# CORBA examples

Source: Orfali et al

# IDL file

```
// Count.idl

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

# IDL file

```
// Count.idl

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

# Server-side Java

- Count *interface*
- _CountImplBase *skeleton*
- CountImpl *class implementation (you write)*
- CountServer *main program (you write)*

## Count *interface*

```
package Counter;
public interface Count extends org.omg.CORBA.Object
 {
     public void sum(int sum);
     public int sum();
     public int increment();
}
```

## CountImpl: *implementation*

```
class CountImpl extends Counter._CountImplBase
{
  private int sum;

  // Constructors
  CountImpl(String name)
  { super(name);
    System.out.println("Count Object Created");
    sum = 0;
  }
  // get sum
  public  int sum()
  { return sum;
  }
```

## CountImpl cont'd

```
// set sum
  public  void sum(int val)
  { sum = val;
  }

  // increment method
  public int increment()
  { sum++;
    return sum;
  }
}
```

## CountServer: *main server program*

```
class CountServer
{ static public void main(String[] args)
  { try
    { // Initialize the ORB
   org.omg.CORBA.ORB orb = org.omg.CORBA.ORB.init(args,
                                                  null);

      // Initialize the BOA
      org.omg.CORBA.BOA boa = orb.BOA_init();

      // Create the Count object
      CountImpl count = new CountImpl("My Count");
```

## CountServer cont'd

```
        // Export to the ORB the newly created object
        boa.obj_is_ready(count);

        // Ready to service requests
        boa.impl_is_ready();
        }
        catch(org.omg.CORBA.SystemException e)
        { System.err.println(e);
        }
    }
}
```

## Client-side Java

- _st_Count: *client-side stub*
- CountHelper*: helper functions, eg. Bind, narrow*
- CountHolder: *support for out and inout parameters*
- CountClient: *client program (you write)*

# CountClient

```
class CountClient
{ public static void main(String args[])
  { try
    { // Initialize the ORB
      System.out.println("Initializing the ORB");
      org.omg.CORBA.ORB orb = org.omg.CORBA.ORB.init(args,
                                                     null);
      // Bind to the Count Object
      System.out.println("Binding to Count Object");
      Counter.Count counter = Counter.CountHelper.bind(orb, "My
                                                     Count");
      // Set sum to initial value of 0
      System.out.println("Setting sum to 0");
      counter.sum((int)0);
```

```
            // Calculate Start time
            long startTime = System.currentTimeMillis();
            // Increment 1000 times
            System.out.println("Incrementing");
            for (int i = 0 ; i < 1000 ; i++ )
            { counter.increment();
            }
            // Calculate stop time; print out statistics
            long stopTime = System.currentTimeMillis();
            System.out.println("Avg Ping = "
                        + ((stopTime - startTime)/1000f) + " msecs");
            System.out.println("Sum = " + counter.sum());
          } catch(org.omg.CORBA.SystemException e)
          { System.err.println("System Exception");
            System.err.println(e);
          }
        }
      }
```

# POA & Naming Service example

## CountPortableClient

```
import org.omg.CosNaming.*;

class CountPortableClient
{ public static void main(String args[])
  { try
    { // Initialize the ORB
      System.out.println("Initializing the ORB");
      org.omg.CORBA.ORB orb = org.omg.CORBA.ORB.
                                                  init(args, null);

      // Get a reference to the Naming service
      org.omg.CORBA.Object nameServiceObj =
              orb.resolve_initial_references ("NameService");
```

## CountPortableClient cont'd

```
if (nameServiceObj == null)
    {
      System.out.println("nameServiceObj = null");
      return;
    }
    org.omg.CosNaming.NamingContext nameService =
            org.omg.CosNaming.NamingContextHelper.
                                    narrow (nameServiceObj);
    if (nameService == null) {
      System.out.println("nameService = null");
      return;
    }
```

## CountPortableClient cont'd

```
// resolve the Count object in the Naming service
    NameComponent[] countName = {new
                            NameComponent("countName", "")};
    CounterPortable.Count counter =
      CounterPortable.CountHelper.
                    narrow(nameService.resolve(countName));

    // Set sum to initial value of 0
    System.out.println("Setting sum to 0");
    counter.sum((int)0);
    System.out.println("Incrementing");
    for (int i = 0 ; i < 1000 ; i++ )
     counter.increment();
```

## CountPortableImpl

```
// CountPortableImpl.java: The Count Implementation
class CountPortableImpl extends CounterPortable.
                                                    _CountImplBase
{
  private int sum;
  // Constructors
  CountPortableImpl()
  { super();
    System.out.println("Count Object Created");
    sum = 0;
  }
  // get sum
  public synchronized int sum()
  { return sum; }
```

