CS4/MSc Distributed Systems
Tutorial 3: Java and CORBA

Stephen Gilmore
stg@dcs.ed.ac.uk

Tutorial: M 12, Room 5325
Office hour: Th 2, Room 1608

Yussuf Abu-Shaaban
yna@dcs.ed.ac.uk

Tutorial: M 12, Room 6207
Office hour: M 4, Room 3311

A CORBA IDL example

This example is taken from *Client/Server Programming with Java and CORBA* (Wiley 1997)

```
// Count.idl

module Counter
{
  interface Count
  { attribute long sum;
    long increment();
  };
};
```


Compiling the example

- This example is compiled with `idlj -fall count.idl`
- The following files are generated in `./Counter/`:
  - `Count.java` (an interface)
  - `CountHelper.java` (an abstract class with `insert()`, `extract()`, `type()`, `id()`, `read()`, `write()`, `narrow()`)
  - `CountHolder.java` (a final class with `value`, `type()`, `read()`, `write()`)
  - `CountOperations.java` (an interface)
  - `_CountImplBase.java` (an abstract class)
  - `_CountStub.java` (a class)

How attributes are represented in Java

- IDL provides the concept of an *attribute*, which does not exist in Java.
- An attribute is represented in Java using a field of an object (to store data) and a pair of methods (one to get the current value of the attribute and the other to set it).
- The IDL to Java mapping give these two methods the same name as the attribute itself:
  - easier than generating a new name;
  - Java supports *method overloading*.

Generated Java interface

```
package Counter;

/**
* Counter/CountOperations.java
* Generated by the IDL-to-Java compiler (portable),
* version "3.0" from count.idl
* Monday, February 4, 2002 10:06:33 AM GMT
*/

public interface CountOperations  {
  int sum ();
  void sum (int newSum);
  int increment ();
} // interface CountOperations
```


How overloading works

- Two Java methods with the same name can be distinguished if they have different numbers or types of parameters.
- This method gets the value of the counter

  ```
  int sum();
  ```
- This method sets the value of the counter

  ```
  void sum(int newSum);
  ```
- Overloaded methods cannot be distinguished by their return type alone.

CORBA errors

- CORBA communication relies on the client and server having the same name for services.
- Names are represented as strings so errors which we might have expected to be reported as compile-time type errors cannot be reported in this way.
- Instead these errors are reported as CORBA exceptions at run-time.
- Here are some example errors.

An example error

Suppose that we have the following code in the server:

```
org.omg.CORBA.Object objRef =
    orb.resolve_initial_references("NameService");
```

and the following code in the client:

```
org.omg.CORBA.Object objRef =
    orb.resolve_initial_references("nameservice");
// Deliberate error!!  It should be NameService
```

When we run this code CORBA reports an exception
`org.omg.CORBA.ORBPackage.InvalidName`

Another example error

Suppose that we have the following code in the server:

```
// Bind the object reference in naming
NameComponent nc = new NameComponent("AcronymServer", "");
```

and the following code in the client:

```
NameComponent nc = new NameComponent("AcronymServer", " ");
// Deliberate error!   We have a inserted a space here ^
```

When we run this code CORBA reports an exception
`org.omg.CosNaming.NamingContextPackage.NotFound`

Errors in set up

- To run CORBA we start the Java IDL name server on our machine choosing an port (for example 1096, chosen at random)

  `tnameserv -ORBInitialPort 1096 &`

- We sould then run our server

  `java AcronymServer -ORBInitialPort 1096 &`

- Suppose that we mistakenly give the *wrong* port number to the client . . .

  `java AcronymClient -ORBInitialPort 1906`

CORBA exception raised

- For this and other similar “plumbing” errors, CORBA raises an `org.omg.CORBA.COMM_FAILURE`
- Here the diagnostic error message will refer us back to the call to `resolve_initial_references()`
- The call to `ORB.init()` simply sets the parameters as passed in from the command line. It is not until we attempt to use these parameters that the error is detected.

Host-related errors

- If we mistakenly give the wrong machine name to a CORBA invokation:

  ```
  java AcronymClient -ORBInitialPort 1096
    -ORBInitialHost scra.dcs.ed.ac.uk
  ```

  (where we had meant to type `scar.dcs.ed.ac.uk`)

- We again get an `org.omg.CORBA.COMM_FAILURE` exception.

Java resources

- The Java Developer Connection on-line tutorials
  `developer.java.sun.com/developer/onlineTraining/`
- The Java Developer Connection Introduction to CORBA
  `developer.java.sun.com/developer/onlineTraining/corba/`
- CORBA technology and the Java platform
  `http://java.sun.com/j2ee/corba/`