Schnittstellen: Das A&O für Enterprise Application Integration

Andres Koch
dipl. El. Ing. HTL / M. Math
Object Engineering GmbH,
Uitikon-Waldegg, Schweiz
Email: akoch@objeng.ch
www.objeng.ch

Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

Inhalt

Schnittstellen:
Das A&O für Enterprise Application Integration

- Einleitung
- Architektur
- Kommunikation
- Middleware
- Schnittstellen
- Praxis-Beispiele
- Zusammenfassung

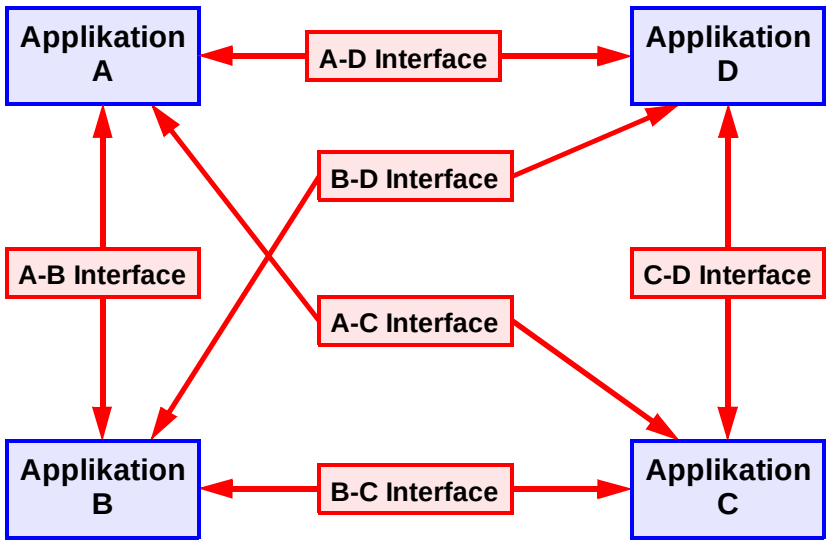
Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

Einführung

- An erster Stelle steht die IT-Architektur inklusive Organisation, Normierung und der Festlegung der zu verwendenden Technologien.
- Der Blueprint der Fahrtwege und Anschlüsse ist vorrangig.
- EAI-Blueprint ist vergleichbar mit dem unternehmensweiten Datenmodell!
- Mit jeder neuen Technologie-Welle (Hype) ändert “nur” das “Rollmaterial” und eventuell die Fahrgeschwindigkeit.
- Somit steht die Technologie an zweiter Stelle
- Technologien sollten auf Industriestandards basieren und nicht proprietären Tools folgen.
- Sich auf die Technologie zu konzentrieren birgt Risiken, welche später teuer zu stehen kommen.
- EAI und Legacy gehen Hand in Hand (typisch für Hostanwendungen).
- Integration von bestehenden Anwendungen als WEBServices erfordert ebenso EAI-Disziplin.
- Also viel Organisation und eine harte Doktrin ist nötig, wenn auch Pragmatik nicht fehlen darf.

Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

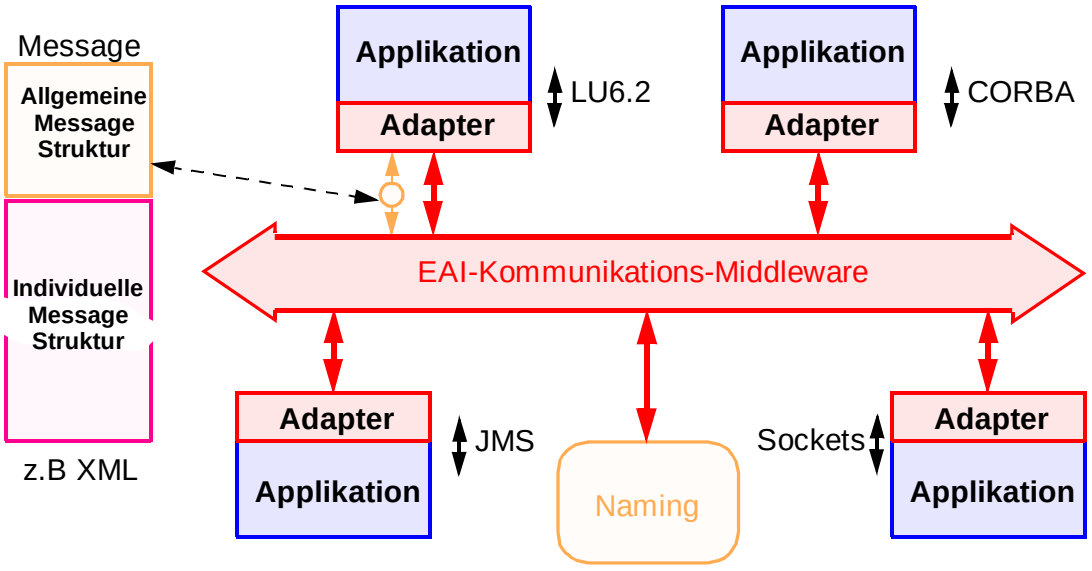
Architektur: Unerwünschte EAI-Architektur



- Eine individuelle Schnittstelle zwischen jedem Applikationspaar ist unerwünscht und aufwendig.

Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

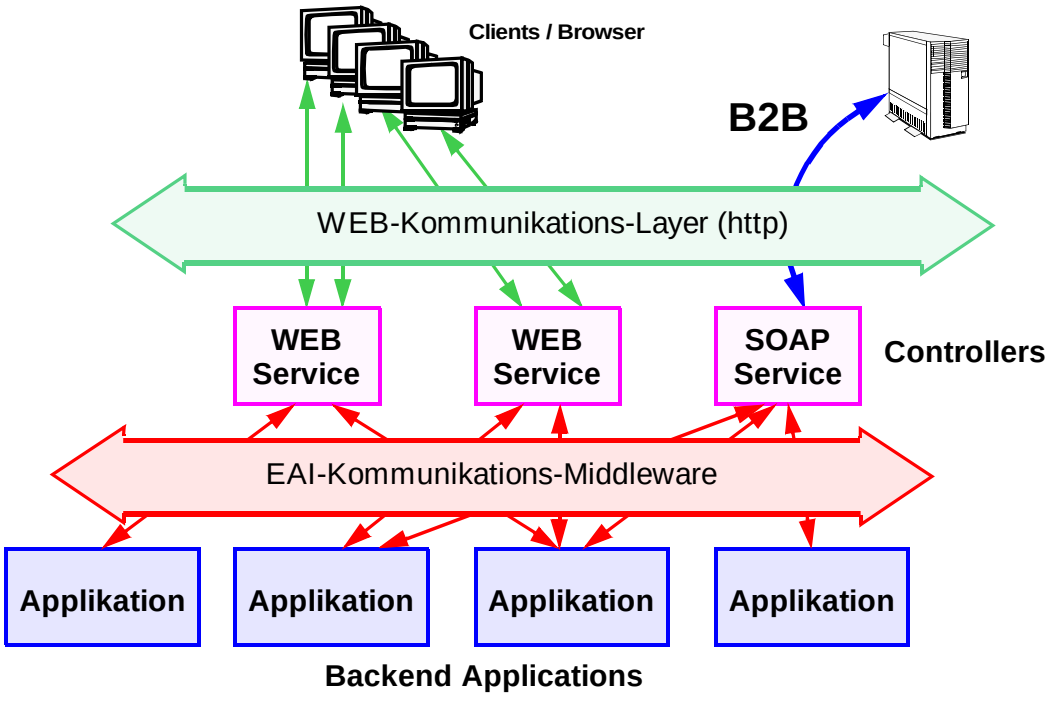
Architektur: Generelle “Bus”-Architektur



- Gemeinsame Komponenten so gut wie möglich generisch halten.
- Meldungsstrukturen normieren (Standard-Header, Individueller Body)

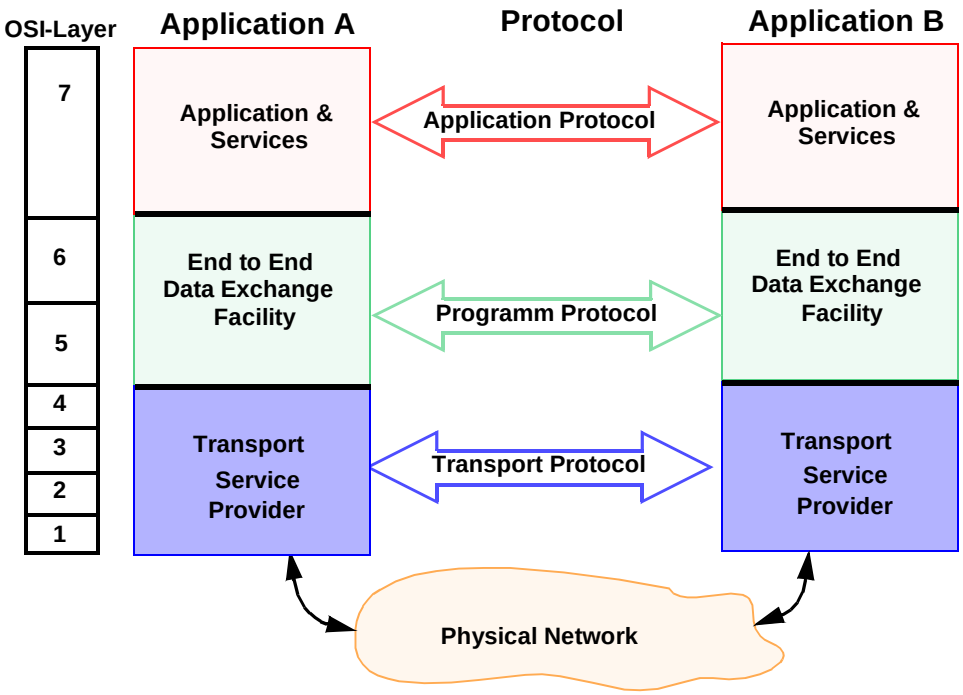
Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

