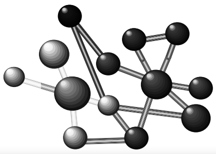
CORBA

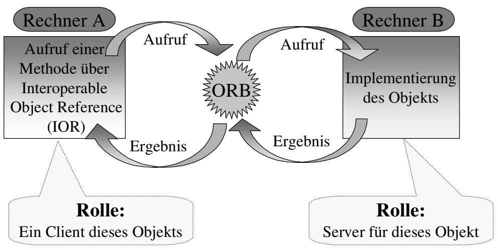


Common
Object
Request
Broker
Architecture

CORBA

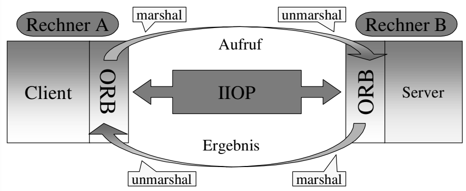
- Object Management Group (OMG)
  - über 800 Mitgliedern
- Standard zur Implementierung verteilter, objektbasierter Anwendungen
- Transparenz
  - Lokalität der Objekte
  - Implementationssprache der Objekte
- Standardisierte CORBA-Dienste
  - Naming, Transaction, Persistence

Object Request Broker (ORB)



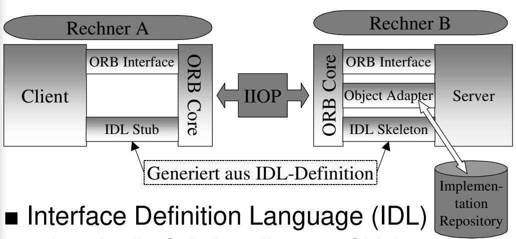
Aufrufsemantik: "at-most-once" (synchron)
"best-effort" (IDL-Attribut: oneway)

ORB: Aufgabenteilung



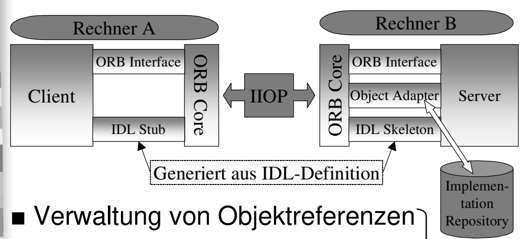
- Plattformunabhänigkeit:
  - Marshalling: Empfänger paßt Daten ggf. an
  - General Inter ORB Protocol (GIOP)
  - Internet Inter ORB Protocol (IIOP)

ORB: Statische Aufrufe



- Interface Definition Language (IDL)
  - beschreibt Schnittstellen von Objekten
- statischer Methodenaufruf
  - Interface bekannt zur Compilezeit (keine statische Methode im Java-Sinn)

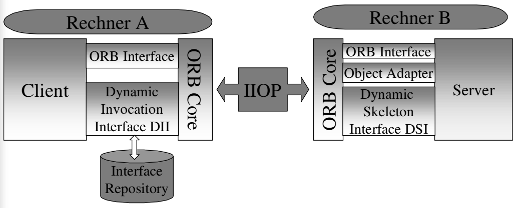
ORB: Object Adapter



- Verwaltung von Objektreferenzen
- Aktivierung des Servers
- Instantiierung neuer Objekte
- spezialisierte Dienste (proprietäre OAs)

BOA

ORB: Dynamische Aufrufe



- dynamischer Methodenaufruf
  - fordert Interfacebeschreibung des Objekts vom Interface Repository an
  - ruft dann über DII Methode auf

Interface Definition Language IDL

- Primitive Datentypen
  - `short, long, float, double, char, boolean, string, octet, enum, any`
- Strukturierte Datentypen
  - `structure, union, array` (feste Länge), `sequence, exception`
- Schnittstellen (+ Vererbung)
  - Attribute (Lese- und Schreiboperation)
  - Operationen
    - Name, Ergebnistyp, Parameterliste (`in,out, inout`), Exceptionliste und Kontextliste
- Module (Namensräume)

IDL: Ein Beispiel

```
module Beispiel{

  interface TextMsg {
    readonly attribute string msg; };

  interface EditMsg : TextMsg {
    exception notext{ string err; };
    string edit(in string newmsg)
           raises(NoTextException);
  };
};
```

IDL: Ein Beispiel

- idl2java Compiler erzeugt Stubs und Skeletons:

```
_TextMsgStub.java
TextMsg.java
TextMsgHolder.java
TextMsgHelper.java
_EditMsgStub.java
EditMsg.java
EditMsgHolder.java
EditMsgHelper.java
_TextMsgImplBase.java
_EditMsgImplBase.java
```

Zur Vereinfachung ohne Ausnahmebehandlung

Beispiel: Client (ORB-Initialisierung)

```
public class EditMsgClient
{ static EditMsg em_ior; // the server object
  public static void main(String args[])
  { ORB orb = ORB.init(args, null);
    org.omg.CORBA.Object objRef =
     orb.resolve_initial_references("NameService");
    NamingContext ncRef =
      NamingContextHelper.narrow(objRef);
    NameComponent nc =
      new NameComponent("EM_Obj", "");
    NameComponent path[] = {nc};
    em_ior =
      EditMsgHelper.narrow(ncRef.resolve(path));
    // Lese- und Sendeschleife !
  }
}
```

