



# Architekturen für verteilte Objekte

Wintersemester 2006/2007

Übung 5: Aspectix & Co.



## 1 Wiederholung

- Aspectix ist CORBA-Compliant
  - ◆ Aspectix-IORs in reinem CORBA-ORB nutzbar
- ORB-Klasse durch eigene ersetzt
  - ◆ JVM-Properties org.omg.CORBA.ORB(Singleton)Class
  - ◆ Erweiterung von resolve\_initial\_references()
    - Liefert nun auch CreationService, ProfileManager, ...
  - ◆ Erweiterung von string\_to\_object(), object\_to\_string()
- CDRIInputStream/CDROutputStream erweitert
  - ◆ IORs erkennen/erzeugen
  - ◆ Aspectix-Profilen auswerten

5.1

© 2007, Andreas I. Schmied, Verteilte Systeme, Univ. Ulm, [2006w-AVO-US.fm, 2007-01-15 11.22] <http://www-vs.informatik.uni-ulm.de/teach/ws06/avo/>

© 2007, Andreas I. Schmied, Verteilte Systeme, Univ. Ulm, [2006w-AVO-US.fm, 2007-01-15 11.22] <http://www-vs.informatik.uni-ulm.de/teach/ws06/avo/>

## 2 Entwicklungsprozess



- Ablauf
  - 1 Schnittstelle in IDL konzipieren
  - 2 Schnittstelle in Java übersetzen mit IDLflex + APX-Mapping
  - 3 Fragment-Implementierungen, Clients erstellen
  - 4 Übersetzen des geschriebenen und generierten Codes
- Beispiele im Aspectix-Paket
  - ◆ Simple-apx-server: Verteilte Hashtable
  - ◆ Chat: Multicast, inkl. Standard-CORBA-Clients
- ▲ Die Beispiele benötigen den GNU C Compiler.  
Unter Windows muss Cygwin + GCC installiert werden.

5.2

© 2007, Andreas I. Schmied, Verteilte Systeme, Univ. Ulm, [2006w-AVO-US.fm, 2007-01-15 11.22] <http://www-vs.informatik.uni-ulm.de/teach/ws06/avo/>

## 2.1 Client-Code



- Code bezieht sich zunächst auf das Beispiel „simple-apx-server“
- Simpledb.getInfo(...)
  - ◆ Findet Daten im lokalen Fragment
  - ◆ Befragt Server-Fragmente nach Daten

```
BufferedReader br = new BufferedReader(
    new FileReader("apxtest2.ior"));
String s = br.readLine();
ORB orb = ORB.init(args, null);
org.omg.CORBA.Object o = orb.string_to_object(s);
Simpledb h = SimpledbHelper.narrow(o);
System.out.println(h.getInfo("Test1"));
```

5.3

© 2007, Andreas I. Schmied, Verteilte Systeme, Univ. Ulm, [2006w-AVO-US.fm, 2007-01-15 11.22] <http://www-vs.informatik.uni-ulm.de/teach/ws06/avo/>

## 2.2 IDL-Datei



- Wird mit IDLflex übersetzt
- Vergleich mit Ausgaben von idlj?
  - ◆ Siehe spätere Folien...

```
#include "aspectix_orb.idl"
module apxtest2 {
    exception NO_SUCH_ELEMENT {};
    interface simpledb : org::aspectix::orb::APXObject {
        string getInfo(in string key) raises (NO_SUCH_ELEMENT);
    };
};
```

5.4

© 2007, Andreas I. Schmied, Verteilte Systeme, Univ. Ulm, [2006w-AVO-US.fm, 2007-01-15 11.22] <http://www-vs.informatik.uni-ulm.de/teach/ws06/avo/>

## 2.4 Einfaches Client-Fragment (mit Cache)



- Wird später als Default-Fragment geladen

```
public class SimpledbFragment extends _SimpledbFragment {
    private Hashtable cache = new Hashtable();

    public SimpledbFragment(org.aspectix.orb.View view) {
        super(view);
    }

    public String getInfo(String key) throws NO_SUCH_ELEMENT
    {
        String s = (String) cache.get(key);
        if (s==null) {
            s = askOtherFragments(key);
            if (s==null) throw new NO_SUCH_ELEMENT();
            else cache.put(key, s);
        }
        return s;
    }
}
```

