



FORMI

1 Idee

- Fragmentiertes Objektmodell mit Java RMI
 - ◆ Design und Implementierung entstanden im Rahmen einer Masterarbeit
- Anforderungen
 - ◆ „friedliche“ Koexistenz von RMI- und fragmentierten FORMI-Objekten
 - ◆ Zugriffstransparenz für Aufrufer
 - kein Unterschied zwischen RMI- und FORMI-Objekt
 - ◆ Modelltransparenz bei Parameterübergabe
 - kein Unterschied bei Parameterübergabe
 - Übergabe von FORMI-Objekten an RMI-Objekte
 - Übergabe von RMI-Objekten an FORMI-Objekte
 - Übergabe von FORMI-Objekten an FORMI-Objekte

I.1

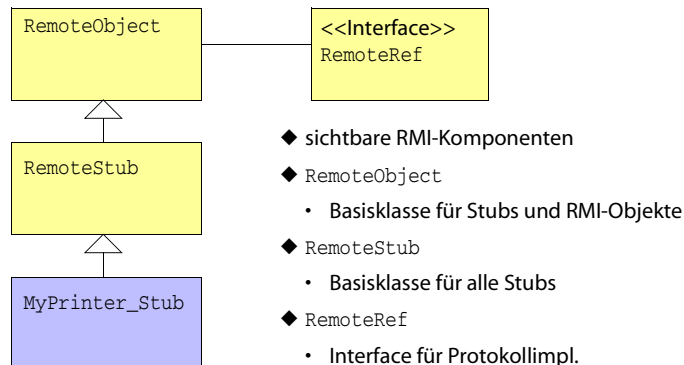
© 2002-2008, Franz J. Hauck, Verteilte Systeme, Univ. Ulm, [2007w-AvO-I-FORMI.fm, 2008-01-18 15.36] <http://www-vs.informatik.uni-ulm.de/teach/ws07/avo/>

I.2

© 2002-2008, Franz J. Hauck, Verteilte Systeme, Univ. Ulm, [2007w-AvO-I-FORMI.fm, 2008-01-18 15.36] <http://www-vs.informatik.uni-ulm.de/teach/ws07/avo/>

2 Design

- Transparente Parameterübergabe
 - ◆ FORMI-Fragmente müssen sich wie RMI-Stubs verhalten
 - müssen mit RMI-Mechanismen serialisierbar sein
- Zur Erinnerung: Aufbau der RMI-Stubs (vgl. Kap. C)



I.3

© 2002-2008, Franz J. Hauck, Verteilte Systeme, Univ. Ulm, [2007w-AvO-I-FORMI.fm, 2008-01-18 15.36] <http://www-vs.informatik.uni-ulm.de/teach/ws07/avo/>

2 Design (2)

- Zur Erinnerung: Abläufe in RMI
 - ◆ Aufruf
 - Stubobjekt wandelt Aufruf in generischen Aufruf um
 - Stubobjekt ruft invoke() -Methode an der RemoteRef auf
 - ◆ Marshalling
 - falls RMI-Objektreferenz vom Typ RemoteStub, dann serialisiere Referenz
 - falls RMI-Objektreferenz exportiertes Objekt, erzeuge Stub und verfare wie oben
 - ◆ Unmarshalling
 - deserialisiere Stubobjekt

I.4

© 2002-2008, Franz J. Hauck, Verteilte Systeme, Univ. Ulm, [2007w-AvO-I-FORMI.fm, 2008-01-18 15.36] <http://www-vs.informatik.uni-ulm.de/teach/ws07/avo/>



2.1 Aufbau eines Fragments



- Anforderung an den Fragmentaufbau
 - ◆ dynamischer Austausch der Implementierung
 - Indirektionsstufe (vgl. Aspectix in Kap. H)
 - Trennung von Fragmentimplementierung und Fragment-Interface
 - ◆ dynamische Entscheidung über Fragmentimplementierung bei Parameterübergabe
 - Einsatz einer Fragment-Implementation-Factory

