SYSTOR®

Erfahrungsbericht Server Side Java



Dr. Gernot Starke
gernot.starke@systor.com
Tel. 02234 - 108 1422

Warum „Java auf dem Server“?

SYSTOR®

- Einheitliche Entwicklungsplattform auf Client und Server
  - Homogene(re) Entwicklungsteams
- Java als Integrationsplattform
  - Viele technische Alternativen (Servlets, JSP, RMI, EJB, CORBA)
  - Flexible Middleware verfügbar
- Portabilität



„Server-side Java“ versus „3-Tier Komplexität“

„*Einfache*“ 3-Schicht Architekturen

- VORSICHT: Komplexität steigt STARK mit Anzahl Schichten!
- Mehr Schnittstellen
- Mehr potentielle Fehlerquellen
- Mehr heterogenes KnowHow notwendig



Server-Side Java: Wege zum Ziel...

SYSTOR®

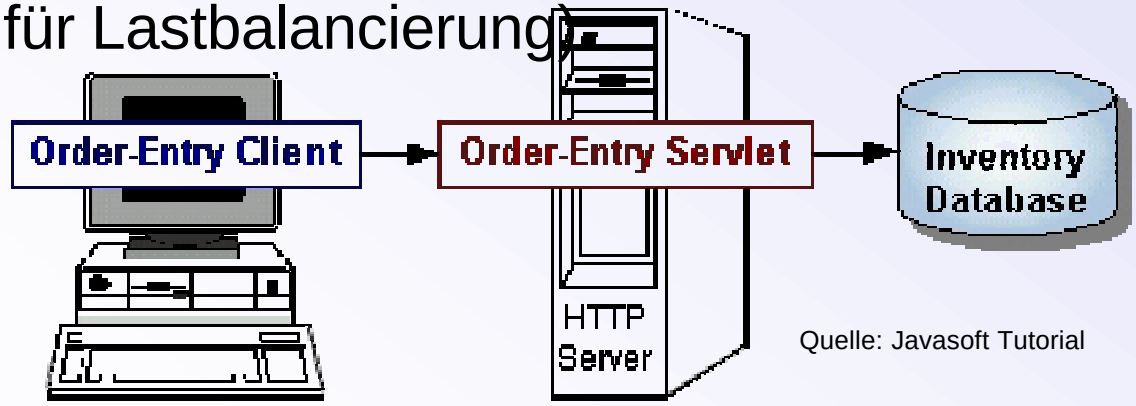
- Servlets
- Java Server Pages
- RMI
- EJB
- CORBA



Servlets

- Einsatzgebiet hauptsächlich: Ersatz von cgi-Skripten (Generierung von HTML-Seiten)
  - „Easier to write, faster to run“ (Javasoft)

- Servlets können
  - parallele Zugriffe behandeln (z.B. für Online-Konferenzen)
  - Anfragen an andere Server und Servlets weiterleiten (z.B. für Lastbalancierung)



Quelle: Javasoft Tutorial

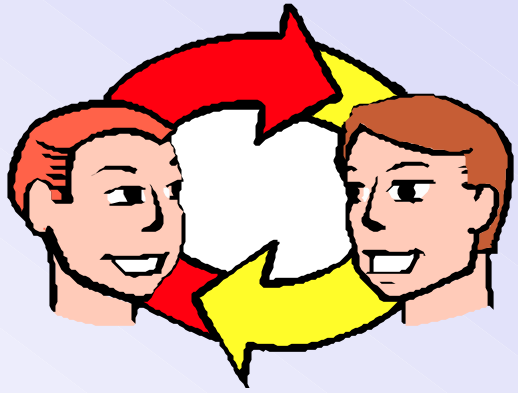
Kommunikation Applet / Servlet

SYSTOR®

Normalfall: textbasiertes HTTP

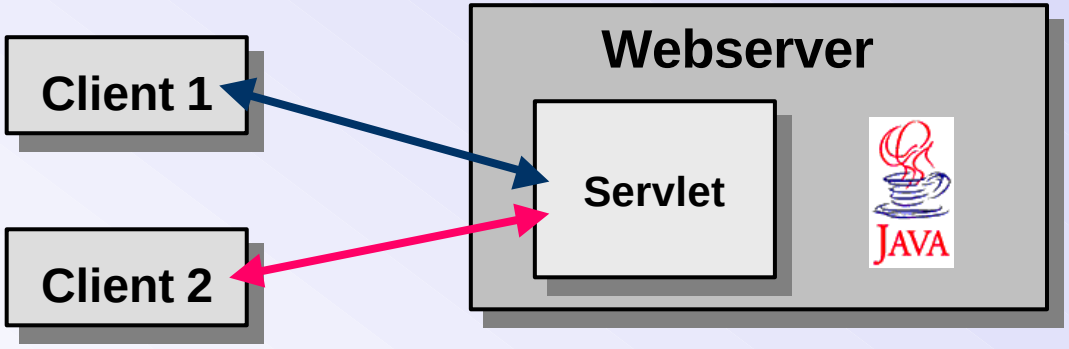
- Einfach!
- Erzeuge im Applet eine URL (HTTP-get oder put)
- Nachteile
  - „Stringify“ von Aufrufen und Parametern
  - dadurch: Verlassen der Objektwelt

- Objektbasiertes HTTP
  - Serialisiere Übergabeparameter
- Sockets
  - „Low level“ Programmierung
- RMI
  - Komplex!



Technisches zu Servlets (2)

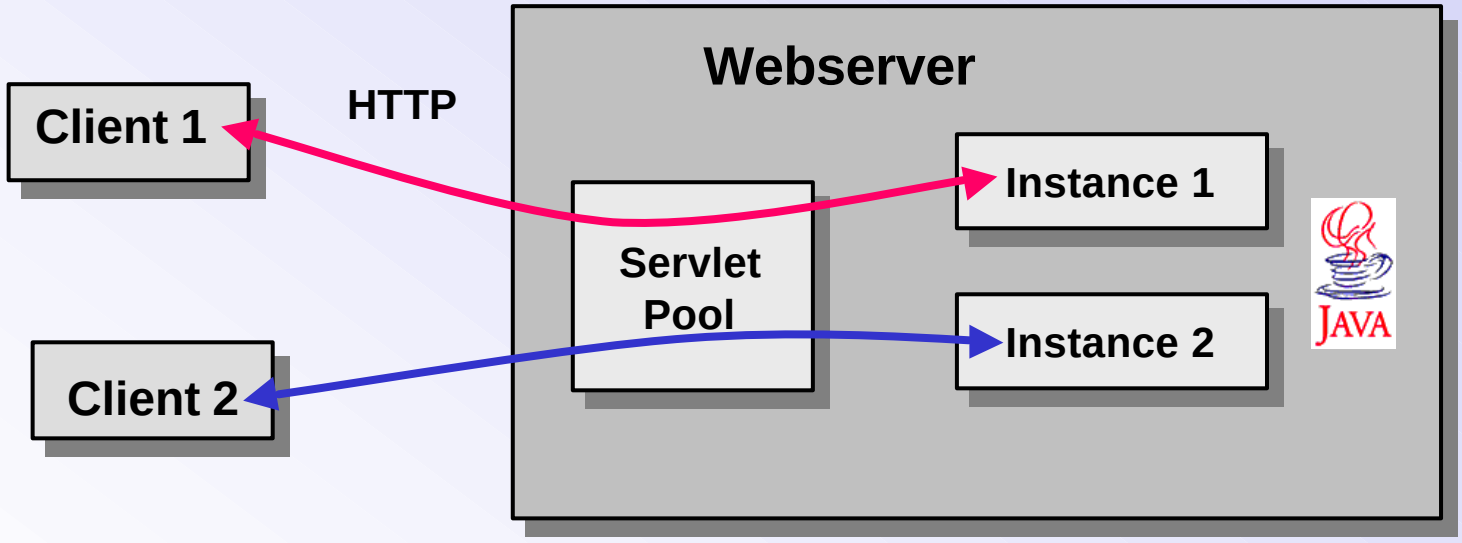
„Normales“ Servlet: single instance for every request