5.6

© 2007, Andreas I. Schmied, Verteilte Systeme, Univ. Ulm, [2006w-AVO-US.fm, 2007-01-15 11.22] <http://www-vs.informatik.uni-ulm.de/teach/ws06/avo/>

## 2.3 Interface übersetzen



- IDLflex aufrufen (Ausschnitt aus ANT-Skript)
  - ◆ Properties für Hilfs-ORB (JavaORB) gesetzt

```
<property name="idlflex.jvmarg" value="
-Dorg.omg.CORBA.ORBClass=JavaORB.CORBA.ORB
-Dorg.omg.CORBA.ORBSingletonClass=JavaORB.CORBA.ORBSingleton
-DAspectixBase=../../aspectix-core/" />

<property name="idlflex.arg" value="
-DPOA -DWRITEPATH=${basedir}/${srcGenDir}" />

<java fork="yes" classname="org.aspectix.IDLflex.IDLflexMain">
    <jvmarg line="${idlflex.jvmarg}" />
    <arg line="${idlflex.arg}" />
    <arg value="-mAPX/APXJavaMapping.xml" />
    <arg value="-i../../aspectix-core/idl" />
    <arg value="-I::apxtest2" />
    <arg value="idl/apx-test2.idl" />
</java>
```

5.5

© 2007, Andreas I. Schmied, Verteilte Systeme, Univ. Ulm, [2006w-AVO-US.fm, 2007-01-15 11.22] <http://www-vs.informatik.uni-ulm.de/teach/ws06/avo/>

## 2.5 Server-Fragment



- Erzeugt IOR inkl. Profildaten, legt Default-Implementierung fest

```
public class SimpledbServerFragment extends _SimpledbFragment {
    public org.omg.CORBA.Object me;
    public org.omg.CORBA.Object getObject() { return me; }
    public SimpledbServerFragment(CreationService ics,
        APXProfileManager mgr) {
        org.omg.IOP.IOR ior =
            ics.createNewIOR("IDL:apxtest/simpledb:1.0");
        org.omg.IOP.IORHolder iorh = new org.omg.IOP.IORHolder(ior);
        mgr.insertProfile(iorh, "apxtest2.SimpledbFragment",
            new String[] {"my CEP-address"} );
        this.prepareBlessedIOR(iorh);
        me = mgr.blessFragment(iorh.value, this);
        view = APXObjectHelper.narrow(me).get_view();
    }
    public void prepareBlessedIOR(org.omg.IOP.IORHolder ior) {...}
    public void confirmBlessedIOR(org.omg.IOP.IOR ior) {...}
}
```

5.7

© 2007, Andreas I. Schmied, Verteilte Systeme, Univ. Ulm, [2006w-AVO-US.fm, 2007-01-15 11.22] <http://www-vs.informatik.uni-ulm.de/teach/ws06/avo/>

## 2.6 Fragmentiertes Objekt initial erzeugen



- Erzeugt initiales Fragment und publiziert IOR

```
org.omg.CORBA.ORB orb = org.omg.CORBA.ORB.init(args, null);

CreationService ics = CreationServiceHelper.narrow(
    orb.resolve_initial_references("CreationService") );

APXProfileManager mgr = (APXProfileManager)
    ics.getProfileManager(APX_PROFILE_TAG.value);

SimpledbServerFragment frag
    = new SimpledbServerFragment(ics, mgr);

Simpledb obj = SimpledbHelper.narrow(frag.getObject());
org.omg.IOP.IOR ior = obj.get_view().getIOR();
org.omg.IOP.IORHolder iorh = new org.omg.IOP.IORHolder(ior);

PrintWriter pw = new PrintWriter(new FileWriter("apxtest2.ior"));
pw.println( ics.IORToString(iorh.value) );
pw.close();
```