I.5

© 2002-2008, Franz J. Hauck, Verteilte Systeme, Univ. Ulm, [2007w-AvO-I-FORMI.fm, 2008-01-18 15.36] <http://www-vs.informatik.uni-ulm.de/teach/ws07/avo/>

2.1 Aufbau eines Fragments (3)



- Lösung in FORMI (fortges.)
 - ◆ Stub bildet Fragment-Interface
 - ◆ Fragment-Interface implementiert `RemoteRef` und referenziert sich selbst
 - vermeidet Marshalling-Probleme
 - `invoke()`-Methode wird normalerweise nicht genutzt
 - `invoke()`-Aufrufe werden in lokale Aufrufe umgewandelt
 - ◆ Fragment-Interface leitet Aufrufe an Fragmentimplementierung weiter
 - ◆ Fragment-Interface referenziert eine Factory für Fragmentimplementierungen
 - Factory fällt Entscheidung welche Implementierung verwendet wird
 - dynamische Entscheidungen möglich (vgl. PDS in Kap. H)
 - ◆ View-Objekt sorgt für Sharing der Fragmentimplementierung
 - View-Objekt wird in einem View-Manager verwaltet (enthält Tabelle aller lokalen Fragmente)

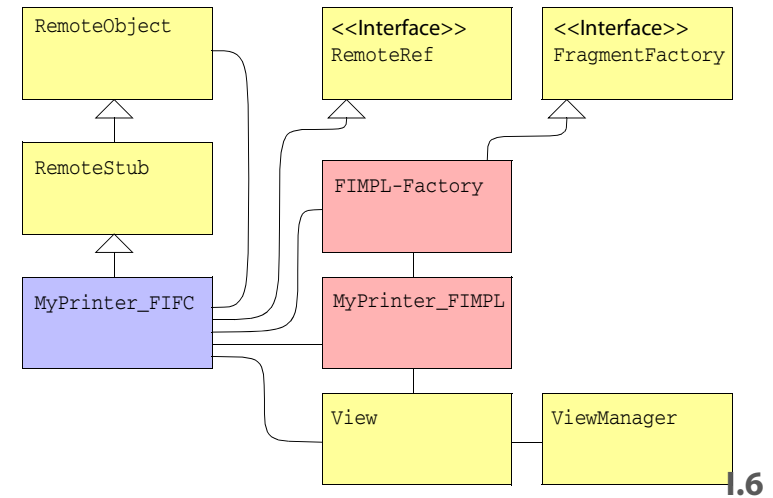
I.7

© 2002-2008, Franz J. Hauck, Verteilte Systeme, Univ. Ulm, [2007w-AvO-I-FORMI.fm, 2008-01-18 15.36] <http://www-vs.informatik.uni-ulm.de/teach/ws07/avo/>

2.1 Aufbau eines Fragments (2)



- Lösung in FORMI



I.6

© 2002-2008, Franz J. Hauck, Verteilte Systeme, Univ. Ulm, [2007w-AvO-I-FORMI.fm, 2008-01-18 15.36] <http://www-vs.informatik.uni-ulm.de/teach/ws07/avo/>

2.2 Aufrufe



- Weiterleitung aus Fragment-Interface in die Fragmentimplementierung

2.3 Marshalling und Unmarshalling

- Marshalling
 - ◆ Serialisierung des Fragment-Interfaces
 - View und Fragmentimplementierung sind als `transient` gekennzeichnet und werden nicht serialisiert
- Unmarshalling
 - ◆ Deserialisierung des Fragment-Interfaces
 - ◆ Suche nach vorhandenem View
 - Verlinken des Views und entsprechender Fragmentimplementierung oder
 - Erzeugen einer neuen Fragmentimplementierung über Factory und Erzeugung eines neuen View-Objekts über View-Manager