- JEDER Client kann nichtlokale Variablen verändern
  - Beachte: Synchronisation!
- Zentrales „Gedächtnis“

Technisches zu Servlets (3)

Alternativ: single thread modell

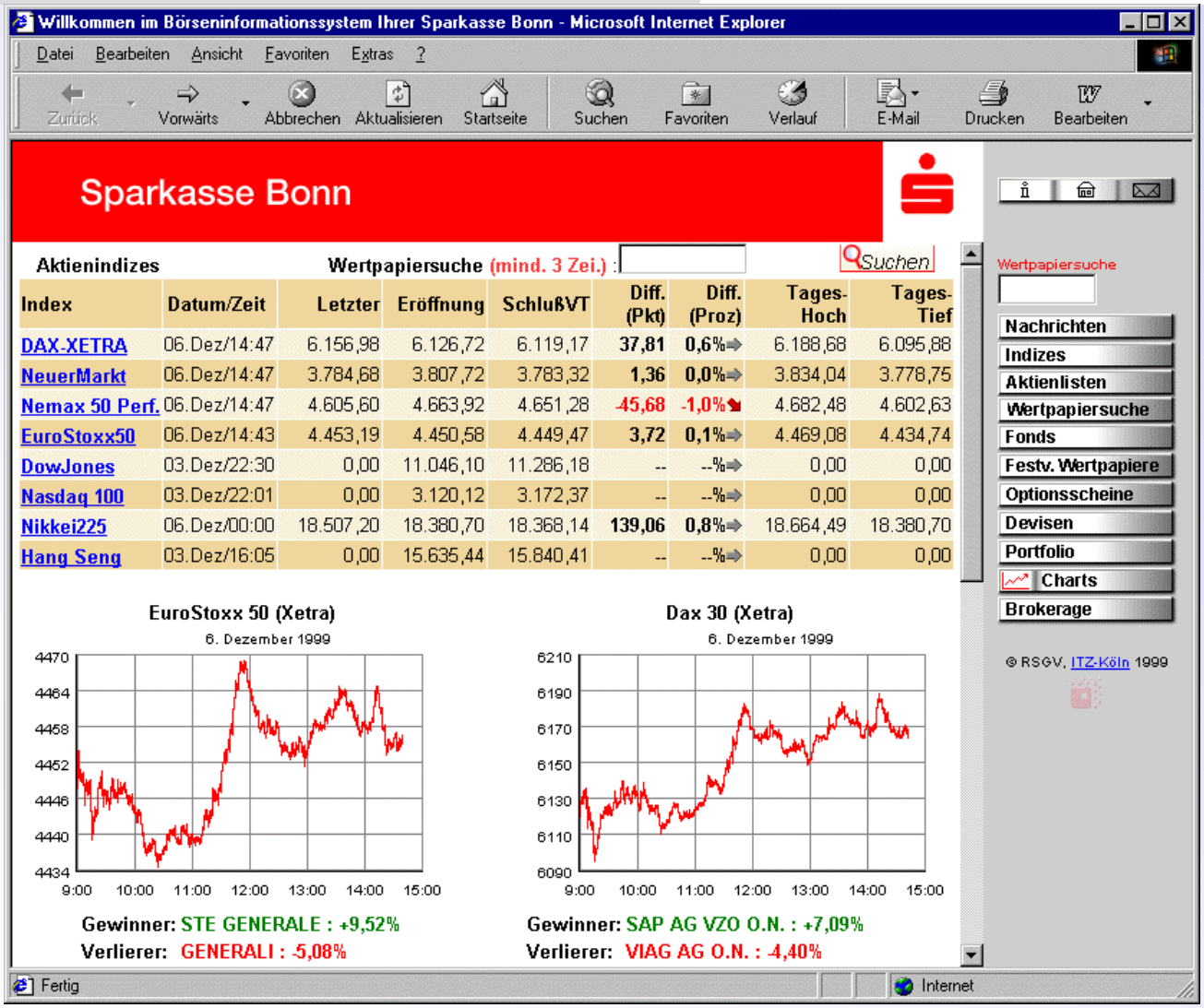


Vorgehen: implementiere Interface:

`javax.servlet.SingleThreadModel`

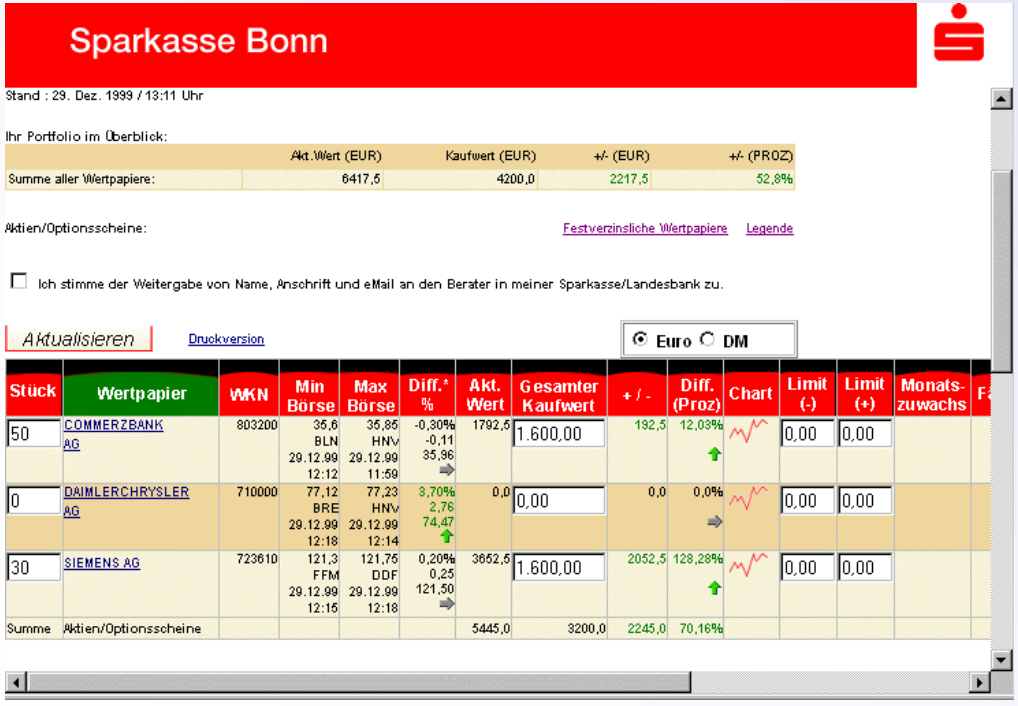
- Garantiert Threadsicherheit
- Keine zentrales „Gedächtnis“

Servlets beispielhaft: ITZ-Börse (1)



ITZ-Börse (2): Integration mit Servlets...