Architektur: Multi-Tier-Architektur



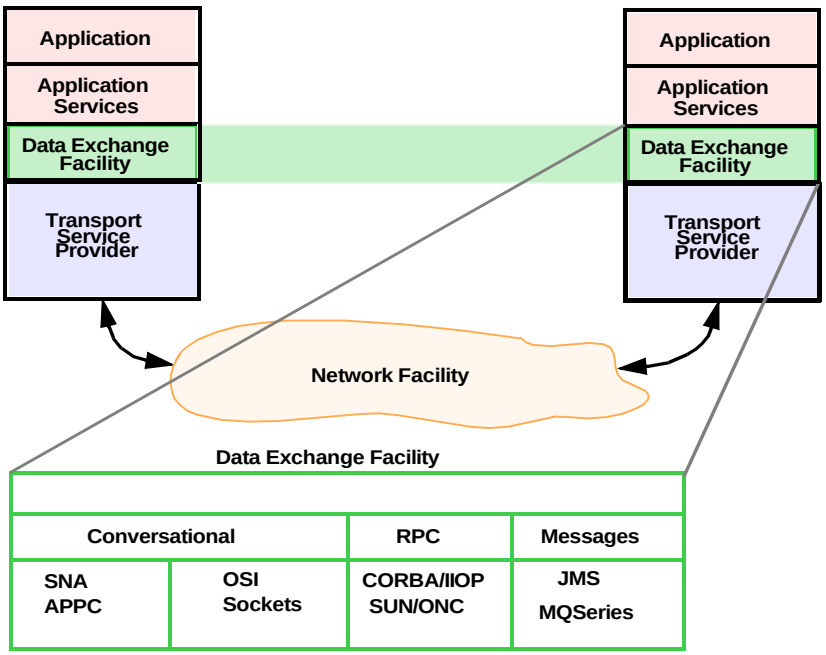
Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

Kommunikation: Drei Ebenen



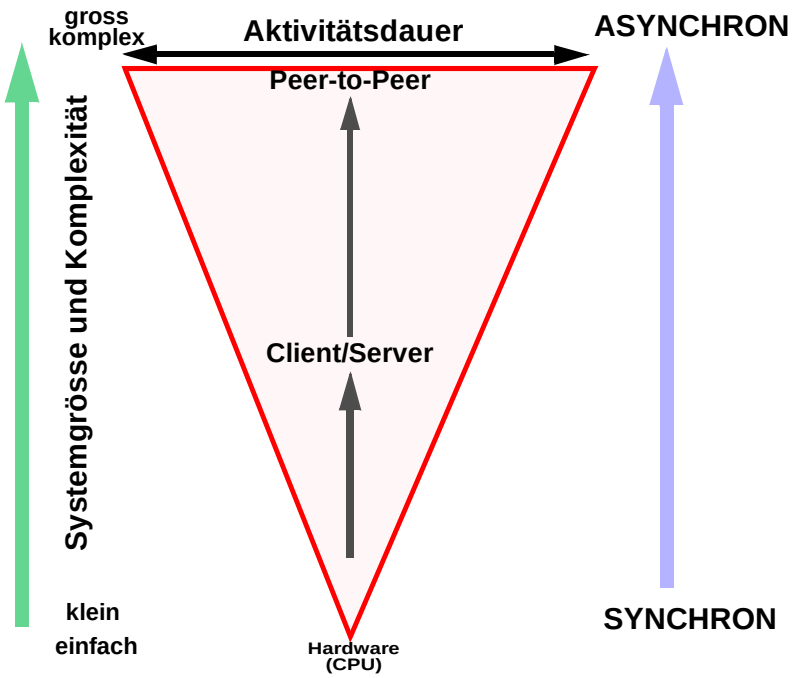
Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

Kommunikation: (A2A) Application to Application



Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

Kommunikation: asynchron oder synchron?



Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

Middleware: Überblick

Generell:

- Möglichst “standardisierte” Technologien benutzen
- Proprietäre Werkzeuge schränken ein
- Es muss nicht immer das Neueste sein, Altbewährtes ist oft besser und nicht zu unterschätzen.

Mögliche Varianten A2A:

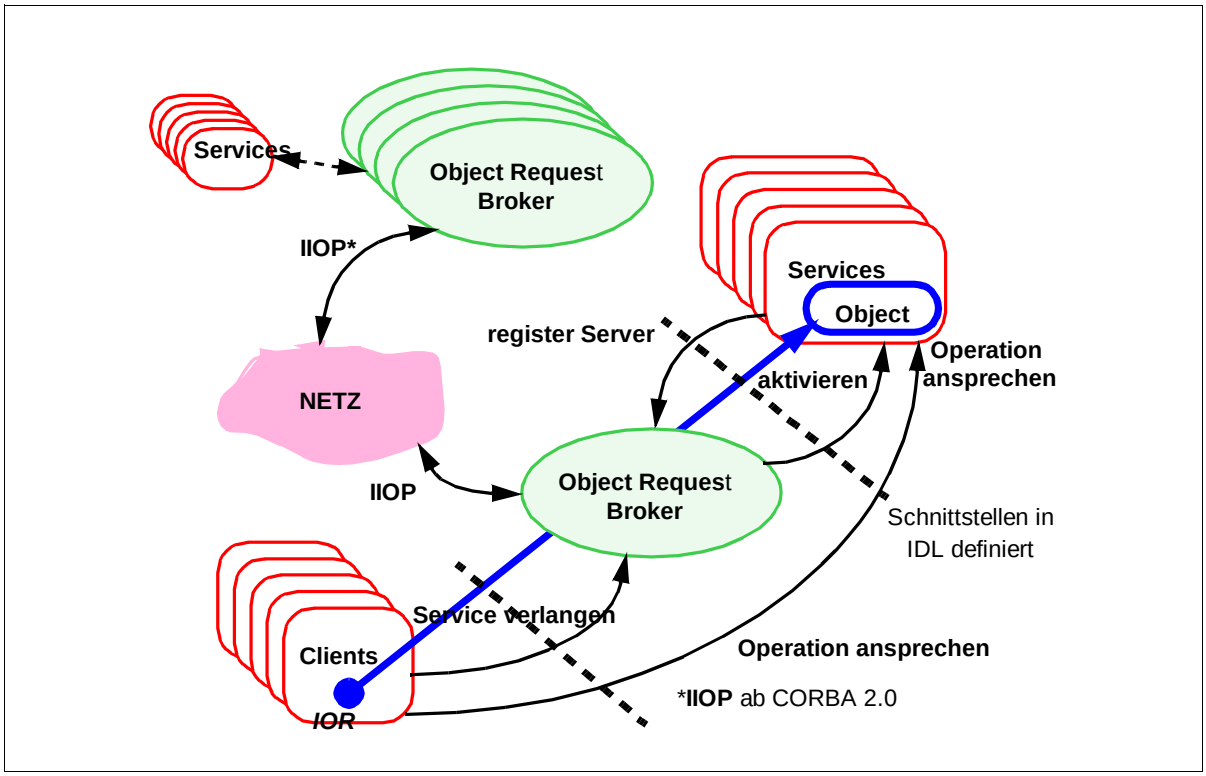
- CORBA basierte Middleware: “*CORBA ist tot, es lebe CORBA*”
- CORBA (IIOP): Sprachen-, System- und Hersteller-übergreifend (Interoperabilität)
- JMS-basierte Systeme basieren auf verschiedenen Messaging-Systemen, sind aber ohne dazutun auf Java beschränkt.
- Socket-Kommunikation funktioniert fast immer, ist aber sehr *low level.*

Mögliche Varianten B2B:

- SOAP (HTTP basiert)
- SMTP nicht unterschätzen wenn verschlüsselte Meldungen verwendet werden.
- Meldungsstrukturen auf XML fördern die Flexibilität aber im schlechten Fall auch das Normierungs-Kopfweh

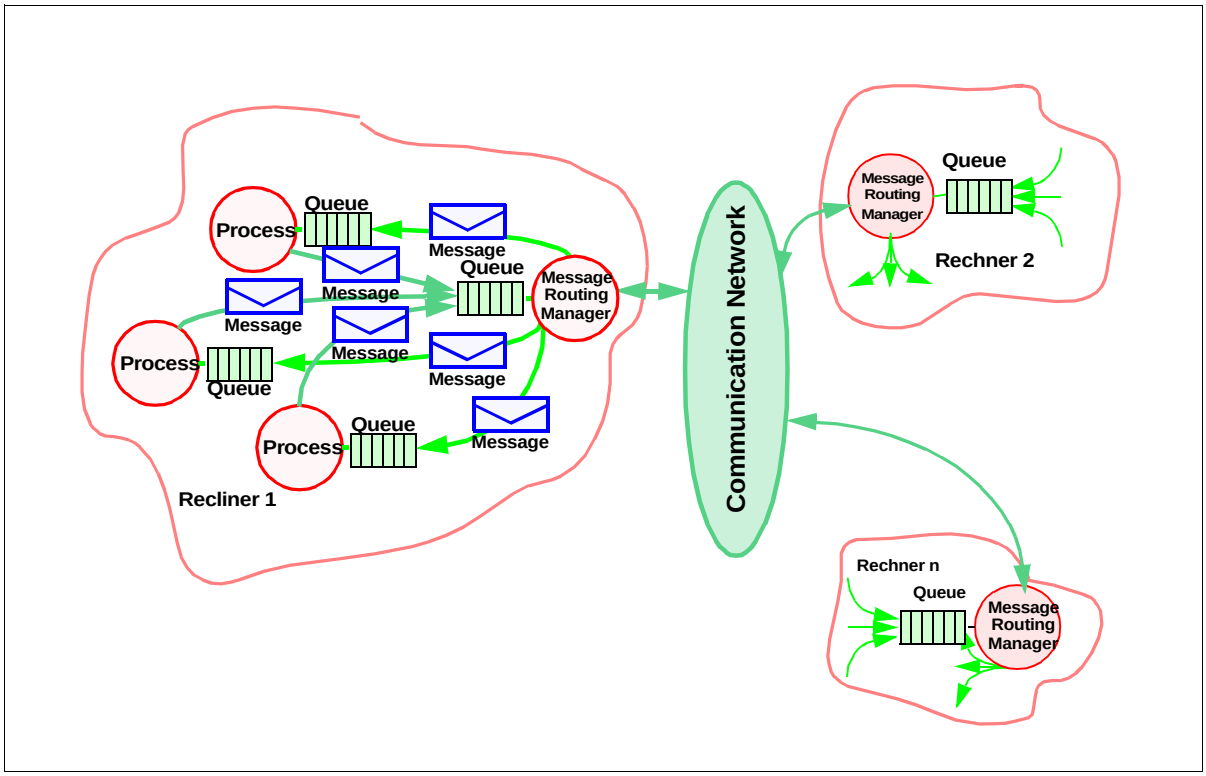
Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

