# Corba-Übung

**TGM / 4AHIT**
**1200 WIEN - WEXSTRAßE**
January 8, 2014
Verfasst von: Bobek, Willinger

## Inhaltsverzeichnis

## Aufgabenstellung

Verwenden Sie das Paket ORBacus oder omniORB bzw. JacORB um Java und C++ ORB-Implementationen zum Laufen zu bringen.

Passen Sie eines der Demoprogramme so an, dass Sie einen Namingservice verwenden, welches ein Objekt anbietet, das von jeweils einer anderen Sprache (Java/C++) verteilt angesprochen wird. Beachten Sie dabei, dass eine IDL-Implementierung vorhanden ist um die unterschiedlichen Sprachen abgleichen zu können.

Vorschlag: Verwenden Sie für die Implementierungsumgebung eine Linux-Distribution, da eine optionale Kompilierung einfacher zu konfigurieren ist.

**Resources**

http://omniorb.sourceforge.net/
http://www.microfocus.com/products/corba/orbacus/
http://www.jacorb.org/
http://omniorb.sourceforge.net/omni41/omniORB.pdf
http://www.ing.iac.es/~docs/external/corba/book.pdf

# Struktur

## Testaufbau

**10.0.104.115 - Willinger's PC (Nameserver & C++ Server)**
**10.0.104.192 - Bobek's PC (Java Client)**

## Verzeichnisse

```
./CORBA-1/ (Stamm)
-- c++ (c++ server)
---- GNUmakefile
---- corba1_server.cc
-- idl (IDL Files)
---- echo.idl
-- java (Java Client)
---- build.xml (für Ant)
---- src
------- corba1_client.java
```

# Installation

## omniORB

### Herunterladen & Entpacken

```
sudo apt-get install python python-dev ant openjdk-7-jdk

wget
http://sourceforge.net/projects/omniorb/files/omniORB/omniORB-
4.1.7/omniORB-4.1.7.tar.bz2/download
mv download omniORB-4.1.7.tar.bz2
bunzip2 omniORB-4.1.7.tar.bz2
tar xfv omniORB-4.1.7.tar
```

### Konfigurieren & Installieren

```
cd omniORB-4.1.7/
mdkir build
cd build/
../configure
make
make install
```

### Log- & config-Verzeichnis anlegen

```
mkdir $HOME/omniorb
mkdir $HOME/omniorb/log
mkdir $HOME/omniorb/etc
cd ..
```

### Beispiel Konfiguration kopieren

```
cp sample.cfg $HOME/omniorb/etc/omniORB.cfg
```

### Konfiguration anpassen

```
vi $HOME/omniorb/etc/omniORB.cfg
```

Anschließend wurde die Datei wie folgt verändert (wir haben nur die bearbeiteten Teile hier angeschrieben, da die Datei ca. 1000 Zeilen hat). Im Folgenden wurden gesetzt: Debug Level, NameServer Referenz, sowie supportBootstrapAgent damit unser Java Client akzeptiert wird. (Standardmäßig aus)

```
{…}
traceLevel = 5
{…}
traceExceptions = 1
{…}
traceInvocations = 1
{…}
traceInvocationReturns = 1
{…}
traceThreadId = 1
{…}
traceTime = 1
{…}
InitRef = NameService=corbaloc::localhost:2809/NameService
{…}
supportBootstrapAgent = 1
{…}
```

### Umgebungsvariablen hinzufügen

```
vi  ~/.bashrc
```

```
{…}
OMNINAMES_LOGDIR=$HOME/omniorb/log;
export OMNINAMES_LOGDIR

OMNIORB_CONFIG=$HOME/omniorb/etc/omniORB.cfg;
export OMNIORB_CONFIG;

LD_LIBRARY_PATH=/home/schueler/downloads/omniORB-
4.1.7/build/lib;
export LD_LIBRARY_PATH;
```

Anschließend starteten wir omniNames zum ersten Mal, um die Funktion zu überprüfen und das Log File anzulegen.

```
omniNames -start
```

### JacORB

Die Installation von JacORB ist schnell erledigt, wir verwenden die aktuellste Version aus dem Github-Repo:

```
git clone https://github.com/JacORB/JacORB.git
cd JacORB/
ant
```

