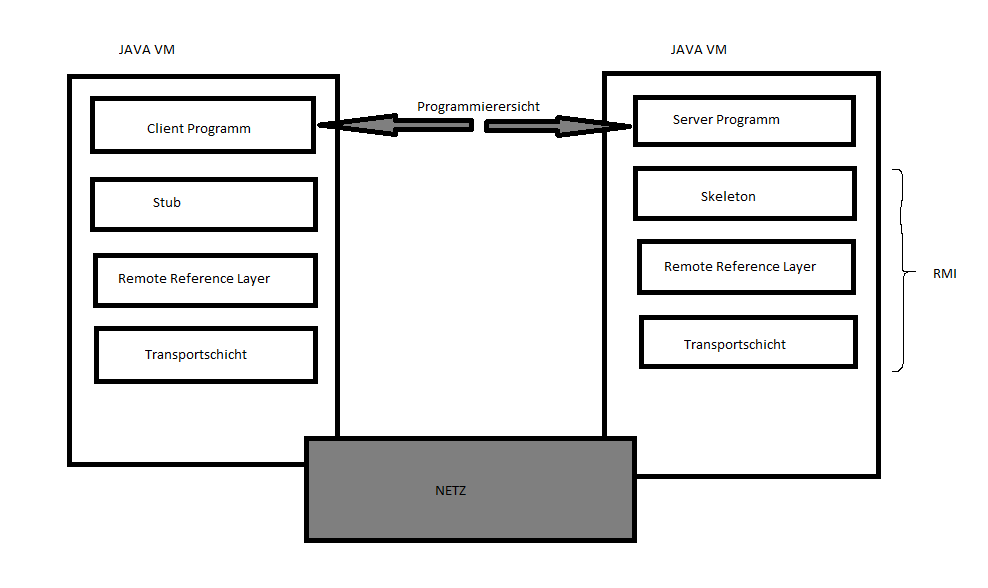
RMI Arbeitsauftrag

**Remote Method Invocation** (RMI, deutsch etwa „Aufruf entfernter Methoden“), gelegentlich auch als Methodenfernaufruf bezeichnet, ist der Aufruf einer [Methode](https://de.wikipedia.org/wiki/Objektorientierte_Programmierung) eines entfernten [Java](https://de.wikipedia.org/wiki/Java_(Programmiersprache))-[Objekts](https://de.wikipedia.org/wiki/Objektorientierte_Programmierung). „Entfernt“ bedeutet dabei, dass sich das Objekt in einer anderen [Java Virtual Machine](https://de.wikipedia.org/wiki/Java_Virtual_Machine) befinden kann, die ihrerseits auf einem entfernten Rechner oder auf dem lokalen Rechner laufen kann. Dabei sieht der Aufruf für das aufrufende Objekt (bzw. dessen Programmierer) genauso aus wie ein lokaler Aufruf, es müssen jedoch besondere [Ausnahmen](https://de.wikipedia.org/wiki/Ausnahmebehandlung) (Exceptions) abgefangen werden, die zum Beispiel einen Verbindungsabbruch signalisieren können.

Auf der [Client](https://de.wikipedia.org/wiki/Client)-Seite kümmert sich der sogenannte **[Stub](https://de.wikipedia.org/wiki/Stub_(Programmierung)" \o "Stub (Programmierung))** um den Netzwerktransport. Der Stub muss entweder lokal oder über das Netz für den Client verfügbar sein. Das Erstellen des Stubs wird von der Java Virtual Machine übernommen.

Auf der Server-Seite kümmert sich der sogenannte **Skeleton** um den Netzwerktransport.

**Codebase:**

Es wird die URL angeben, die auf die Klassen des Servers zeigt, z.B. file:/home/schoepp/rmi/ (letztes ‘/’ ist wichtig!) Die URL muss man angeben, damit die RMI-Implementierung die Stubs aus den Klassen des Servers erzeugen kann. Bei dem Client gilt dasselbe.

**Hostname:**

Die IP-Adresse/Hostname des Servers wird festgelegt. Damit kann auf den Server zugegriffen  
werden. (Standard: localhost).

**useCodebaseOnly:**

Falls die Eigenschaft java.rmi.server.useCodebaseOnly auf true gesetzt ist, dann ladet JDE nur die Klassen von jener Location welche unter CLASSPATH angegeben ist oder welche in der java.rmi.server.codebase gesetzt ist. Dies ist ein einfacher Weg um weitere Sicherheit zu bieten.

**java.rmi.security.policy**

Gibt das Policy-File an, welches die Rechte die es zum Ausführen/Zugreifen des Remote Server Objektes brauch.

Um sich gegen Einschleusen schädlichen Codes in den Client zu schützen, kann die Ausführung von einem Security Manager kontrolliert werden. Hierzu muss eine Policy-Datei angelegt werden. Dies bedeutet mehr Aufwand, mehr Code und mehr Rechenleistung, ist aber dafür sicherer.

Quellen:

<https://books.google.it/books?id=MZq4CQAAQBAJ&pg=PA138&lpg=PA138&dq=java+rmi+hostname+erkl%C3%A4rung&source=bl&ots=V82Rnmcqqu&sig=iK2rimx0NVLrTjfWo7EAPpsCero&hl=de&sa=X&ved=0ahUKEwiDlonxiPbSAhVBDxQKHW8mDiEQ6AEIUDAG#v=onepage&q&f=false>

http://www.ti.uni-tuebingen.de/fileadmin/assets/csp\_ws0809/aufgabe2/aufgabe2a.pdf