5.8

© 2007, Andreas I. Schmied, Verteilte Systeme, Univ. Ulm, [2006w-AVO-US.fm, 2007-01-15 11.22] <http://www-vs.informatik.uni-ulm.de/teach/ws06/avo/>

## 3 Das APX-Mapping



- Ausschnitt aus IDL-Datei des Aspectix-Core

```
interface APXObject {
    View    get_view();
    APXObject duplicate_with_new_view();
};
```

- Ausschnitt aus Quellcode des „Fragment“ und „FragmentIfc“

```
class org.aspectix.orb.Fragment extends LocalObject {
    APXObject duplicate_with_new_view()    {...}
    Fragment getFragOfType(String id)    {...}
    void exchangeFragImpl(Fragment frag)  {...}
    View get_view()                       {...}
    void prepareBlessedIOR(IORHolder ior)  {...}
    void confirmBlessedIOR(IOR ior)       {...}
}

class org.aspectix.orb.FragmentIfc implements CORBA.Object {
    View get_view()
    void _setFragImpl(Fragment frag)
}
```

5.10

© 2007, Andreas I. Schmied, Verteilte Systeme, Univ. Ulm, [2006w-AVO-US.fm, 2007-01-15 11.22] <http://www-vs.informatik.uni-ulm.de/teach/ws06/avo/>

## 2.7 Start



- Zwei Targets (Ausschnitt aus ANT-Skript)

- ◆ Properties für Aspectix-ORB gesetzt

```
<property name="tests.jvmarg" value = "
    -Dorg.omg.CORBA.ORBClass=org.aspectix.orb.ORB
    -Dorg.omg.CORBA.ORBSingletonClass=org.aspectix.orb.ORB
    -Dcustom.props=../aspectix.properties" />

<java classname="apxtest2.SimpledbCreateFragment"
    classpathref="classpath" fork="true" failonerror="true">
    <jvmarg line="${tests.jvmarg}">
</java>

<java classname="apxtest2.SimpledbClient"
    classpathref="classpath" fork="true" failonerror="true">
    <jvmarg line="${tests.jvmarg}">
</java>
```

5.9

© 2007, Andreas I. Schmied, Verteilte Systeme, Univ. Ulm, [2006w-AVO-US.fm, 2007-01-15 11.22] <http://www-vs.informatik.uni-ulm.de/teach/ws06/avo/>

## 3.1 Generierte Schnittstellen des Beispiels



- Operationen, Objektschnittstelle, Fragmentschnittstelle

```
interface SimpledbOperations
    extends org.aspectix.orb.APXObjectOperations {
    public String getInfo(String key) throws NO_SUCH_ELEMENT;
}

interface Simpledb extends SimpledbOperations,
    org.aspectix.orb.APXObject,
    org.omg.CORBA.Object
{}

interface SimpledbFragOpe
    extends org.aspectix.orb.APXObjectFragOpe {
    public String getInfo(String key)
        throws org.aspectix.orb.exc.ROLL_BACK_EXCEPTION,
            NO_SUCH_ELEMENT;
}
```

5.11

© 2007, Andreas I. Schmied, Verteilte Systeme, Univ. Ulm, [2006w-AVO-US.fm, 2007-01-15 11.22] <http://www-vs.informatik.uni-ulm.de/teach/ws06/avo/>

## 3.2 Generierter Fragment-Rumpf



- Konstruktor mit View-Parameter
  - ◆ Notwendig für „implicit binding“

```
class _SimpledbFragment extends Fragment
    implements SimpledbFragOpe {

    public _SimpledbFragment(org.aspectix.orb.View view) {
        super(view);
    }

    public static final String[] _ids_list = {
        "IDL:apxtest2/Simpledb:1.0",
        "IDL:org/aspectix/orb/APXObject:1.0"
    };
}
```

5.12

© 2007, Andreas I. Schmied, Verteilte Systeme, Univ. Ulm, [2006w-AVO-U5.fm, 2007-01-15 11.22] <http://www-vs.informatik.uni-ulm.de/teach/ws06/avo/>

## 4 Beispiel Chat-Dienst



Konzeption:

- CORBA-basiert?
  - ◆ Menge von Objekten
  - ◆ Finden der Beteiligten
  - ◆ Verschicken von Nachrichten
  - ◆ Empfangen über Rückkanal
- Aspectix-basiert?
  - ◆ Fragmente eines Dienstobjekts
  - ◆ Interne Kommunikation: effizient über UDP-Multicast
  - ◆ Integration von reinen CORBA-Clients möglich?