### PATH Variable anpassen

```
vi ~/.bashrc
```

```
{…}
PATH=$HOME/downloads/JacORB/bin":$PATH;
```

# Kompilierung

## omniOrb

### „Echo“ Beispiel kopieren

```
cd $HOME/downloads/omniORB-4.1.7/src/examples/echo
cp -R *  ~/CORBA-1/c++/
cp $HOME/downloads/omniORB-4.1.7/idl/echo.idl ~/CORBA-1/idl
```

**Makefile anpassen:**

Wir haben unser Makefile angepasst, damit es nicht von den Standard-Makefiles (inkludiert) abhängt.

```
vi GNUmakefile
```

```
LIB_DIR=$(HOME)/downloads/omniORB-4.1.7/build/lib
INCLUDE_DIR=$(HOME)/downloads/omniORB-4.1.7/build/include

all:: check_dir prebuild_idl prebuild_server build_idl build

clean::
        rm -f *.o *.d corba1_server
        rm -R stub/

check_dir:
        -mkdir stub/

prebuild_idl:
        omniidl -bcxx -Wba -C./stub  ../idl/echo.idl

build_idl:
        cd stub/; omkdepend -D__cplusplus -D__GNUG__ -
D__GNUC__ -D__OMNIORB4__ -D_REENTRANT -I. -I. -I$(INCLUDE_DIR)
-D__OSVERSION__=2 -D__linux__ -D__x86_64__ echoDynSK.cc
echoSK.cc; cd -
        cd stub/; g++ -c -O2 -Wall -Wno-unused -fexceptions  -
D__OMNIORB4__ -I../stub -D_REENTRANT -I. -I. -I$(INCLUDE_DIR)
-D__OSVERSION__=2 -D__linux__ -D__x86_64__ -o echoSK.o
echoSK.cc; cd -

prebuild_server:
        omkdepend -D__cplusplus -D__GNUG__ -D__GNUC__ -
D__OMNIORB4__ -D_REENTRANT -I. -I./ -I$(INCLUDE_DIR) -Istub/ -
D__OSVERSION__=2 -D__linux__ -D__x86_64__ corba1_server.cc
        g++ -c -O2 -Wall -Wno-unused -fexceptions  -
D__OMNIORB4__ -D_REENTRANT -I. -I./ -I$(INCLUDE_DIR) -Istub/ -
D__OSVERSION__=2 -D__linux__ -D__x86_64__ -o corba1_server.o
corba1_server.cc

build:
        rm -f corba1_server
        g++ -o corba1_server -O2 -Wall -Wno-unused -
fexceptions -L$(LIB_DIR) corba1_server.o stub/echoSK.o -
lomniORB4 -lomnithread -lpthread
```

### JacORB

Wir müssen die Konfigurationsdatei so anpassen, dass sie zum korrekten NameServer verbindet.

```
vi ~/orb.properties
```

```
[..]
ORBInitRef.NameService=corbaloc::10.0.104.115:2809/NameService
[..]
```

# Testdurchführung

## Programme kompilieren

### C++ Server

```
cd $HOME/CORBA-1/c++
!! Pfade im Makefile müssen zur kompilierten OmniORB Source zeigen(build/)!
make
```

