandolijn

  rmiregistry
  - Let op!
    - Starten in map die toegang heeft tot je klassen



## Remote service maken

- Remote interface maken
  - Methodes die client kan aanroepen
- Implementatie van de remote interface
  - Object waarvan de client methodes wil oproepen
- Hulpobjecten genereren
  - Stubs en skeletons
- RMI registry starten
  - Hier haalt de client zijn proxy
- Remote Service starten
  - Service-object instantiëren en registreren



## Remote Service starten

- Commandolijn

  java MyRemoteImpl

## Inhoud

- Wat is RMI?
- Remote Service maken
- Client
- Voorbeeld



## Client van Remote Service

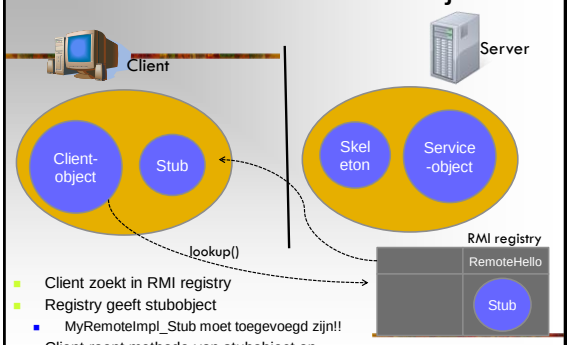
- Hoe komt client aan STUB?
- Client zoekt in RMI registry

```
import java.rmi.*;
public class MyRemoteClient {
    public void go(String host) {
        try {
            Registry registry = LocateRegistry.getRegistry(host);
            MyRemote service = (MyRemote) registry.lookup("Hello");
            String resultaat = service.sayHello();
        } catch (Exception e) {…}

    }
    public static void main (String[] args) {
        new MyRemoteClient().go(arg[0]);
    }
}
```



## Hoe komt client aan stubobject?




- Client zoekt in RMI registry
- Registry geeft stubobject
  - MyRemoteImpl_Stub moet toegevoegd zijn!!
- Client roept methode van stubobject op



## Referentie naar registerobject op server ophalen

- java.rmi.registry.LocateRegistry
  - Registratie object vinden (standaardpoort)

**public static Registry getRegistry(String host)**
**throws RemoteExceptionegistry**



## Referentie naar remote object opzoeken in registerobject

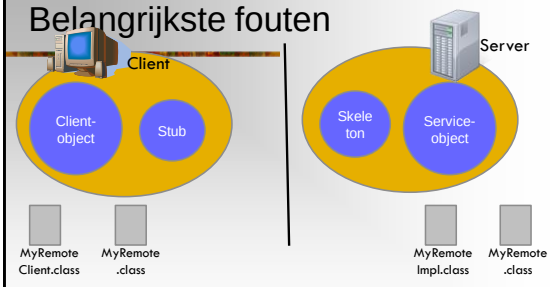
- java.rmi.registry.Registry

**Remote lookup(String name) throws RemoteException, NotBoundException, AccessException**

- Referentie naar een remote object bepalen op basis van de naam in het register



## Belangrijkste fouten




- RMI registry vergeten starten voor remote service
- Serialiseren argumenten en teruggeefwaarden

## Inhoud

- Wat is RMI?
- Remote Service maken
- Client
- Voorbeeld



## Ontwikkelen RMI-applicatie

- Voorbeeld
- example.hello.Hello
- example.hello.Server
- example.hello.Client
- http://java.sun.com/j2se/1.5.0/docs/guide/rmi/hello/hello-world.html



## Opruimen remote object

- Programma eindigt
  - Remote object blijft bestaan zolang er een referentie naar is in een andere JVM
- ComputerEngine-object verdwijnt
  - Verwijdert uit register
  - Én
  - Geen remote references meer bij de client



## Informatie

- Java tutorial (http://java.sun.com/docs/books/tutorial/rmi/)