- Relationale Datenbanken
  - permanente Aktualisierung von Kursinformationen über Satellitenleitung
  - Verwaltung der Benutzerdepots (=Portfolios).



Sparkasse Bonn

Stand : 29. Dez. 1999 / 13:11 Uhr

Ihr Portfolio im Überblick:

	Akt.Wert (EUR)	Kaufwert (EUR)	+/- (EUR)	+/- (PROZ)
Summe aller Wertpapiere:	6417,5	4200,0	2217,5	52,8%

Aktien/Optionsscheine: Festverzinsliche Wertpapiere Legende

Ich stimme der Weitergabe von Name, Anschrift und eMail an den Berater in meiner Sparkasse/Landesbank zu.

Aktualisieren Druckversion Euro DM

Stück	Wertpapier	WKN	Min Börse	Max Börse	Diff.* %	Akt. Wert	Gesamter Kaufwert	+ / -	Diff. (Proz)	Chart	Limit (-)	Limit (+)	Monats-zuwachs
50	COMMERZBANK AG	803200	35,6 BLN 29.12.99 12:12	35,85 HNV 29.12.99 11:59	-0,30% -0,11 35,96	1792,5	1.600,00	192,5	12,03%		0,00	0,00	
0	DAIMLERCHRYSLER AG	710000	77,12 BRE 29.12.99 12:18	77,23 HNV 29.12.99 12:14	3,70% 2,76 74,47	0,0	0,00	0,0	0,0%		0,00	0,00	
30	SIEMENS AG	723610	121,3 FFM 29.12.99 12:15	121,75 DDF 29.12.99 12:18	0,20% 0,25 121,50	3652,5	1.600,00	2052,5	128,28%		0,00	0,00	
Summe	Aktien/Optionsscheine					5445,0	3200,0	2245,0	70,16%				

EMail bei Limitüber-/unterschreitung
Mehrere Depots pro Benutzer möglich
Auf Wunsch Transfer an Bankberater

ITZ-Börse (3): Integration mit Servlets...

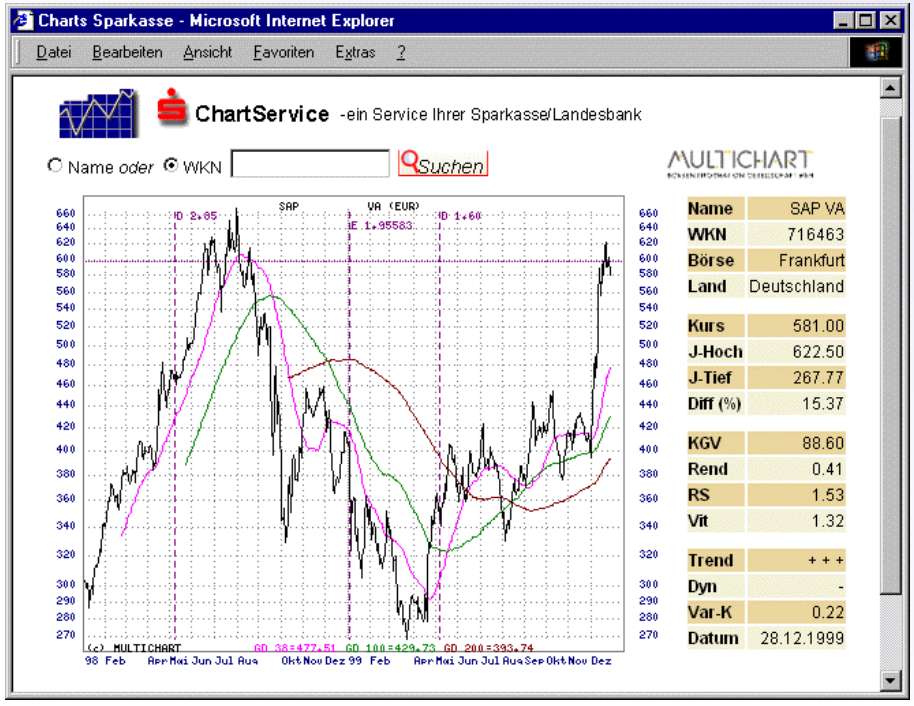
SYSTOR®

- Anwendung für historische Börsendaten
  - werden den Servlets als Tabellen oder als Grafiken geliefert.
- Online-Anbindungen weiteren Datenquellen
  - Fondsgesellschaften
  - Nachrichtenagenturen.
- SMS-Gateway (in Mobilfunk)
- Konfiguration und Betrieb für mehr als 200 Sparkassen
  - inklusive Logging und Accounting

ITZ-Börse: Erfahrungen

Komplexes Umfeld:

- Linux, Solaris und NT
- Teile der Applikation in Smalltalk (VisualWave)
- Integration externer Programme zum Datenempfang via Satellit (C und C++)

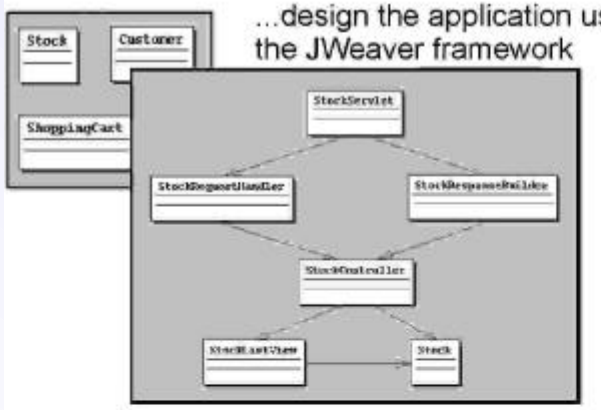


Probleme:

- Performance bei Lastspitzen
- Debugging verteilter heterogener Applikationen

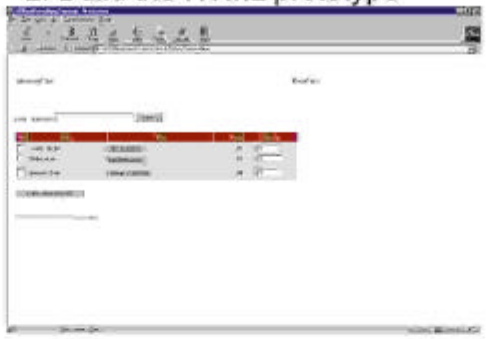
Servlet Framework „JWeaver“

1. Create the BOM and...



...design the application using the JWeaver framework

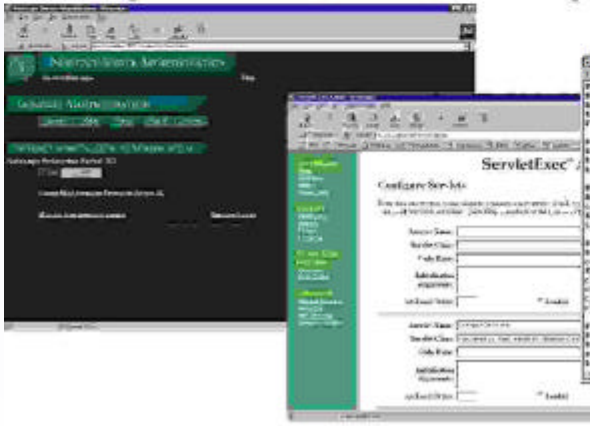
2. Build the HTML prototype



3. Write java code



4. Configure the Web Server / Servlet Engine...



...and the configuration files



5. Run the Servlet application



Servlets: Referenzen und weitere Infos

SYSTOR®

- Jason Hunter: Java Servlet Programming. O‘Reilly 1998

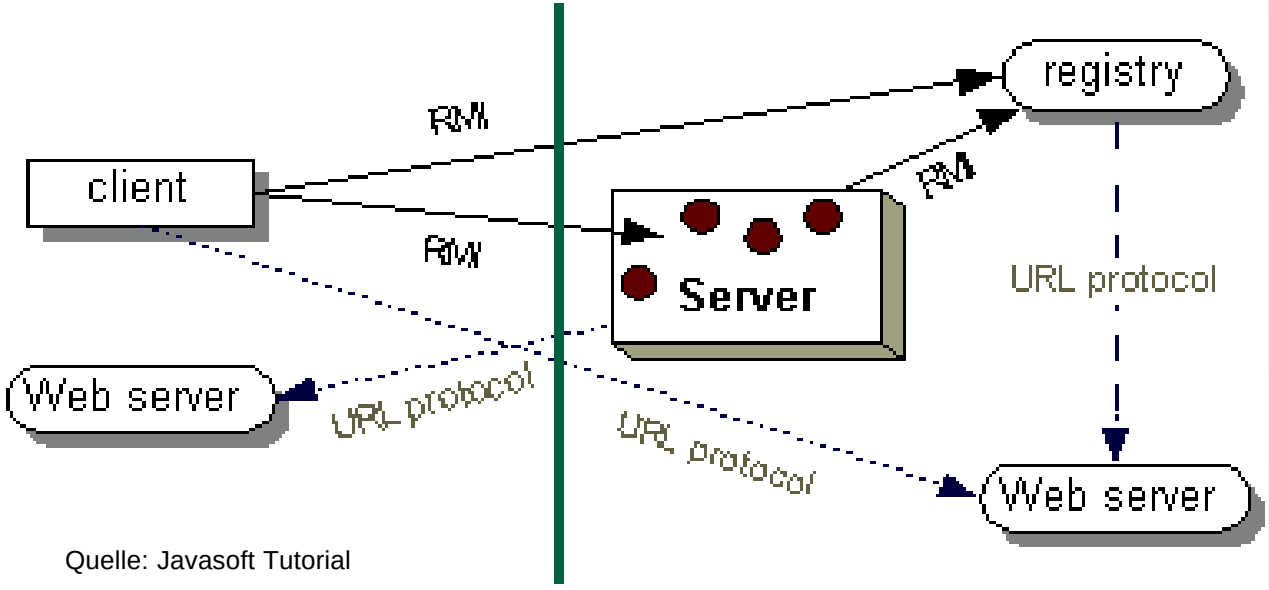
  Hervorragende Einführung in viele (alle?) Aspekte der Servlet- Programmierung.

- Servlet-Section des Javasoft Tutorial (erhältlich unter java.sun.com)

- www.servletcentral.com
- www.servlets.com

Remote Method Invocation (RMI)

- (Java-) Standardisiertes Modell zur Objektverteilung
- Einfach
- Nachteil: Nur Namensdienst (registry)
  - Keine client-/session spezifische Instanziierung (Nachbildung analog EJB-Mechanismen leicht möglich)
  - Keine Tools/API zur Konfiguration und Administration
  - Keine Lastbalancierung, Skalierung, Ausfallsicherung



Quelle: Javasoft Tutorial



Beispiel RMI: Call Center für Finanzdienstleister (1)

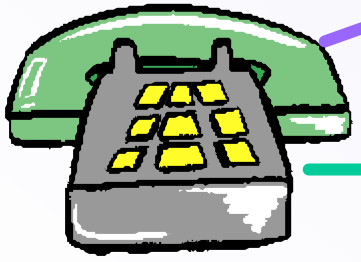
SYSTOR®

Ziele: (Finanz-CC)