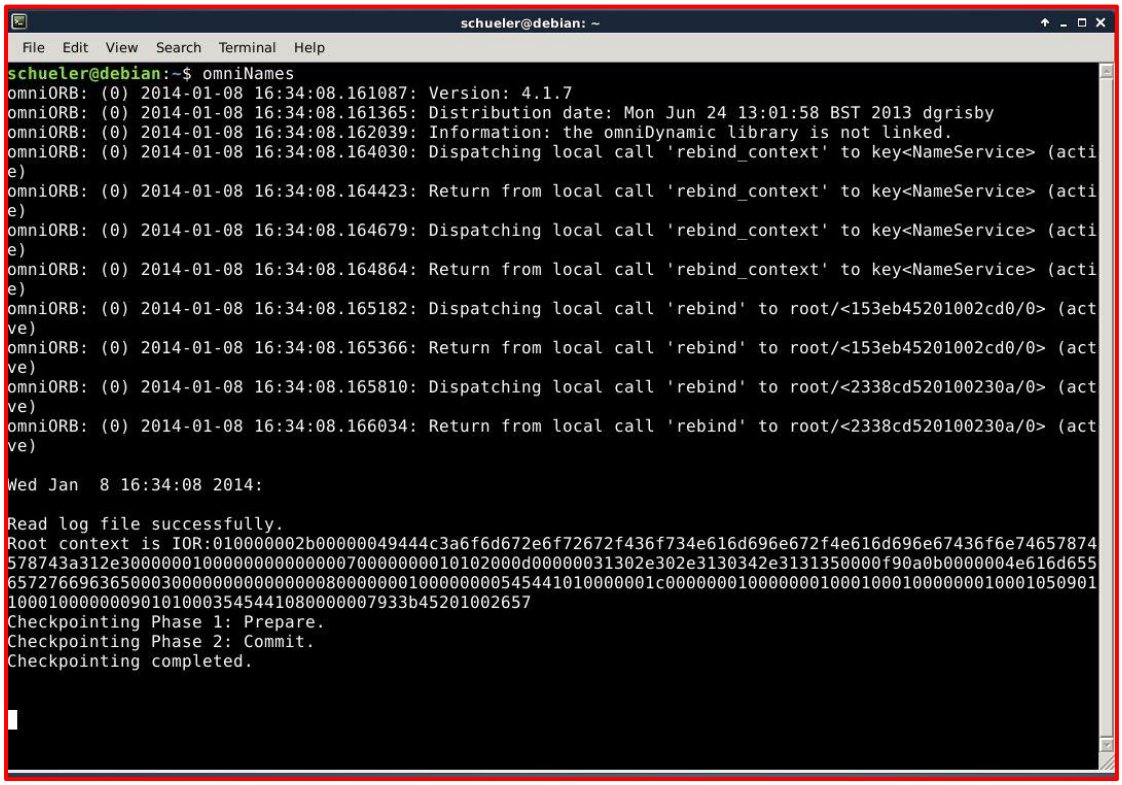
### Java-Client

```
cd ../java/
ant
```

## Ausführen

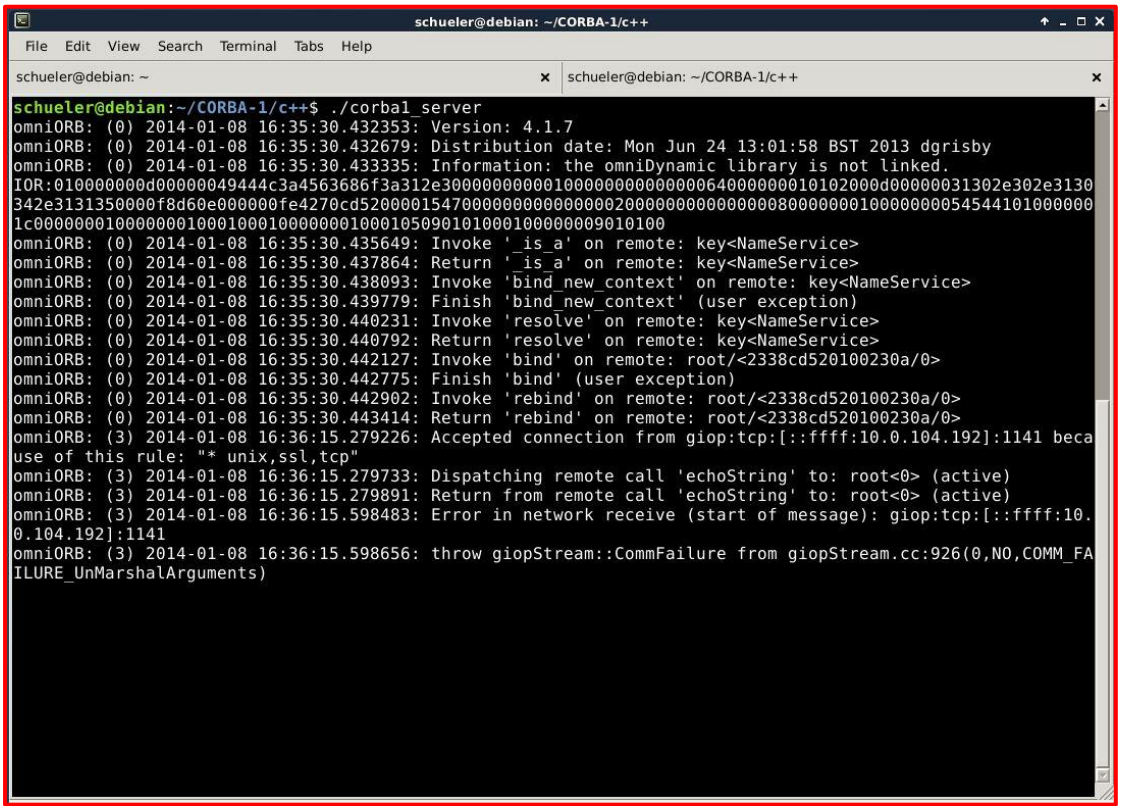
Zuerst den NameService starten, damit sich Client/Server gegenseitig finden. Die Abfolge (NameServer, Server, Client) ist wichtig, damit sich die Programme gegenseitig finden.

```
omniNames
```