## CountPortableImpl cont'd

```
  // set sum
    public synchronized void sum(int val)
    { sum = val;
    }

    // increment method
    public synchronized int increment()
    { sum++;
      return sum;
    }
  }
```

## CountPortableServer

```
// CountPortableServer.java: The Count Server main program

import org.omg.CosNaming.*;

class CountPortableServer
{ static public void main(String[] args)
  { try
    { // Initialize the ORB
      org.omg.CORBA.ORB orb = org.omg.CORBA.ORB.
                                                    init(args, null);

      // Create the Count object
      CountPortableImpl count = new CountPortableImpl();
```

## CountPortableServer cont'd

```
// Export the newly create object
      orb.connect(count);

      // Get a reference to the Naming service
      org.omg.CORBA.Object nameServiceObj =
              orb.resolve_initial_references ("NameService");
      if (nameServiceObj == null)
      {
        System.out.println("nameServiceObj = null");
        return;
      }
```

## CountPortableServer cont'd

```
org.omg.CosNaming.NamingContext nameService =
            org.omg.CosNaming.NamingContextHelper.narrow
                                                (nameServiceObj);
    if (nameService == null)
    {
      System.out.println("nameService = null");
      return;
    }

    // bind the Count object in the Naming service
    NameComponent[] countName = {new NameComponent
                                                ("countName", "")};
```

## CountPortableServer cont'd

```
    nameService.rebind(countName, count);

        // wait forever for current thread to die
        Thread.currentThread().join();

      } catch(Exception e)
      { System.err.println(e);
      }
    }
  }
```

# DII Example

# CountCliDii

```
// CountClientDii.java  Dynamic Client, VisiBroker for Java

import org.omg.CosNaming.*;

class CountClientDii
{ public static void main(String args[])
  {
    boolean loop_all = false;
    long startTime, stopTime;
    org.omg.CORBA.Request request;
```

```
try
   { // Initialize the ORB.
     System.out.println("Initializing the ORB");
     org.omg.CORBA.ORB orb = org.omg.CORBA.ORB.init(args,
                                                    null);

     // Get a reference to the Naming service
     org.omg.CORBA.Object nameServiceObj =
              orb.resolve_initial_references ("NameService");
     if (nameServiceObj == null)
     {
       System.out.println("nameServiceObj = null");
       return;
     }
```

```
org.omg.CosNaming.NamingContext nameService =
              org.omg.CosNaming.NamingContextHelper.narrow
                                          (nameServiceObj);
     if (nameService == null)
     {
       System.out.println("nameService = null");
       return;
     }

     // resolve the Count object reference
     NameComponent[] countName = {new
                              NameComponent("countName", "")};
     CounterPortable.Count counter =
       CounterPortable.CountHelper.narrow
                              (nameService.resolve(countName));
// Set Sum to initial value of 0
     System.out.println("Setting Sum to 0");
     counter.sum((int)0);
```

```
if ((args.length != 0) &&
        (args[0].compareTo("loop_all") == 0))
       loop_all = true;

     if (loop_all)
     {
       System.out.println("Starting IR lookup + invoke test");
       // Calculate Start time
       startTime = System.currentTimeMillis();

       for (int i = 0 ; i < 1000 ; i++ )
       {
         request = buildRequest(counter);
         request.invoke();
       }
     }else
```

```
{     System.out.println("Starting invoke only test");
       request = buildRequest(counter);
   // Calculate Start time
       startTime = System.currentTimeMillis();
     // Increment 1000 times
       for (int i = 0 ; i < 1000 ; i++ )
       { request.invoke();
       }
     }
     // Calculate stop time; print out statistics
     stopTime = System.currentTimeMillis();
     System.out.println("Avg Ping = "  + ((stopTime - startTime)
                                        /1000f)  + " msecs");
     System.out.println("Sum = " + counter.sum());
   }
   catch(Exception e)
   { System.err.println("System Exception");  System.err.println(e)
   } }
```

```
public static org.omg.CORBA.Request buildRequest(CounterPortable.Count
                                                                    counter)
  {
    //get interface for Count object
    org.omg.CORBA.InterfaceDef CountInterface = counter._get_interface();
 // describe interface for Count object
    org.omg.CORBA.InterfaceDefPackage.FullInterfaceDescription intDesc =
    CountInterface.describe_interface();

    if (intDesc.operations[0].name.compareTo("increment") == 0)
    { //create request object for dynamic increment
      org.omg.CORBA.Request request = counter._request("increment");
      // initialize result value
      request.result().value().insert_long(0);
      return request;
    } else
      System.out.println("Unknown method");
    return null;
  }
}
```