- 1.) Effektive Geschäftsprozesse
- 2.) Kundenzufriedenheit

Hohe Anforderungen an Flexibilität

- Neue „Produkte“ im Call Center bearbeiten!
- Aber: „Historische Altlasten“



Sofortbearbeitung

Etwa: Beratung, Auskunft, Buchungen,

Dateneingabe

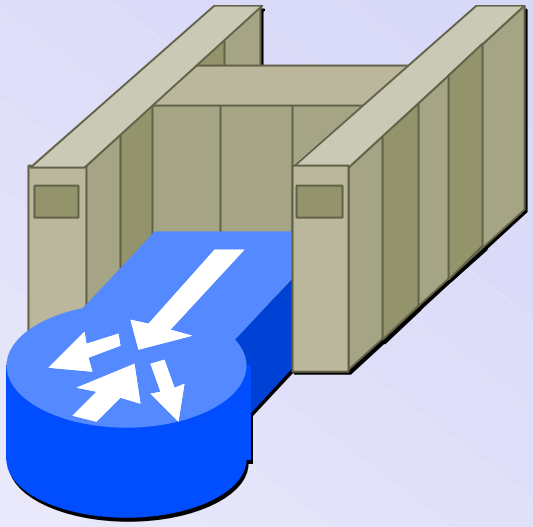
Backoffice

Beispiel RMI Finanz-CC (2): Rahmenbedingungen

SYSTOR®

Voraussetzung: Mainframe als „Backend“

- Typisch für große Firmen
- prägend für Projektstruktur UND technisches Design



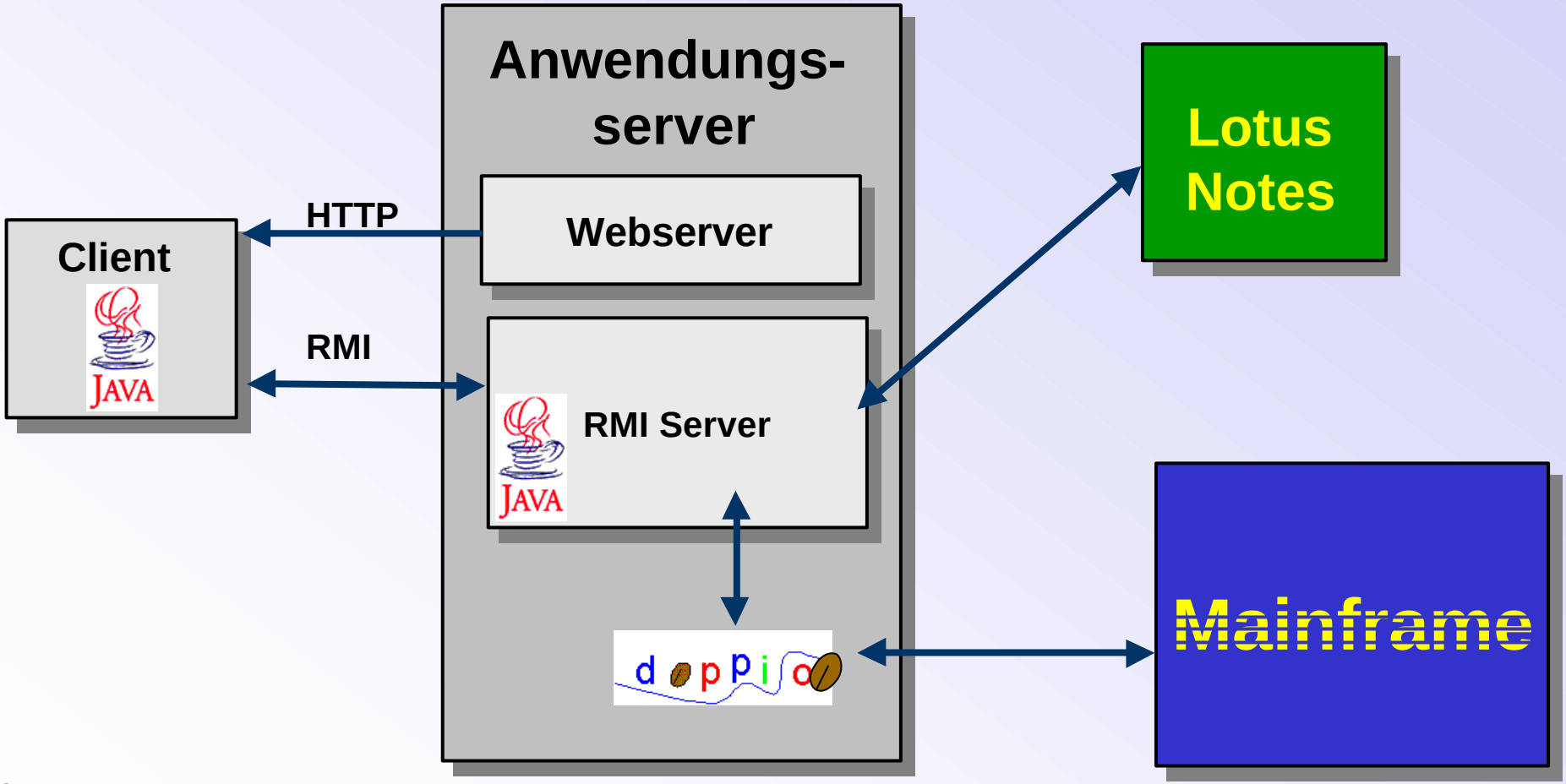
- Nutze bestehende Daten
  - DB2 und IMS-Datenbanken
  - Starker Einfluß auf Datenmodell
- Nutze bestehende Programme
  - IMS und CICS
  - Wichtig: Flexible Middleware (doppio)
- OO-Kapselung „beliebig“ schwierig

Beispiel RMI Finanz-CC (3): Rahmenbedingungen

SYSTOR®

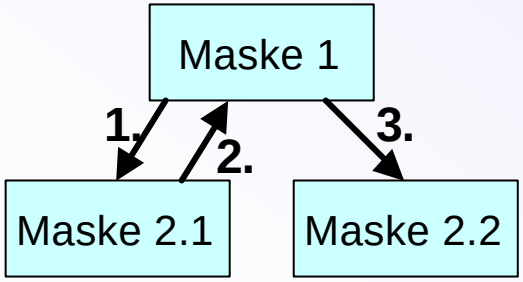
- Wunscharchitektur „EJB & Application Server“
  - Eingeschränkte Produktauswahl: „ **I** T-mäßig **B**lue-**M**inded “
- EJB-Prototyp (Anfang 1999) mit kritischen Ergebnissen:
  - (Einizelne) EJB-Tools instabil und wenig ausgereift
  - Produktsupport (war) schlecht
  - Kaum Praxis-KnowHow verfügbar
- Resultat: RMI als „Zwischenlösung“
  - Migration nach EJB „as soon as possible“
  - Designziel: Migrationsfähigkeit
- Projektumfang: (Technisches Design, Implementierung, Test): >> 4000 PT
  - Fachliche Analyse, Produkttest, Einführung: > 1000 PT)

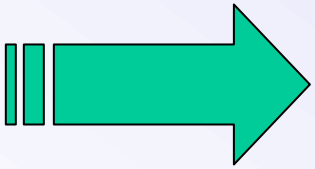
Beispiel RMI Finanz-CC (4): Systemüberblick



Beispiel RMI Finanz-CC (5): Einsatz von RMI

- Instanziierung von Objekten analog EJB-Mechanismus
  - Über (Singleton) „HomeInterfaceContainer“
  - Fachobjekte: Per Client
- Verteilte MVC Architektur
  - Pragmatische Aufteilung des Model zwischen Client und Server
  - Aufwendig: Koordination von Dialogfluß und Modellaufbau (Vor-/Zurück Navigation zwischen Masken)





Führt leicht zu inkonsistenten oder unerlaubten Modellzuständen!

Erfahrungen Finanz-CC (1): RMI auf dem Prüfstand

SYSTOR®

- Sehr robust (IBM JDK >1.1.7 unter NT, >1.1.6 unter AIX)
- Fehlende Administrations-/Konfigurationswerkzeuge
  - Beispiel: Parametrierung verschiedener Testserver
- Kritisch: Garbage Collection auf Server
  - Durch Klassenmodell bestanden „Verweisketten“ (DGC erkennt solche nur teilweise)
  - Problematisch, weil Server 7x24h läuft (...laufen soll)
  - Distributed Garbage Collector (DGC) benötigt Unterstützung durch Designer/Entwickler
  - Lösungen sind schwer testbar!

Erfahrungen Finanz-CC (2)

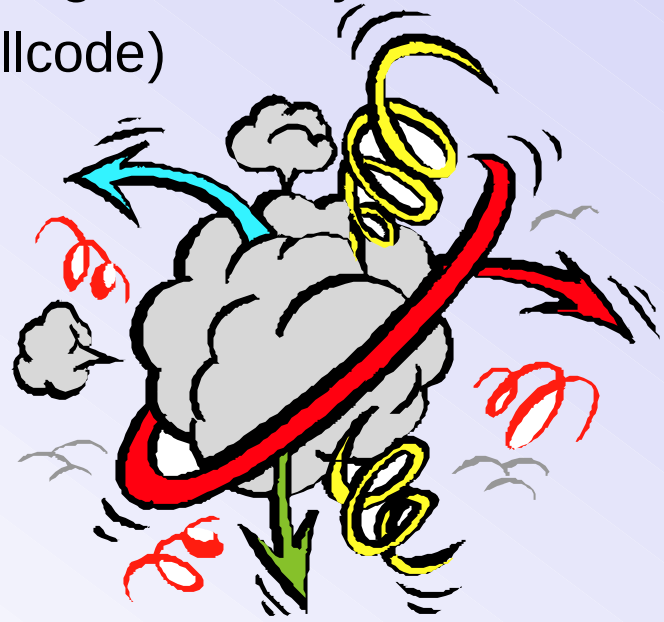
SYSTOR®

- VisualAge/NT als Entwicklungsplattform
  - NT 4.0 (>=SP4), 256MB, 360MHz (knapp) ausreichend
    (für Debugger zu wenig/zu langsam)
  - Extrem robustes Repository
    (unter AIX, >> 1000 Entwicklertage ohne Ausfall)
  - Entwicklungsumgebung (ziemlich) stabil
    (Im Mittel alle 6 Wochen Neuinstallation erforderlich)
  - Hervorragender Debugger, auch für Remote-Objekte
    (Änderungen an Klassen während einer debug-Session möglich)
  - Wechsel der Swing-Version nicht möglich
- Enttäuschend: Rational Rose
  - Reengineering-Komponente (hier) nicht nutzbar
    (verträgt package-Namen der Form de.firma.projekt.komponente nicht)