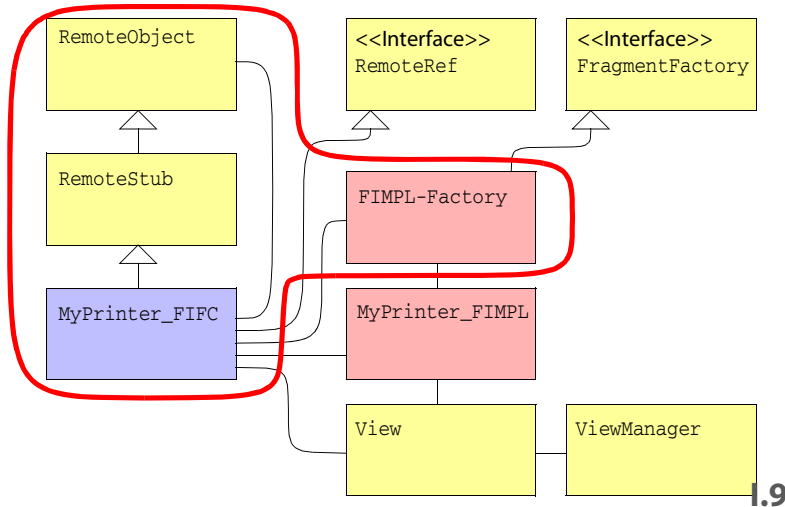
I.8

© 2002-2008, Franz J. Hauck, Verteilte Systeme, Univ. Ulm, [2007w-AvO-I-FORMI.fm, 2008-01-18 15.36] <http://www-vs.informatik.uni-ulm.de/teach/ws07/avo/>

2.3 Marshalling und Unmarshalling (2)



■ Serialisierungsbereich



I.9

© 2002-2008, Franz J. Hauck, Verteilte Systeme, Univ. Ulm, [2007w-AvO-I-FORMI.fm, 2008-01-18 15.36] <http://www-vs.informatik.uni-ulm.de/teach/ws07/avo/>

2.5 Kommunikation zwischen Fragmenten



■ Beliebige Kommunikation

- ◆ Java-Sockets stehen zur Verfügung
- ◆ Einsatz von Standard-RMI für aufrufbasierte Kommunikation zwischen Fragmenten

■ Finden der Fragmente

- ◆ zur Zeit kein Ortsdienst verfügbar
 - kann selbst implementiert werden
 - Verankerung in der Fragment-Implementation-Factory
- ◆ statische Kommunikationsadressen
 - Implantieren der Adressen in die Fragment-Implementation-Factory
- ◆ Verwendung der Registry als Ersatz für Ortsdienst
 - z.B. zum Finden von Standard-RMI-Objekten, die anderes Fragment repräsentieren

I.11

© 2002-2008, Franz J. Hauck, Verteilte Systeme, Univ. Ulm, [2007w-AvO-I-FORMI.fm, 2008-01-18 15.36] <http://www-vs.informatik.uni-ulm.de/teach/ws07/avo/>

2.4 Erzeugen von FORMI-Objekten



■ Erzeugen eines ersten Fragments

- ◆ Aufruf einer speziellen Funktion mit Übergabe von
 - Fragment-Implementation-Factory
 - FORMI-Objektyp
 - erste Fragmentimplementierung
- ◆ erstes Fragment für Client als RMI-Stub sichtbar
 - bei Parameterübergabe: Marshalling wie oben beschrieben (kein Exportieren nötig)

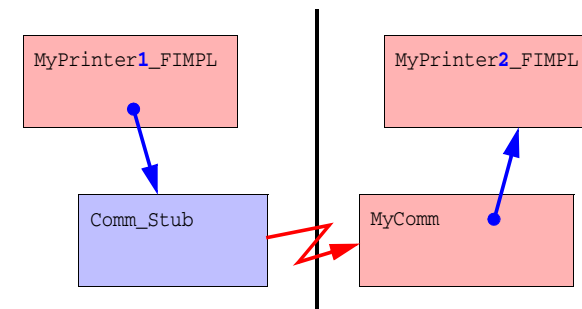
I.10

© 2002-2008, Franz J. Hauck, Verteilte Systeme, Univ. Ulm, [2007w-AvO-I-FORMI.fm, 2008-01-18 15.36] <http://www-vs.informatik.uni-ulm.de/teach/ws07/avo/>