5.14

© 2007, Andreas I. Schmied, Verteilte Systeme, Univ. Ulm, [2006w-AVO-U5.fm, 2007-01-15 11.22] <http://www-vs.informatik.uni-ulm.de/teach/ws06/avo/>

## 3.3 Generierter Fragment-Stub



```
class _SimpledbFragmentIfc extends FragmentIfc
    implements Simpledb, CORBA.Object {

    public String getInfo(String arg) throws NO_SUCH_ELEMENT {
        return ((SimpledbFragOpe)_FragImpl).getInfo(arg);
    }

    public APXObject duplicate_with_new_view() {
        // new View and FragIfc
        View view = new org.aspectix.orb.View(_View);
        _SimpledbFragmentIfc fragIfc = new _SimpledbFragmentIfc(view);
        // initial implementation for APXObject
        Fragment fragImpl = new org.aspectix.orb.Fragment(view);
        view.setFragImpl(fragImpl);
        return (APXObject) fragIfc;
    }
}
```

5.13

© 2007, Andreas I. Schmied, Verteilte Systeme, Univ. Ulm, [2006w-AVO-U5.fm, 2007-01-15 11.22] <http://www-vs.informatik.uni-ulm.de/teach/ws06/avo/>

## 4.1 Lösungsskizze



- Kommunikation über Multicast-Sockets
  - ◆ Eigener Thread zum Empfang der Nachrichten
- Erstes Fragment explizit erzeugt
  - ◆ Erstellt und publiziert IOR
  - ◆ Schreibt Multicast-Adresse in Aspectix-Profil
- Mittels IOR erzeugte Fragmente beziehen diese aus Profil
- Reine CORBA-Clients an erstem Fragment angemeldet
  - ◆ Weiterleitung jeder Nachricht
  - ◆ Spezielle Fragment-Implementierung „ReflectiveChatFragment“

5.15

© 2007, Andreas I. Schmied, Verteilte Systeme, Univ. Ulm, [2006w-AVO-U5.fm, 2007-01-15 11.22] <http://www-vs.informatik.uni-ulm.de/teach/ws06/avo/>

## 4.2 IDL-Schnittstelle



```
#include "aspectix_orb.idl"

module org { module aspectix { module examples {
module chat {

    exception TRANSMISSION_FAILURE {};

    interface Sink {
        void push(in string msg);
    };

    interface Chat : org::aspectix::orb::APXObject {
        void send (in string msg) raises (TRANSMISSION_FAILURE);
        void join (in Sink s);
        void leave (in Sink s);
    };

};};};};
```

5.16

© 2007, Andreas I. Schmied, Verteilte Systeme, Univ. Ulm, [2006w-AVO-US.fm, 2007-01-15 11.22] <http://www-vs.informatik.uni-ulm.de/teach/ws06/avo/>

## 4.4 Standard-CORBA-Clients „cmdLoopWithSink“



```
SinkPOA _s = new SinkPOA() {
    public void push(String msg) { System.out.println(msg); }
};

Sink s = SinkHelper.narrow(_s._this_object());
c.join(s);
cmdLoop(c);
c.leave(s);
```

## 4.5 Chat-Konsole „cmdLoop“

```
BufferedReader r = new BufferedReader(
    new InputStreamReader(System.in));

String cmd=null, name="unknown";

while ((cmd=r.readLine()) != null) {
    if (cmd.equals("quit")) break;
    if (cmd.startsWith("name ")) name = cmd.substring(5);
    else c.send(name+": "+cmd);
}
```