Middleware: CORBA



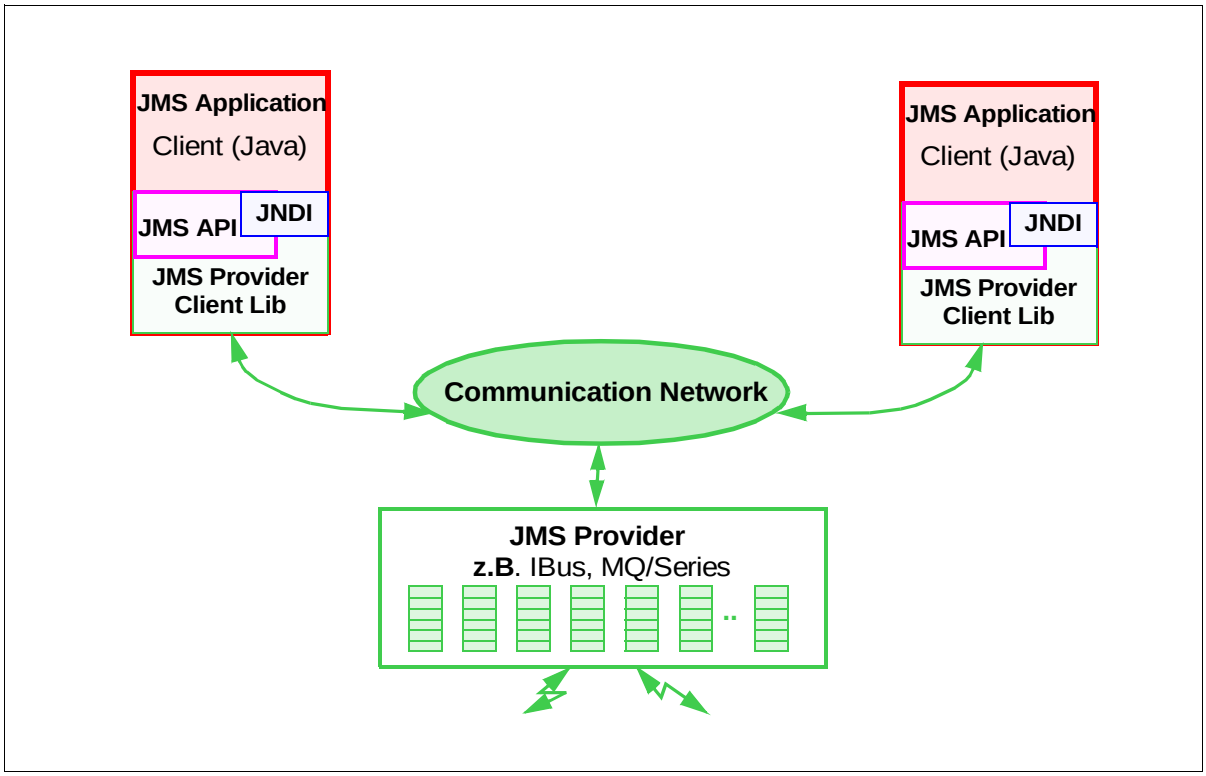
Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

Middleware: Messaging



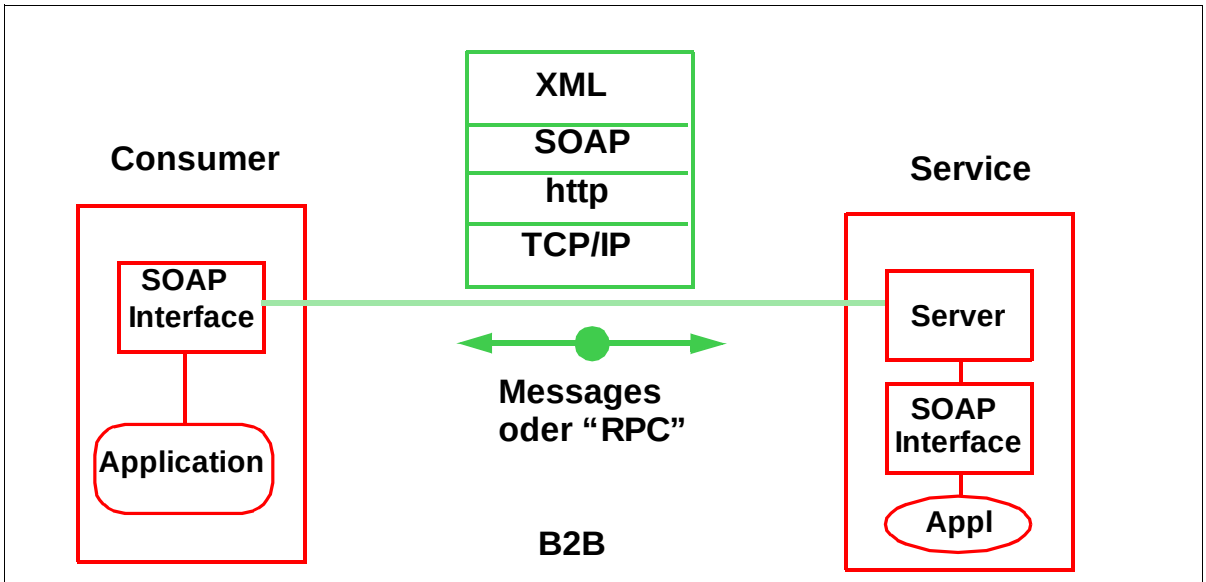
Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

