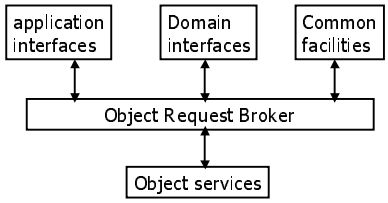
Objektiväliohjelmistoista

Objektijärjestelmä: CORBA

- Common Object Request Broker Architecture



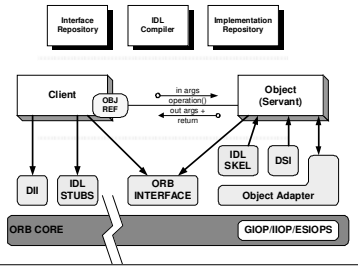
CORBA object services

- Object naming service
- Trading service
- Event service and Notification
- Persistence
- Transaction service
- Lifecycle services
- Externalisation
- Systems management services
- Relationship service

Overview of CORBA

- CORBA defines *interfaces*, not *implementations*
- It simplifies development of distributed applications by automating/encapsulating
  - Object location
  - Connection & memory mgmt.
  - Parameter (de)marshaling
  - Event & request demultiplexing
  - Error handling & fault tolerance
  - Object/server activation
  - Concurrency
  - Security

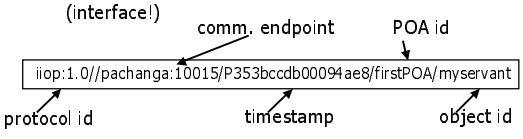
Overview of CORBA



- CORBA shields applications from heterogeneous platform *dependencies*
  - *e.g.*, languages, operating systems, networking protocols, hardware

Some CORBA terminology

- Object
  - CORBA programming entity that consists of an identity and an implementation, which is known as a servant
- Object reference
  - Immutable, opaque reference to object (instance) (interface!)



Some CORBA terminology

- Servant
  - An implementation programming language entity that defines the operations that support a CORBA IDL interface.
- Client
  - The program entity that invokes an operation on an object implementation; ideally, identical call mechanism than with object method call within the programming language

Elements of CORBA

- ORB interface
  - Abstract interface, standard for communication and helper services
  - Hides object location, implementation, execution state, communication mechanisms
  - IIOP, GIOP

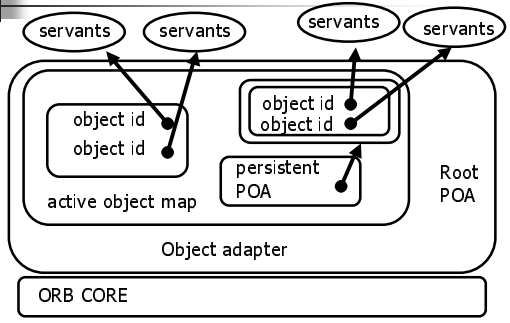
Elements of CORBA

- CORBA IDL
  - Stubs, skeletons
  - Language mappings
  - Interface inheritance
  - Interface repository
  - Dynamic invocation for gateway construction

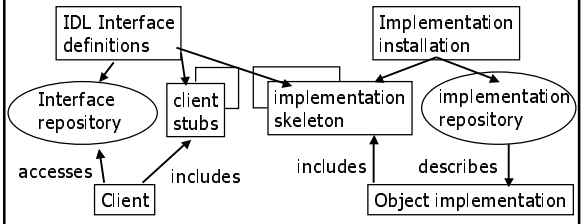
Elements of CORBA

- Object adapter (BOA-> POA)
  - Object registration
  - Object reference generation
  - Server process activation
  - Object activation
  - Request multiplexing
  - Object upcalls
  - Support for transactions, security, threading, etc.

Overview of POA architecture



Hajautettu CORBA-sovellus



Hajautettu CORBA-sovellus

- IDL

```
module HelloApp{
        interface Hello {
                string sayHello();
        };
};
```

Hajautettu CORBA-sovellus

- Asiakas

```
Import HelloApp.*;
//contains generated stubs
Import org.omg.Cosnaming.*;
Import org.omg.CORBA*;
//must be in all CORBA applications

Public class HelloClient {
    public static void main(String args[]) {
    try {
          // create and initialize the ORB
          ORB orb = ORB.init(args, null);
          // get the root naming context
          org.omg.CORBA.object objRef =
            = orb.resolve_initial_references("NameService");
```

```
NamingContext ncRef =
      NamingContextHelper.narrow(objRef);
// Resolve the object reference in naming
NameComponent nc = new NameComponent ("Hello", "");
NameComponent path[] = {nc};
// get the stub
Hello HelloRef =
      HelloHelper.narrow(ncRef.resolve(path));
// Call the Hello server object and print results
   String hello = helloRef.sayHello();
   system.out.println(hello);
} catch (Exception e) {
     System.out.println("Error : " + e);
     e.printStachTrace(System.out);
}
}
```

Hajautettu CORBA-sovellus

- Palvelupuoli

```
Import HelloApp.*;
// contains generated stubs
Import org.omg.Cosnaming.*;
Import omg.org.cORBA.*;

Public class HelloServer {
    public static void main(String args[]);
    //registration, entering server loop
    try{
        ORB orb = ORB.init(args, null);
```

```
// creae the servant and register with orb
HelloServant helloRef = new HelloServant();
orb.connect(helloRef);
//get root naming context
omg.org.CORBA.object objRef =
   orb.resolve_initial_references("NameService");
NamingContext ncRef =
   NamingContextHelper.narrow(objRef);
//bind the object reference to name
NameComponent nc = new NameComponent("Hello", "");
NameComponent path[] = {nc};
ncRef.rebind(path, helloRef);
```

```
//wait for invocations from clients
java.lang.object sync = new java.lang.Object ();
syncronized (sync)  {
   sync(wait);
}
catch (Exection e) {
   system.err.println("Error: " + s);
   s.printStackTrace (System.out);
}
}
```

```
//the actual service

Class HelloServant extends _HelloImplBase {
  public string sayHello () {
  return "\nHello world!\n";
}
}
```

CORBA limitations

- No standard way to deploy object implementations
- limited standard support for common CORBA server programming patterns
- limited extension of object functionality
  - Inheritance, 1-1 relationship between implementations and interfaces
- Availablity of CORBA object services is not defined in advance
- No object/system management services defined
  - No standard object life cycle management

Object Models and Components

- Issues that lead to different models
  - Communication vs portability
  - Business functionality vs management
  - Advanced functionality vs guaranteed reuse
- Object models describe invocation
- Component models describe component execution environment

Kurssilukemisto

- Vinoski, S., CORBA: Integrating Diverse Applications within Distributed Heterogeneous Environments
- OMG, A Discussion of the Object Management Architecture. Chapters 1-4. ??? saatavissa?

Muuta materiaalia

- POA-kalvoja
  - Schmidt, D., Developing Distributed Object Computing Applications with CORBA. Slide set from http://www.enq.uci.edu/~schmidt/PDF/corba4.pdf
- Ohjelmointitutoriaaleja
  - http://www.cs.wustl.edu/~schmidt/ACE_wrappers/TAO/docs/tutorials/Quoter/
  - useimmilla toimittajilla omansa (Sun+Java+CORBA, etc)
- CORBA-sivustoja
  - http://www.cs.wustl.edu/~schmidt/corba.html
  - http://www.puder.org/corba/