5.18

© 2007, Andreas I. Schmied, Verteilte Systeme, Univ. Ulm, [2006w-AVO-US.fm, 2007-01-15 11.22] <http://www-vs.informatik.uni-ulm.de/teach/ws06/avo/>

## 4.3 Client-Code



### ■ Zumindest erster Client verwendet Aspectix

```
void createIOR(ORB orb, String addr, int port, String iorPath) {
    ReflectiveChatFragment frag
        = new ReflectiveChatFragment(orb, addr, port);
    printStringToFile(frag.getIORasString(), iorPath);    // s.u.
    Chat c = ChatHelper.narrow(frag.getObject());
    cmdLoop(c);
}
```

### ■ Weitere Clients können auch reinen CORBA-ORB verwenden

#### ◆ Stringvergleich mit ORB-Klassenname statt instanceof: apxorb.jar unnötig

```
void startUsingIOR(ORB orb, String ior) {
    Chat c = ChatHelper.narrow( orb.string_to_object(ior) );
    if (orb.getClass().getName().equals("org.aspectix.ORB"))
        cmdLoop(c);
    else
        cmdLoopWithSink(c);
}
```

5.17

© 2007, Andreas I. Schmied, Verteilte Systeme, Univ. Ulm, [2006w-AVO-US.fm, 2007-01-15 11.22] <http://www-vs.informatik.uni-ulm.de/teach/ws06/avo/>

## 4.6 ReflectiveChatFragment



### ■ Wird immer explizit erzeugt

```
class ReflectiveChatFragment extends NormalChatFragment {

    Set sinks = new HashSet();

    public ReflectiveChatFragment(orb, String addr, int port) {
        super(orb, addr, port);
    }

    public void recv(String msg) {
        Iterator i=sinks.iterator();
        while(i.hasNext()) ((Sink)i.next()).push(msg);
    }

    public void join(Sink s) throws ROLL_BACK_EXCEPTION {
        sinks.add(s);
    }

    public void leave(Sink s) throws ROLL_BACK_EXCEPTION {
        sinks.remove(s);
    }
}
```

5.19

© 2007, Andreas I. Schmied, Verteilte Systeme, Univ. Ulm, [2006w-AVO-US.fm, 2007-01-15 11.22] <http://www-vs.informatik.uni-ulm.de/teach/ws06/avo/>

## 4.7 NormalChatFragment



- Zwei Konstruktoren zum impliziten und expliziten Erzeugen

```
class NormalChatFragment extends _ChatFragment {  
  
    // Fragment dynamisch von ORB erzeugt  
    public NormalChatFragment(org.aspectix.ORB view) {  
        super(view);  
        init(view.getORB(), null, 0);  
    }  
  
    // Initiales Fragment von Programmierer erzeugt  
    public NormalChatFragment(ORB orb, String addr, int port) {  
        init(orb, addr, port);  
    }  
  
    ...  
}
```

5.20

© 2007, Andreas I. Schmied, Verteilte Systeme, Univ. Ulm, [2006w-AVO-US.fm, 2007-01-15 11.22] <http://www-vs.informatik.uni-ulm.de/teach/ws06/avo/>

## 4.9 init() für explizites Erzeugen



- Erstes Fragment erzeugt und konfiguriert IOR

```
else { // new IOR  
    IOR ior = ics.createNewIOR(ChatHelper.id());  
    IORHolder iorh = new IORHolder(ior);  
  
    apxpm.insertProfile(iorh,  
        "org.aspectix.examples.chat.NormalChatFragment",  
        new String[] { _addr } );  
  
    prepareBlessedIOR(iorh); // ... confirming ...  
    ior=iorh.value;  
  
    APXProfileManagerImpl _apxpm = (APXProfileManagerImpl) apxpm;  
    org.omg.CORBA.Object me = _apxpm.blessFragment(ior, this);  
    super.view = APXObjectHelper.narrow(me).get_view();  
}  
  
addr = InetAddress.getByName(_addr);  
socket = new MulticastSocket(port);  
socket.joinGroup(addr);  
packet = createPacket();  
...
```

