

**Laborprotokoll**

**Verteilte Objekte mit RMI**

**Systemtechnik Labor**

**4CHIT 2015/16, Gruppe C**

**Johannes Ucel**

|  |  |
| --- | --- |
|  | **Version 1.0** |
| **Note:** | **Begonnen am 1. April 2016** |
| **Betreuer: Prof. Michael Borko** | **Beendet am 1. April 2016** |

Inhaltsverzeichnis

[1 Einführung 3](#_Toc447291734)

[1.1 Ziele 3](#_Toc447291735)

[1.2 Voraussetzungen 3](#_Toc447291736)

[1.3 Aufgabenstellung 3](#_Toc447291737)

[2 Quellen 4](#_Toc447291738)

[3 Theorie 4](#_Toc447291739)

[3.1 RMI 4](#_Toc447291740)

[3.2 Policy 4](#_Toc447291741)

[3.3 ANT 4](#_Toc447291742)

[3.4 Umgebungsvariablen 4](#_Toc447291743)

[4 Ergebnisse 5](#_Toc447291744)

[4.1 Java Tutorial 5](#_Toc447291745)

[4.2 Implementierung des Command - Patterns 5](#_Toc447291746)

[4.3 Implementierung des Client - Callbacks 5](#_Toc447291747)

# Einführung

*„Verteilte Objekte haben bestimmte Grunderfordernisse, die mittels implementierten Middlewares leicht verwendet werden können. Das Verständnis hinter diesen Mechanismen ist aber notwendig, um funktionale Anforderungen entsprechend sicher und stabil implementieren zu können.“* [BOR16]

## Ziele

*„Diese Übung gibt eine einfache Einführung in die Verwendung von verteilten Objekten mittels Java RMI. Es wird speziell Augenmerk auf die Referenzverwaltung sowie Serialisierung von Objekten gelegt. Es soll dabei eine einfache verteilte Applikation in Java implementiert werden.“*[BOR16]

## Voraussetzungen

*„Grundlagen Java und Software-Tests*

*Grundlagen zu verteilten Systemen und Netzwerkverbindungen*

*Grundlegendes Verständnis von nebenläufigen Prozessen“*[BOR16]

## Aufgabenstellung

*„Folgen Sie dem offiziellen Java-RMI Tutorial, um eine einfache Implementierung des PI-Calculators zu realisieren. Beachten Sie dabei die notwendigen Schritte der Sicherheitseinstellungen (SecurityManager) sowie die Verwendung des RemoteInterfaces und der RemoteException.*

*Implementieren Sie ein Command-Pattern [2] mittels RMI und übertragen Sie die Aufgaben/Berechnungen an den Server. Sie können am Client entscheiden, welche Aufgaben der Server übernehmen soll. Die Erweiterung dieser Aufgabe wäre ein Callback-Interface auf der Client-Seite, die nach Beendigung der Aufgabe eine entsprechende Rückmeldung an den Client zurück senden soll. Somit hat der Client auch ein RemoteObject, welches aber nicht in der Registry eingetragen wird sondern beim Aufruf mittels Referenz an den Server übergeben wird.“* [BOR16]

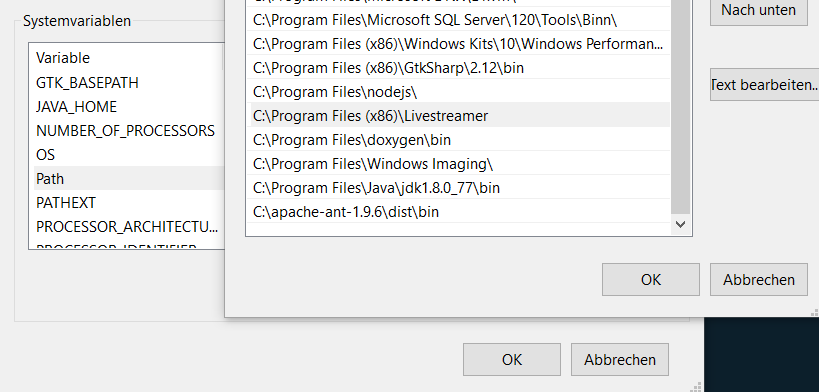
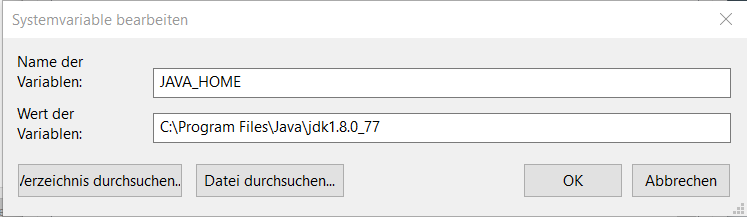
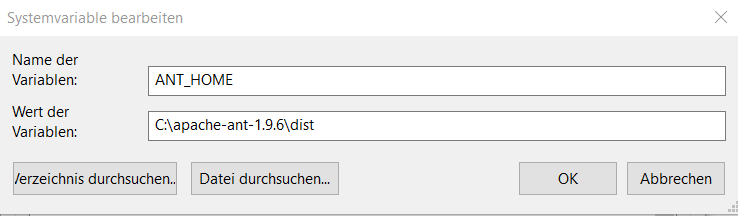
# Theorie

## RMI

## Policy

## ANT

## Umgebungsvariablen



# Ergebnisse

GIT – Repository:  
<https://github.com/TGM-HIT/Ucel_RMI>

## Java Tutorial

## Implementierung des Command - Patterns

## Implementierung des Client – Callbacks

## Probleme und Fehler

* Umgebungsvariablen
* Policy

# Quellen

[1] "The Java Tutorials - Trail RMI"; online: <http://docs.oracle.com/javase/tutorial/rmi/>

[2] "Command Pattern"; Vince Huston; online: <http://vincehuston.org/dp/command.html>

[3] "Beispiel Konstrukt für Command Pattern mit Java RMI"; Michael Borko; online: <https://github.com/mborko/code-examples/tree/master/java/rmiCommandPattern>

[BOR16] Michael Borko. (2016, April). DezSys-04 "Verteilte Objekte mit RMI" [Online]. Available at: https://elearning.tgm.ac.at/mod/assign/view.php?id=47277 [abgerufen am 01.04.2016]