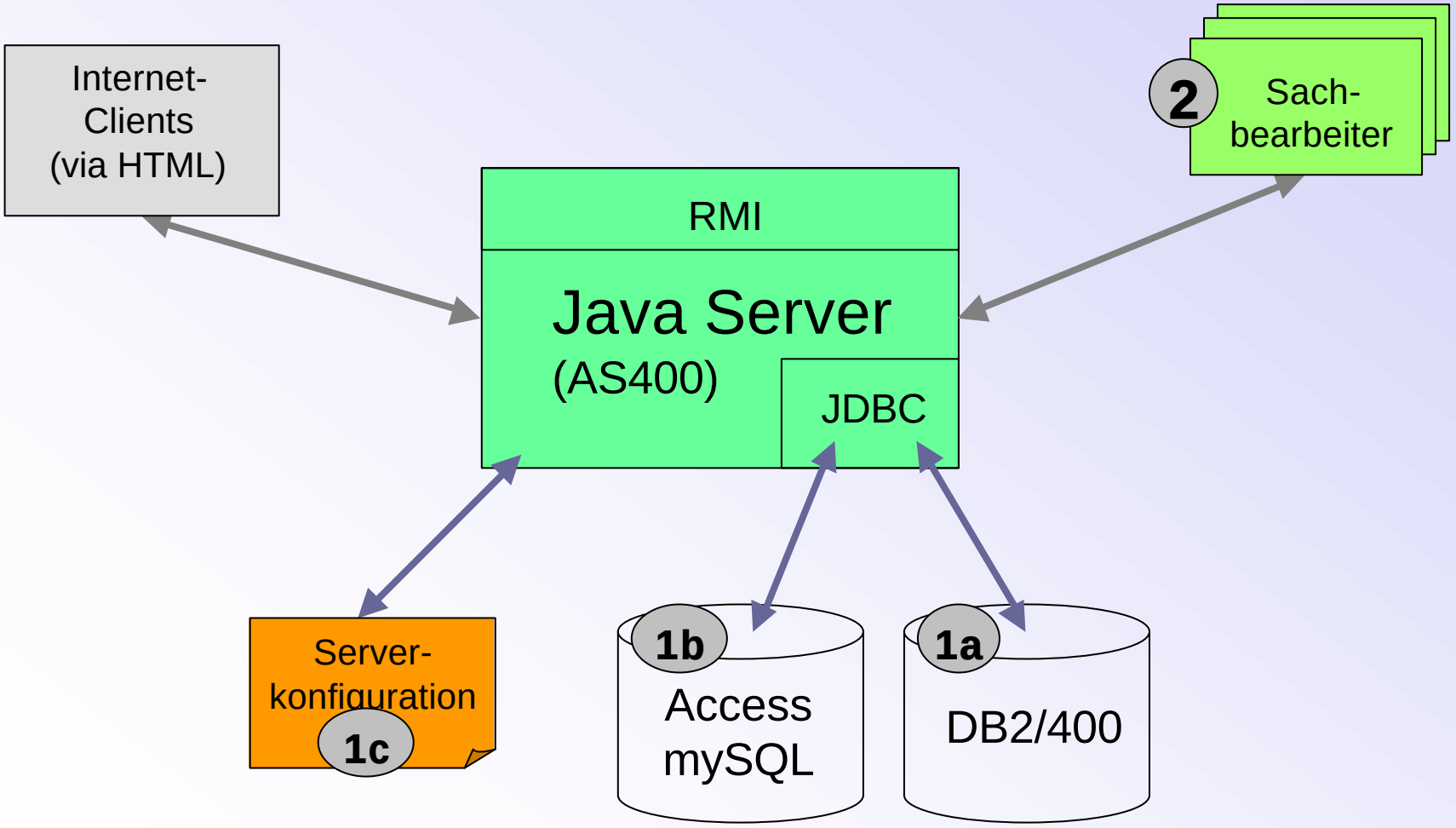
Erfahrungen Finanz-CC (4)

Heterogene Integrationsprojekte: Welten prallen aufeinander...

- Cobol-Guru versus Java-Freak
  - Schwierig: „human factors“
- Versions- und Konfigurationsverwaltung „out of sync“
  - VisualAge Repository (nur Java Quellcode)
  - Endeavour (Mainframe)
  - HTML, config-Files, sonstige Skripte
- Management der Betriebssysteme
  - Unix: ps, kill, reboot
  - MVS: RZ-Telefonhotline...
- Testmanagement und -strategie



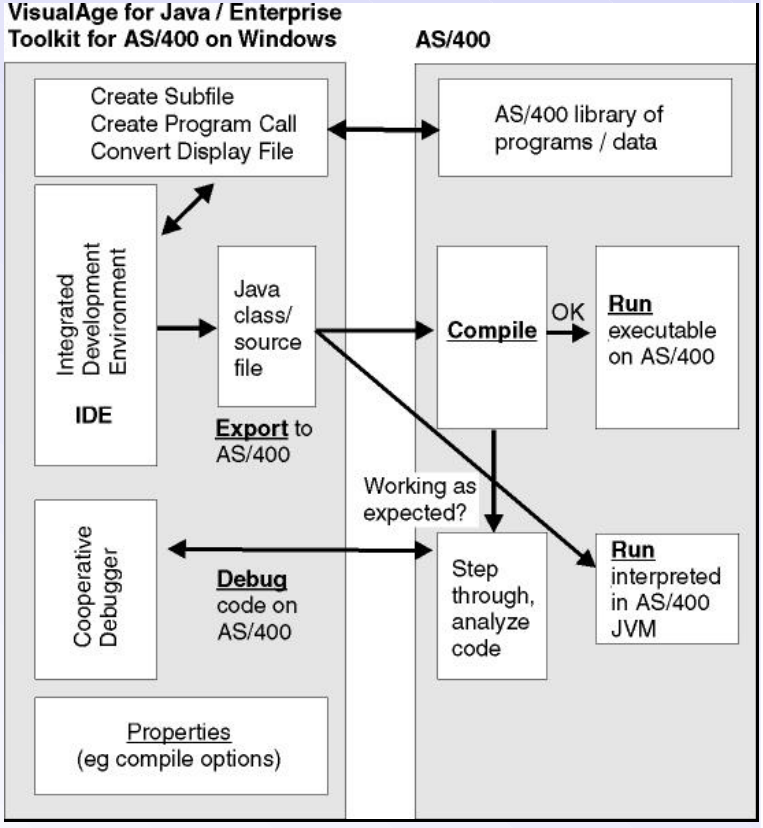
Erfahrungsbericht „Java auf AS400“



Erfahrungen mit Java & AS400 (1)

- Laut IBM gehört AS400 zu den performantesten Java-Plattformen (stimmt, für GROSSE Hardware)
- JDK als Bestandteil des Betriebssystems OS/400 (V4R4 enthält 1.1.7, V4R5 entweder 1.2 oder 1.3.1)
- Entwicklungsunterstützung durch VisualAge/Java:
  - Enterprise Toolkit für AS400
  - AS400 Toolkit für Java