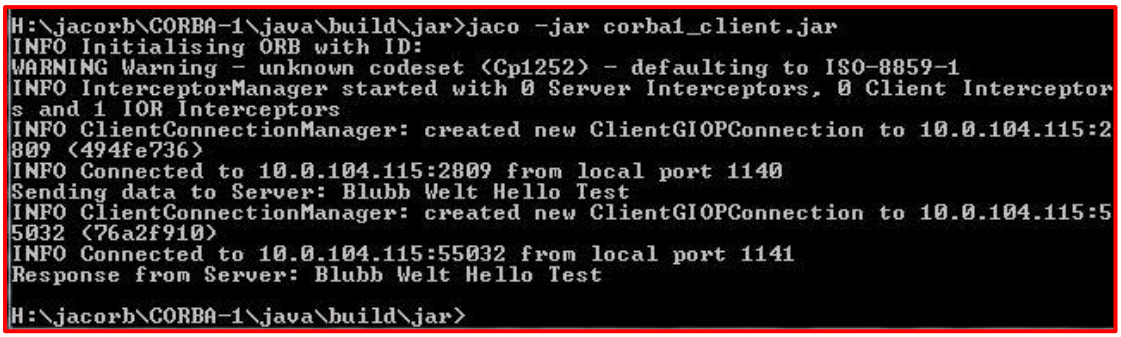
## Server

```
cd ../c++/
./corba1_server
```




## Client

```
cd ../java/
jaco -jar build/jar/corba1_client.jar
```




Wir mussten den Client auf Java ausführen, da die Linux Instanz auf Bobek's PC immer in einem anderen Subnet war und deswegen keinen Zugriff auf den Server hatte.

## Arbeitsaufteilung

| Name | Tätigkeit | Zeitaufwand in min |
|---|---|---|
| **Willinger** | Pfadanpassung | 5 |
| **Willinger** | Cleanup | 15 |
| **Willinger** | Erstellung des Makefiles | 45 |
| **Willinger** | Optimierung des Makefiles | 20 |
| **Bobek** | Java Implementation | 60 |
| **Willinger** | C++ Implementation | 60 |
| **Willinger** | omniORB | 120 |
| **Bobek** | jacORB | 30 |
| **Willinger** | Kleine Anpassungen | 10 |
| **Willinger** | Debugging | 60 |
| **Bobek** | Testen | 45 |
| **Bobek** | Dokumentation | 100 |

| Gesamtzeitaufwand in min | |
|---|---|
| **Willinger Andreas** | 335 |
| **Bobek Christian** | 235 |

## Endzeitaufteilung

| Name | Tätigkeit | Zeitaufwand in min |
|---|---|---|
| **Willinger** | Pfadanpassung | 5 |
| **Willinger** | Cleanup | 10 |
| **Willinger** | Erstellung des Makefiles | 50 |
| **Willinger** | Optimierung des Makefiles | 15 |
| **Bobek** | Java Implementation | 35 |
| **Willinger** | C++ Implementation | 45 |
| **Willinger** | omniORB | 120 |
| **Bobek** | jacORB | 30 |
| **Willinger** | Kleine Anpassungen | 10 |
| **Willinger** | Debugging | 50 |
| **Bobek** | Testen | 45 |
| **Bobek** | Dokumentation | 110 |

| Gesamtzeitaufwand in min | |
|---|---|
| **Willinger Andreas** | 305 |
| **Bobek Christian** | 225 |

## Quellen

http://www.omniorb-support.com
http://www.jacorb.org/
http://omniorb.sourceforge.net/omni41/omniORB.pdf