

# Corba-Übung

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

## Inhaltsverzeichnis

<b>Aufgabenstellung.....</b>	<b>2</b>
<b>Struktur .....</b>	<b>3</b>
Testaufbau .....	3
Verzeichnisse .....	3
<b>Installation.....</b>	<b>4</b>
omniORB.....	4
JacORB .....	6
<b>Kompilierung .....</b>	<b>7</b>
omniOrb .....	7
JacORB .....	9
<b>Testdurchführung .....</b>	<b>10</b>
Programme kompilieren .....	10
Ausführen .....	10
<b>Arbeitsaufteilung.....</b>	<b>12</b>
<b>Endzeitaufteilung .....</b>	<b>12</b>
<b>Quellen .....</b>	<b>13</b>

## 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 Namensservice 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
---- corbal_server.cc
-- idl (IDL Files)
---- echo.idl
-- java (Java Client)
---- build.xml (für Ant)
---- src
----- corbal_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/
mkdir 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 corbal_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__GNUG__ -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__GNUG__ -
D__OMNIORB4__ -D__REENTRANT__ -I. -I./ -I$(INCLUDE_DIR) -Istub/ -
D__OSVERSION__=2 -D__linux__ -D__x86_64__ corbal_server.cc
    g++ -c -O2 -Wall -Wno-unused -fexceptions -
D__OMNIORB4__ -D__REENTRANT__ -I. -I./ -I$(INCLUDE_DIR) -Istub/ -
D__OSVERSION__=2 -D__linux__ -D__x86_64__ -o corbal_server.o
corbal_server.cc

build:
    rm -f corbal_server
    g++ -o corbal_server -O2 -Wall -Wno-unused -
fexceptions -L$(LIB_DIR) corbal_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

```

schueler@debian: ~
File Edit View Search Terminal Help

schueler@debian:~$ omniNames
omniORB: (0) 2014-01-08 16:34:08.161087: Version: 4.1.7
omniORB: (0) 2014-01-08 16:34:08.161365: Distribution date: Mon Jun 24 13:01:58 BST 2013 dgrisby
omniORB: (0) 2014-01-08 16:34:08.162039: Information: the omniDynamic library is not linked.
omniORB: (0) 2014-01-08 16:34:08.164030: Dispatching local call 'rebind_context' to key<NameService> (acti
e)
omniORB: (0) 2014-01-08 16:34:08.164423: Return from local call 'rebind_context' to key<NameService> (acti
e)
omniORB: (0) 2014-01-08 16:34:08.164679: Dispatching local call 'rebind_context' to key<NameService> (acti
e)
omniORB: (0) 2014-01-08 16:34:08.164864: Return from local call 'rebind_context' to key<NameService> (acti
e)
omniORB: (0) 2014-01-08 16:34:08.165182: Dispatching local call 'rebind' to root/<153eb45201002cd0/0> (act
ve)
omniORB: (0) 2014-01-08 16:34:08.165366: Return from local call 'rebind' to root/<153eb45201002cd0/0> (act
ve)
omniORB: (0) 2014-01-08 16:34:08.165810: Dispatching local call 'rebind' to root/<2338cd520100230a/0> (act
ve)
omniORB: (0) 2014-01-08 16:34:08.166034: Return from local call 'rebind' to root/<2338cd520100230a/0> (act
ve)

Wed Jan 8 16:34:08 2014:

Read log file successfully.
Root context is IOR:010000002b00000049444c3a6f6d672e6f72672f436f734e616d696e672f4e616d696e67436f6e74657874
578743a312e300000010000000000000070000000010102000d00000031302e302e3130342e3131350000f90a0b00000004e616d655
657276696365000300000000000000080000000100000000545441010000001c000000010000000100010001000000010001050901
1000100000000901010003545441080000007933b45201002657
Checkpointing Phase 1: Prepare.
Checkpointing Phase 2: Commit.
Checkpointing completed.

```

## Server

```
cd ../c++/
./corbal_server
```

```
schueler@debian: ~/CORBA-1/c++
schueler@debian: ~
schueler@debian:~/CORBA-1/c++$ ./corbal_server
omniORB: (0) 2014-01-08 16:35:30.432353: Version: 4.1.7
omniORB: (0) 2014-01-08 16:35:30.432679: Distribution date: Mon Jun 24 13:01:58 BST 2013 dgrisby
omniORB: (0) 2014-01-08 16:35:30.433335: Information: the omniDynamic library is not linked.
IOR:010000000d000000049444c3a4563686f3a312e300000000001000000000000064000000010102000d000000031302e302e3130
342e3131350000f8d60e000000fe4270cd5200001547000000000000002000000000000008000000010000000054544101000000
1c0000000100000001000100000001000105090101000100000009010100
omniORB: (0) 2014-01-08 16:35:30.435649: Invoke 'is_a' on remote: key<NameService>
omniORB: (0) 2014-01-08 16:35:30.437864: Return 'is_a' on remote: key<NameService>
omniORB: (0) 2014-01-08 16:35:30.438093: Invoke 'bind_new_context' on remote: key<NameService>
omniORB: (0) 2014-01-08 16:35:30.439779: Finish 'bind_new_context' (user exception)
omniORB: (0) 2014-01-08 16:35:30.440231: Invoke 'resolve' on remote: key<NameService>
omniORB: (0) 2014-01-08 16:35:30.440792: Return 'resolve' on remote: key<NameService>
omniORB: (0) 2014-01-08 16:35:30.442127: Invoke 'bind' on remote: root/<2338cd520100230a/0>
omniORB: (0) 2014-01-08 16:35:30.442775: Finish 'bind' (user exception)
omniORB: (0) 2014-01-08 16:35:30.442902: Invoke 'rebind' on remote: root/<2338cd520100230a/0>
omniORB: (0) 2014-01-08 16:35:30.443414: Return 'rebind' on remote: root/<2338cd520100230a/0>
omniORB: (3) 2014-01-08 16:36:15.279226: Accepted connection from giop:tcp[:ffff:10.0.104.192]:1141 beca
use of this rule: "* unix,ssl,tcp"
omniORB: (3) 2014-01-08 16:36:15.279733: Dispatching remote call 'echoString' to: root<0> (active)
omniORB: (3) 2014-01-08 16:36:15.279891: Return from remote call 'echoString' to: root<0> (active)
omniORB: (3) 2014-01-08 16:36:15.598483: Error in network receive (start of message): giop:tcp[:ffff:10.
0.104.192]:1141
omniORB: (3) 2014-01-08 16:36:15.598656: throw giopStream::CommFailure from giopStream.cc:926(0,N0,COMM_FA
ILURE_UnMarshalArguments)
```

## Client

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

```
H:\jacob\CORBA-1\java\build\jar>jaco -jar corbal_client.jar
INFO Initialising ORB with ID:
WARNING Warning - unknown codeset <Cp1252> - defaulting to ISO-8859-1
INFO InterceptorManager started with 0 Server Interceptors, 0 Client Interceptor
s and 1 IOR Interceptors
INFO ClientConnectionManager: created new ClientGIOPConnection to 10.0.104.115:2
809 <494fe736>
INFO Connected to 10.0.104.115:2809 from local port 1140
Sending data to Server: Blubb Welt Hello Test
INFO ClientConnectionManager: created new ClientGIOPConnection to 10.0.104.115:5
5032 <76a2f910>
INFO Connected to 10.0.104.115:55032 from local port 1141
Response from Server: Blubb Welt Hello Test
H:\jacob\CORBA-1\java\build\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>