    (u.a. Zugriff auf sämtliche AS400 Resourcen, Programme, Daten, Dateien, Queues)



Erfahrungen AS400 & Java (2)

- CLASSPATH unter OS400:
  - Beachte: „normale“ classes.zip genügt nicht!
  - Konzept der „Umgebungsvariablen“ ungebräuchlich...
- Shell (namens Qshell) mit eingeschränkten Möglichkeiten
  - kein make
  - Basis: 5250-Session
- Alternative JDBC Treiber verfügbar
  - native DB2 sowie Pure-Java DB2
- Registry braucht eigene 5250 Session
- Gute Kompatibilität zwischen Server-Code

Fazit Java & AS400

- Funktioniert (wie erwartet) problemlos
- Im Einzelfall entscheiden!
  - Kriterien:
    Wartung durch (bisherige) AS400 Teams
    Auslastung der vorhandenen Maschinen
    Performance
    Anteil an „vorhandenen Systemen“
- Auf jeden Fall: Anwendung pilotieren

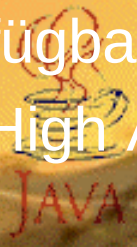
RMI: Resourcen und weitere Informationen

SYSTOR®

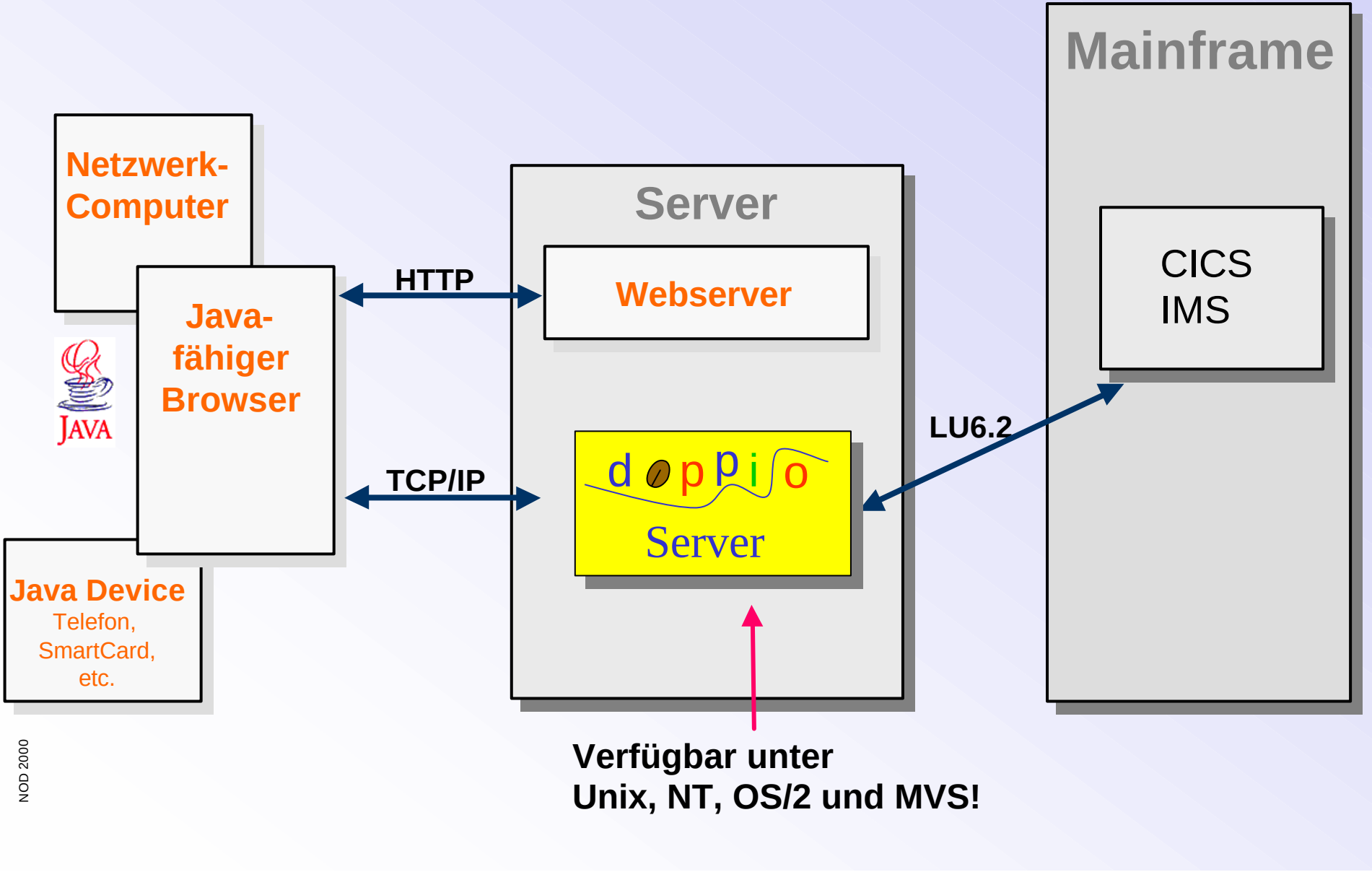
- JDK 1.1 Dokumentation: HelloWorld und StockQuote Beispiel
- Javasoft Tutorial: Beispiel zur verteilten Berechnung von Pi
- java.sun.com, RMI Spezifikation
- RMILite Implementierung von MetaMata: www.metamata.com
- Bruce Eckel: Thinking in Java. Prentice Hall 1998. Hervorragende Einführung in Java, kurzes und prägnantes Kapitel über RMI

Fazit: Server Side Java ist …

- Praxistauglich: Reif für kommerziellen Einsatz
  - Performant, stabil
  - Für alle wichtigen Plattformen verfügbar
  - Aber: Offene Punkte hinsichtlich „High Availability“
- Effektiv und flexibel:
  - Ideale Integrationsplattform
  - Programmlogik unabhängig von Technik (Servlet, RMI, EJB, CORBA)