2.5 Kommunikation zwischen Fragmenten (2)



■ Einsatz von Standard-RMI als Kommunikationsmechanismus



- ◆ Standard RMI-Objekt (im Bsp. mit Interface Comm)
- ◆ Referenzermittlung z.B. über Registry

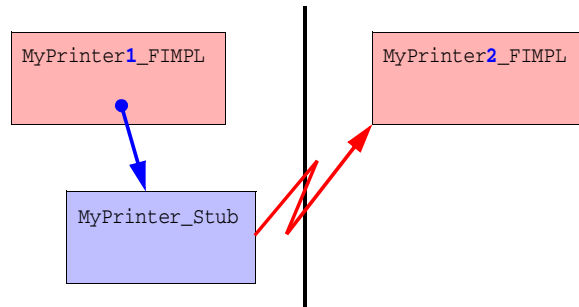
I.12

© 2002-2008, Franz J. Hauck, Verteilte Systeme, Univ. Ulm, [2007w-AvO-I-FORMI.fm, 2008-01-18 15.36] <http://www-vs.informatik.uni-ulm.de/teach/ws07/avo/>

2.5 Kommunikation zwischen Fragmenten (3)



■ Einsatz von Standard-RMI mit Optimierung



- ◆ Instanz von MyPrinter2_FIMPL ist
 - Fragmentimplementierung und gleichzeitig
 - exportiertes Standard-RMI-Objekt

I.13

© 2002-2008, Franz J. Hauck, Verteilte Systeme, Univ. Ulm, [2007w-AvO-I-FORMI.fm, 2008-01-18 15.36] <http://www-vs.informatik.uni-ulm.de/teach/ws07/avo/>

3 Vergleich mit theoretischen Anforderungen



- FORMI hat den fragmentierten Ansatz
 - ◆ weltweit eindeutige **Objektbezeichner** zum Parametertransport zwischen Fragmenten
 - serialisiertes Fragment-Interface
 - RMI-kompatibel
 - **dynamisches Laden** von Code durch RMI-Classloader und Codebase-Angabe
 - Erzeugung lokaler Fragmente mit Fragment-Interface, Fragment-Implementierung, View-Objekt
 - ◆ **Erzeugung** neuer verteilter Objekte
 - spezielle Funktion zur Erzeugung der ersten Fragmentimplementierung
 - ◆ **Namensdienst** zum Finden von Objekten
 - Verwendung der RMI-Registry

I.15

© 2002-2008, Franz J. Hauck, Verteilte Systeme, Univ. Ulm, [2007w-AvO-I-FORMI.fm, 2008-01-18 15.36] <http://www-vs.informatik.uni-ulm.de/teach/ws07/avo/>

2.6 Unterstützung bei der Objektentwicklung



■ Eigener RMI-Compiler

- ◆ **formic**-Werkzeug
 - aus rmic-Implementierung von Sun abgeleitet
 - erzeugt Fragment-Interfaces (spezielle RemoteStubs für FORMI)

■ Fragmentimplementierung

- ◆ muss lediglich Remote-Interface implementierung und ...
- ◆ ... von `FragmentImplementation` erben
- ◆ keine Code-Erzeugung notwendig

I.14

© 2002-2008, Franz J. Hauck, Verteilte Systeme, Univ. Ulm, [2007w-AvO-I-FORMI.fm, 2008-01-18 15.36] <http://www-vs.informatik.uni-ulm.de/teach/ws07/avo/>

3 Vergleich mit theoretischen Anforderungen (2)



- ◆ **Ortsdienst** zum Finden von Fragmenten
 - bisher nicht vorgesehen
- ◆ **Kommunikationsdienste** für Interfragmentkommunikation
 - Zugriff auf Java-Sockets
 - Nutzung von Standard-RMI bei aufrufbasierter Kommunikation
- ◆ **Code-Generatoren** zur Unterstützung
 - Generator `formic` für Fragment-Interfaces (aka Stubs)

I.16

© 2002-2008, Franz J. Hauck, Verteilte Systeme, Univ. Ulm, [2007w-AvO-I-FORMI.fm, 2008-01-18 15.36] <http://www-vs.informatik.uni-ulm.de/teach/ws07/avo/>