5.22

© 2007, Andreas I. Schmied, Verteilte Systeme, Univ. Ulm, [2006w-AVO-US.fm, 2007-01-15 11.22] <http://www-vs.informatik.uni-ulm.de/teach/ws06/avo/>

## 4.8 init() für implizites Erzeugen



- Allgemeiner Teil: ICS, Aspectix-PM beziehen

```
if (!(orb instanceof org.aspectix.ORB))  
    throw new RuntimeException("Needs Aspectix ORB");  
CreationService ics = CreationServiceHelper.narrow(  
    orb.resolve_initial_references("CreationService") );  
APXProfileManager apxpm = (APXProfileManager)  
    ics.getProfileManager(APX_PROFILE_TAG.value);
```

- Kommunikationsadresse aus existierender IOR extrahieren

```
if (_addr == null) { // existing IOR  
    IORHolder iorh = new IORHolder(getIOR());  
    APXProfile ap = APXProfileManagerImpl.getAPXProfile(iorh);  
    if (ap.peers.length!=1) throw ...;  
  
    port = Integer.parseInt(ap.peers[0]...);  
}  
  
... // naechste Folie
```

5.21

© 2007, Andreas I. Schmied, Verteilte Systeme, Univ. Ulm, [2006w-AVO-US.fm, 2007-01-15 11.22] <http://www-vs.informatik.uni-ulm.de/teach/ws06/avo/>

## 4.10 Hilfsmethoden im Fragment



- Konzeptionelles FO und IOR beziehen

```
public org.omg.CORBA.Object getObject() { return me; }  
  
public IOR getIOR() {  
    Chat chat = ChatHelper.narrow(me);  
    return chat.get_view().getIOR();  
}
```

- IOR in String umwandeln

```
public String getIORAsString() {  
    try {  
        org.omg.CORBA.ORB orb = getView().getORB();  
        IORHolder iorh = new org.omg.IOP.IORHolder(getIOR());  
        CreationService ics = CreationServiceHelper.narrow(  
            orb.resolve_initial_references("CreationService") );  
        return ics.IORToString(iorh.value);  
    }  
    catch(InvalidName e) { e.printStackTrace(); return ""; }  
}
```

5.23

© 2007, Andreas I. Schmied, Verteilte Systeme, Univ. Ulm, [2006w-AVO-US.fm, 2007-01-15 11.22] <http://www-vs.informatik.uni-ulm.de/teach/ws06/avo/>

## 5 FORMI – Fragmentierte RMI-Objekte



- Fragmentierte Objekte ohne Änderung der RMI-Runtime
  - ◆ Remote Reference Layer + Stub Layer → Fragment-Interface

```
public abstract class FormiFragmentIfc
    extends RemoteStub implements java.rmi.server.RemoteRef;
```
  - ◆ FragmentFactory
    - IOR-ähnliche Aufgaben (OID, Kommunikationsadr., Typen, Init-Parameter)
    - FO-instanzspezifisch: kann Einfluss nehmen auf Default-Impl.
- Eigener RMI-Compiler für Formi-Stubs
  - ◆ Konfiguration in Datei „src/sun/rmi/rmic/resources/rmic.properties“

```
generator.class.default=org.aspectix.formi.compiler.FormiGenerator
```
- Beispiele
  - ◆ README.txt lesen, ant examples ausführen, Server/Client starten
  - ◆ Smartcache: fragmentierte java.util.Hashtable<String, String>
  - ◆ Audiosample: fragmentierte Multicast-Server/Client, javax.sound.\*\*

5.24

© 2007, Andreas I. Schmied, Verteilte Systeme, Univ. Ulm, [2006w-AVO-US.fm, 2007-01-15 11.22] <http://www-vs.informatik.uni-ulm.de/teach/ws06/avo/>