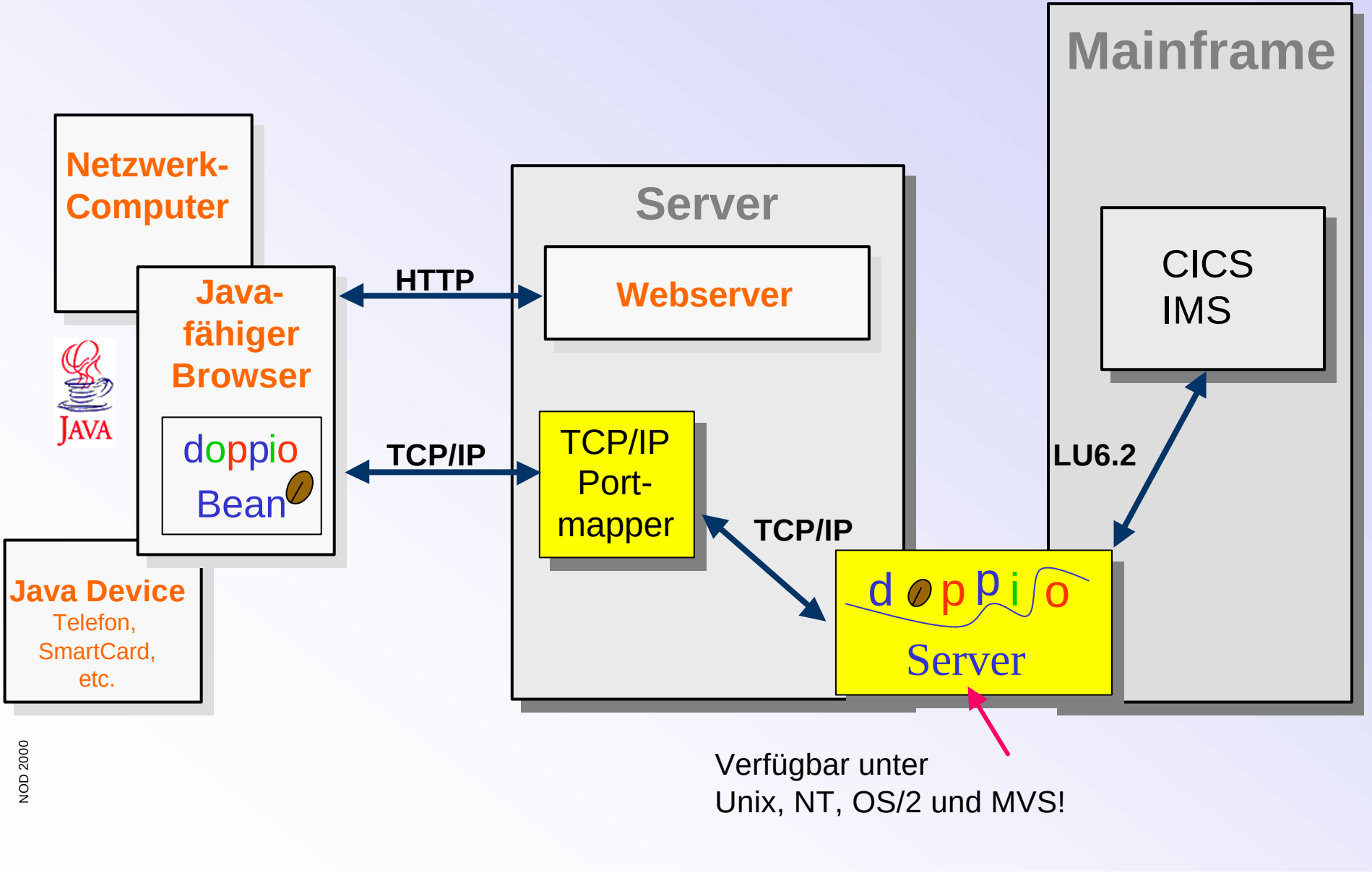


Anhang 1: Übersicht: doppio



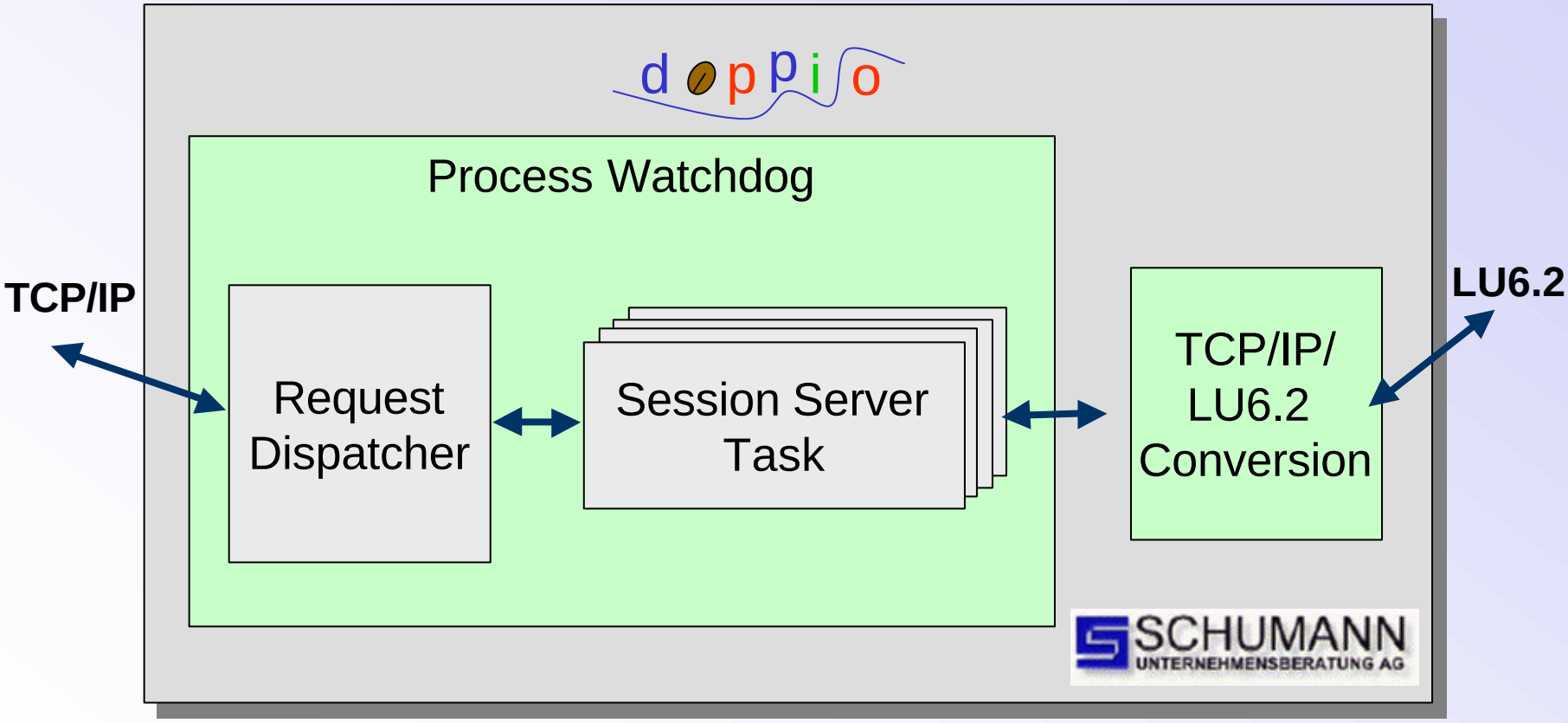
NOD 2000

Übersicht: doppio (2)



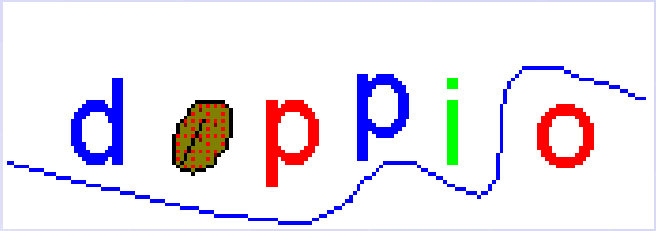
NOD 2000

Struktur von doppio



Java-To-Mainframe per „Espresso“

SYSTOR®



- Skalierbar & Performant
- Bedient CICS und IMS
- (Meist) ohne Änderung der Mainframe-Applikationen
- Reines Kommunikationswerkzeug

- In Projekten der Schumann AG entstanden
- Zufriedene Referenzkunden!