Beispiel: Client (Lese- und Sendeschleife)

```
public class EditMsgClient
{ public static void main(String args[])
  { // ORB-Initialisierung etc.
    DataInputStream data=
      new DataInputStream(System.in);
    String msg="start";
    while (!msg.equals("stop"))
     { try { em_ior.edit(msg); }
       catch (EditMsgPackage.NoTextException e )
        { System.out.println("Client: No Text!"); }
       msg=data.readLine();
     }
  }
}
```

Beispiel: Server (Object Implemenation)

```
class EditMsgServant extends _EditMsgImplBase
{ public String msg;
  public String msg() { return msg; }
  // if attribute msg were not readonly:
  // public void msg(String s) { msg=s }

  public String edit(String newmsg)
   { msg=newmsg;
     System.out.println(msg);
     return msg;
   }
}
```

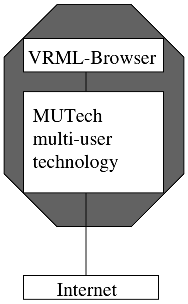
Zur Vereinfachung ohne Ausnahmebehandlung

Beispiel: Server (ORB-Initialisierung)

```
public class EditMsgServer {
 public static void main(String args[])
  { ORB orb = ORB.init(args, null);
    EditMsgServant em_ior = new EditMsgServant();
    orb.connect(em_ior);
    org.omg.CORBA.Object objRef =
     orb.resolve_initial_references("NameService");
    NamingContext ncRef =
      NamingContextHelper.narrow(objRef);
     NameComponent nc =
      new NameComponent("EM_Obj", "");
     NameComponent path[] = {nc};
     ncRef.rebind(path, em_ior);

     java.lang.Object sync = new java.lang.Object();
     synchronized (sync) { sync.wait(); }
  }
}
```

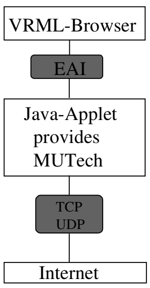
Proprietary Multi-User Worlds



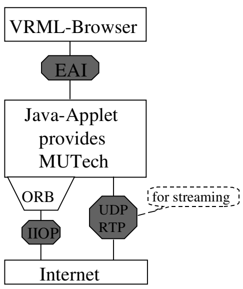
VRML und CORBA

Implementierung einer einfachen, verteilten, virtuellen Welt mit VRML, Java und CORBA

EAI-based Multi-User Worlds



CORBA-based Multi-User Worlds



Why CORBA?

- The “signature” of the protocol can be defined in IDL
- Via CORBA’s interface repositories different MUTech’s could detect common methods
- CORBA provides services, e.g. naming, for distributed systems, which we don’t have to implement from scratch.

The Protocol

- IDL Definition of the protocol

```
module MUTech
{ enum CommandType { changePosition,
                     changeOrientation, ... };
  struct Command { CommandType type;
                   long id;
                   float x, y, z, angle;
                   ...  };
  struct CommandListContainer {
          sequence< Command > commands; };
```

The Protocol

- IDL Definition of the protocol

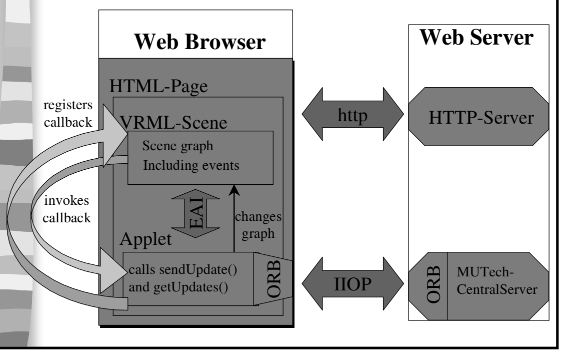
```
module MUTech
{ ...
  interface MUTechCentral {
   string getInitialWorld();
   CommandListContainer getUpdates(in long id);
   void sendUpdate(in Command c,in long id);
   ...
   };
};
```

Implementation of the Protocol

- Client
  - `centralRef.sendUpdate(change,myID);`
  - `changelist=centralRef.getUpdates(myID);`
- Server

```
– public void sendUpdate(Command c, int id)
  { for(int i=0;i<clients;i++)
      if (i!=id)
        commands.elementAt(i).addElement(c ); }
– public CommandListContainer getUpdates(int id)
  { CommandListContainer clc;
   // copies all entries in commands.elementAt(id) to clc
   return clc; }
```

Architecture



Conclusion

- Multi-User Worlds can be implemented using wide-spread web browsers and a set of (almost) open standards.
- CORBA and built-in ORBs in web browsers allow for lean implementations.
- Lean Implementations can be loaded over the internet when entering the world.