### 6.1 Implementierung



- Aufbau: Dynamic Proxy → Flake → Implementierung

```
class flakes.Flake implements InvocationHandler {
    transient Object impl;
    transient flakes.Reference ref;
    transient flakes.util.Proxy proxy;
    Object getObject() { return proxy; } // die Flake-Referenz!
    ...
}
```
- Mechanismen
  - 1 **Statische Factory-Methoden:** flakes.Flake.create(...) Serialisierbare flakes.Reference { OID, Typen, Codebase, Impl-Name, Daten }
  - 2 Serialisierung: Proxy/Flakes.writeReplace() schreibt Reference  
Deserialisierung: Reference.readReplace() erzeugt oder findet Proxy+Flake
  - 3 Dynamic Proxy als Indirektionsstufe vor Implementierung  
**Generischer Dispatcher in flakes.Flake**
  - 4 **Inter-Fragment-Kommunikation beliebig**, z.B. mit Sockets, RMI, P2P-System

5.26

© 2007, Andreas I. Schmied, Verteilte Systeme, Univ. Ulm, [2006w-AVO-US.fm, 2007-01-15 11.22] <http://www-vs.informatik.uni-ulm.de/teach/ws06/avo/>

## 6 Flakes – Fragmentierte POJOs



POJO = Plain Old Java Object (hier: ohne RMI)

- Fragestellung: Was braucht man minimal für FO in Java?
  - 1 Explizite Fragment-Erzeugung und Adressierung
  - 2 Marshalling „by-Reference“ und „Implicit Binding“, Codebase-Loader
  - 3 Ersetzbarkeit der Implementierung
  - 4 Inter-Fragment-Kommunikation
- Ziele
  - ◆ Einfache Implementierung
  - ◆ Einfaches Programmiermodell
  - ◆ Unabhängigkeit von Kommunikationsmechanismen
  - ◆ Beschränkung auf Java-Plattform
  - ◆ Nur Standard-Java-Serialisierung

5.25

© 2007, Andreas I. Schmied, Verteilte Systeme, Univ. Ulm, [2006w-AVO-US.fm, 2007-01-15 11.22] <http://www-vs.informatik.uni-ulm.de/teach/ws06/avo/>

### 6.2 Diskussion



- RMI-Naming nicht benutzbar
  - ◆ Verlangt RemoteObject, Flakes sind aber POJOs
  - ◆ Alternative: flakes.map.Mapper fragmentiert java.util.Map<String, Object>
- Dynamic Proxy musste erweitert werden
  - ◆ Reference statt Proxy-Instanz und Handler serialisieren
- Benutzung für Anwendungsentwickler nicht transparent
  - ◆ Referenz „this“ darf nicht nach außen gegeben werden, sondern nur die Flake-Referenz (der Proxy)
    - Problem besteht auch in anderen Systemen mit Indirektion, z.B. EJB
    - Kann durch statische Code-Analyse+Transformation/AOP entschärft werden
  - ◆ Interaktion von Flake und Implementierung notwendig
    - Interna weitergeben (Reference-, Flake-Objekt, ...)
    - Implementierungswechsel anstoßen
- Dynamic Classloading noch nicht integriert...

5.27

© 2007, Andreas I. Schmied, Verteilte Systeme, Univ. Ulm, [2006w-AVO-US.fm, 2007-01-15 11.22] <http://www-vs.informatik.uni-ulm.de/teach/ws06/avo/>

## 7 Aufgabe



AudioCD-System in Flakes (, Formi, Aspectix) nachbauen:

- Verteilter Lookup-Dienst
  - ◆ Lookup-Dienst im lokalen Netz per Multicast erreichbar
  - ◆ Lookup-Fragmente durch TCP-Verbindungen vernetzt
- ▲ P2P-ähnliche Konzepte
- AudioCD-Fragmente
  - ◆ Reines Stub-Fragment
    - Holt Daten von Master-Fragment
  - ◆ Caching-Fragment
    - Holt Daten einmalig von Master-Fragment, danach lokaler Zugriff
  - ◆ Kommunikation intern mittels RMI
    - RemoteExceptions werden intern abgehandelt und ggf. als AudioCDException nach außen weitergegeben

5.28