Middleware: JMS-Architektur



Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

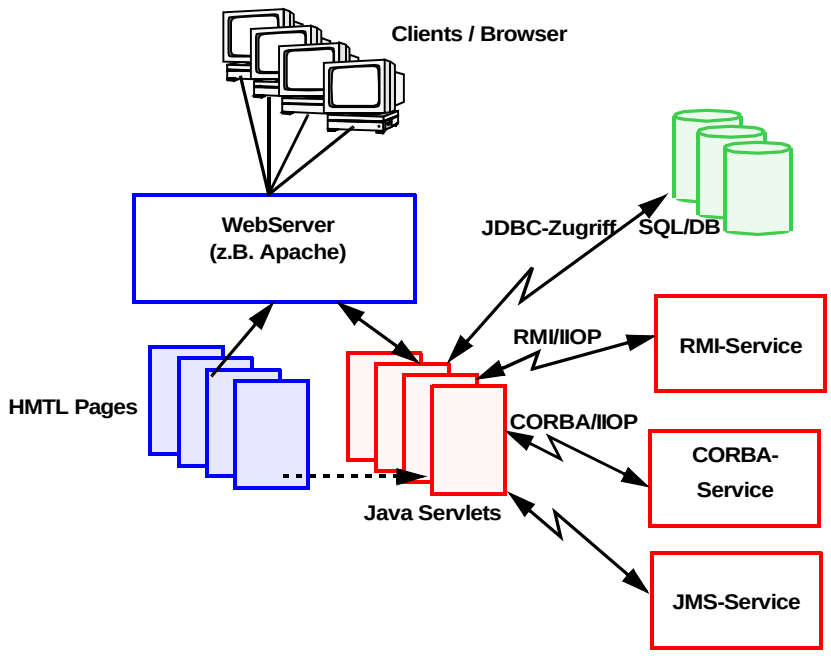
Middleware: Simple Object Access Protocol



- Stark im Trend
- Speziell für *Business to Business* Kommunikation
- Kommunikation quasi normiert, Meldungstrukturen sind “*frei*”

Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

Middleware: WebServer, Servlets und EAI



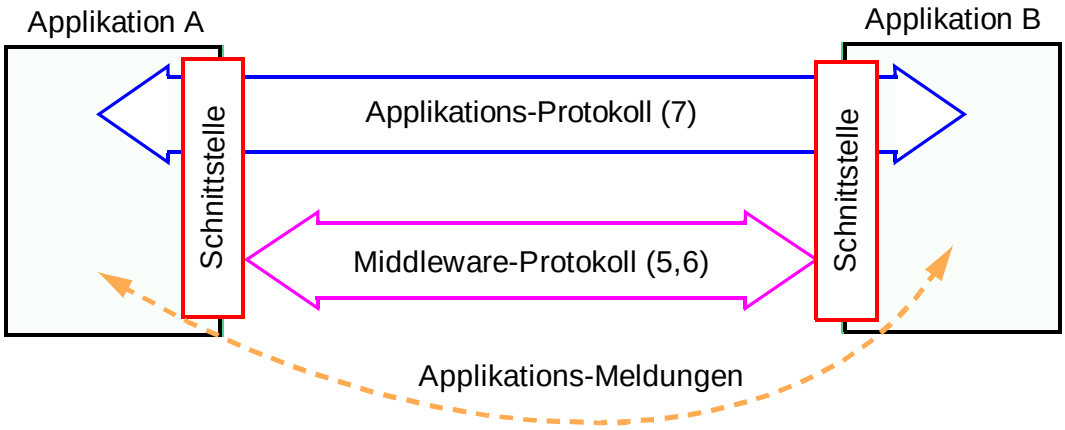
Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

Schnittstellen: Generell

- “Programm to an interface, not an implementation” (Gamma etal)
- Architektur und Schnittstellen gehören zusammen.
- Verteilte Applikationen benötigen Schnittstellen
- Schnittstellen tendieren in der Regel komplex zu werden.
- Oft wird über die Technologie (u.a) mehr diskutiert, als über das Design von Schnittstellen nachgedacht wird.
- Das Design von Schnittstellen von Anwendungen haben oft Mängel, welche sich erst nach geraumer Zeit zeigen.
- CORBA-IDL anzuwenden und daraus Client und Serverseite zu implementieren ist einfaches “Handwerk”.
- Verifikation von Schnittstellen ist wichtig und spart später unnötigen Aufwand.
- Changemanagement gehört zur grossen Herausforderung für verteilte Systeme und somit für EAI.
- Schnittstellen-Aenderung wirken sich praktisch immer auf Client- und Service-Seite aus.
- Schnittstellen welche die Implementation zeigen, werden zum Pferdefuss beim weiteren Ausbau
- Das Erfolgsrezept bei Schnittstellen liegt in **“Einfachheit“**

Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

Schnittstellen zwischen Applikationen



- Middleware-Protokoll ist weitgehend normiert (IIOP, JMS u.a.)
- Das Applikations-Protokoll ist aber frei definierbar und erfordert individuelle Anpassungen.
- Applikationsmeldungen in XML verhindern häufige Anpassung der Schnittstelle.

Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

Schnittstellen: Entwurf

- Den Schnittstellen von Modulen sollte in der Entwurfsphase genügend Aufmerksamkeit und Sorgfalt gewidmet werden.
- Eine Schnittstelle, die während der Entwurfsphase ändert, ist vertretbar, dies nach deren Inbetriebnahme zu tun kann fatal sein.
- Schnittstellen dürfen nur erweitert werden, dass eine Aufwärtskompatibilität sichergestellt ist.
- Meldungsformate entsprechen den Schnittstellen im meldungsorientierten Paradigma (Formate in XML)
- Prozedur-, Funktions- oder Methoden-Aufrufe einer Komponente entsprechen den Schnittstellen im objektorientierten oder prozeduralen Paradigma (RPC).
- Die Abfolge der Benutzung der Schnittstellen entspricht dem Protokoll auf höherer Ebene (Applikations-Protokoll).
- Mit der CORBA-Interface Definition Language (CORBA-IDL) Schnittstellen können definiert werden, die unabhängig von der in der Implementation verwendeten Programmier-Sprache ist.

Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

Schnittstellen: Meyer's Entwurfs-Kriterien

Wenige Schnittstellen

- Jedes Modul sollte mit möglichst wenigen andern Modulen kommunizieren.
- Beschränkte Gesamtzahl der Kommunikations-Kanäle zwischen Modulen:

  Anzahl der Operationen (in interfaces) in Grenzen halten

Schmale Schnittstellen (Lose Kopplung)

- Wenn zwei Module überhaupt kommunizieren, sollten sie so wenig Information wie möglich austauschen.

  Anzahl der Parameter der Operationen beschränken (<=5)

Ref: Bertrand Meyer

Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

Schnittstellen: Changemanagement

Beim Erstellen

- Schnittstelle so generisch wie möglich halten
- Keine zu hohe Granularität wählen (Netzwerkzugriff)
- Nur den unmittelbaren und bekannten Bedarf abdecken. (*If-Use Cases*)
- Spätere Erweiterungen ist leichter möglich
- Einfachheit ist auch hier der Schlüssel zum Erfolg.

Beim Aendern & Erweitern

Wenn eine Schnittstelle erweitert wird, dann:

- neue Strukturen einfügen, statt bestehende ändern
- neue Operationen, statt bestehende ändern
- durch "Vererbung" erweitern, damit bestehende Schnittstellen beibehalten werden können

Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

Schnittstellen: Flexibilität vs. Performance

Performance

- Flexibilität und Performance schliessen sich fast gegenseitig aus
- Man braucht beides und muss optimieren
- Netzwerk-Calls müssen berücksichtigt werden für Performance
- Verarbeitungszeit muss berücksichtigt werden (Millionen von Elementen)
- Kardinalität in den Dutzenden kann vernachlässigt werden, aber im Bereich von Tausenden oder Millionen wird ihr System “ruiniert”.

Flexibilität

- Verwendung von NamedValue oder XML-Strukturen
- Unbegrenzte Listen (sequences in CORBA-IDL)
- Schnittstellen Trennung nach Bereich
- Mischen zwischen schnellen und flexiblen Mechanismen
- XML kombiniert mit CORBA für die Zukunft als flexible Lösung

Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

Schnittstellen: IDL-Schnittstelle

die *IDeaLe* Schnittstelle

- Die Schnittstelle wird einmal definiert
- genau einmal
- und ändert während des ganzen Lebenszyklus des Systems nie mehr
- dies entspricht nicht der Realität und ist nicht voraussehbar.

Einfluss einer Schnittstellen-Veränderung

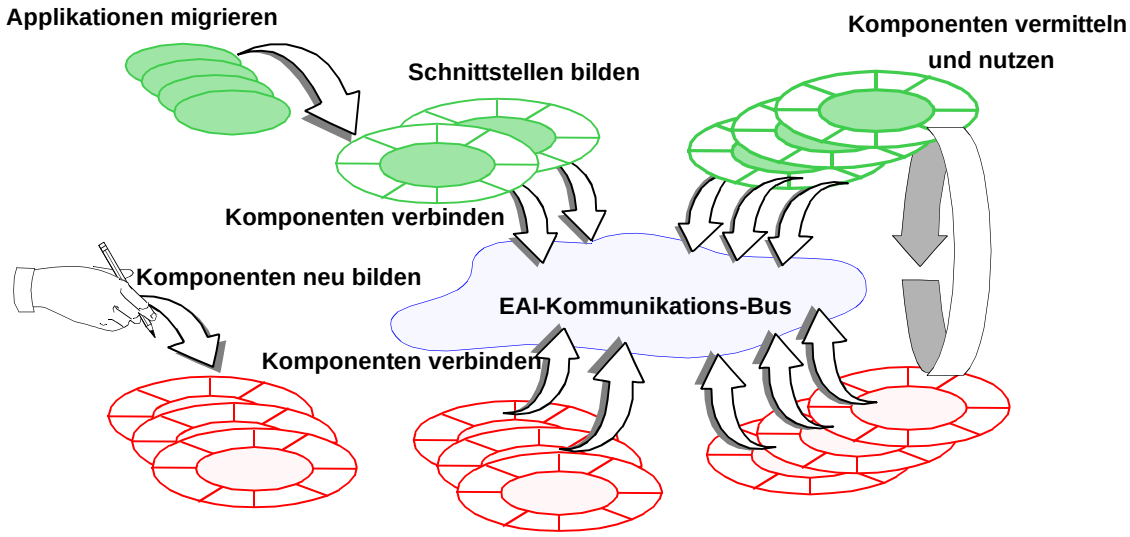
- Client und Server müssen in der Regel beide mit der neuen IDL-Definitionen neu erstellt (generiert, konstruiert, “built”) werden (Build-Time Effect)
- Entweder Client oder Server muss nicht neu erstellt werden (Runtime-Effect)

Regel:

> Wenn immer Sie eine ausgebreitete Schnittstellen-Definition ändern, denken Sie an die Möglichkeit, dass irgendwo “da draussen” ein (oder viele) Client-Applikation mit einer älteren Version existiert und operativ ist.

Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

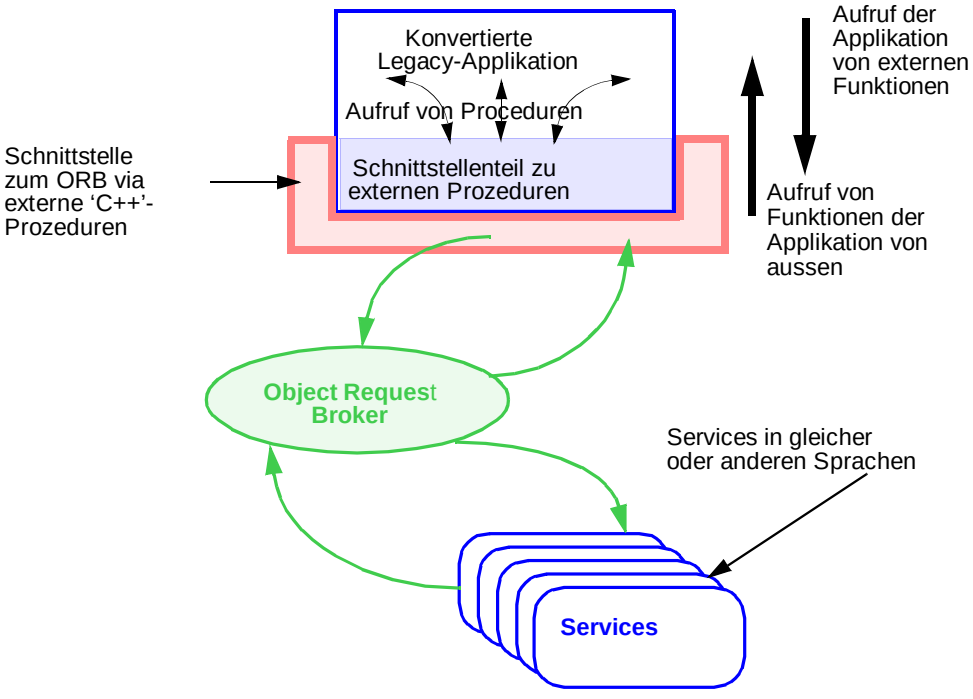
Praxis: Applikations-Integration



- Wrapping ist sowohl für Legacy-Migration wie auch für EAI ein bewährtes Prinzip.
- Ideen, Kreativität und Pragmatik ist hier gefragt.

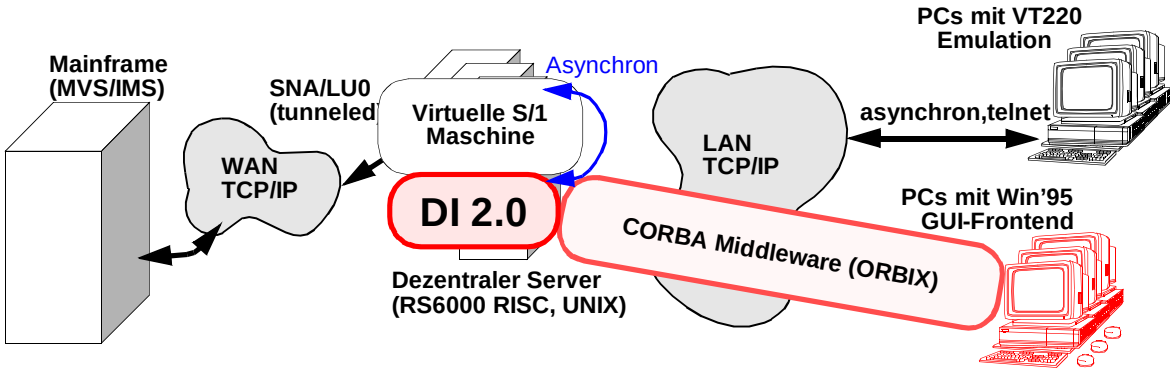
Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

Praxis: Wrapping



Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

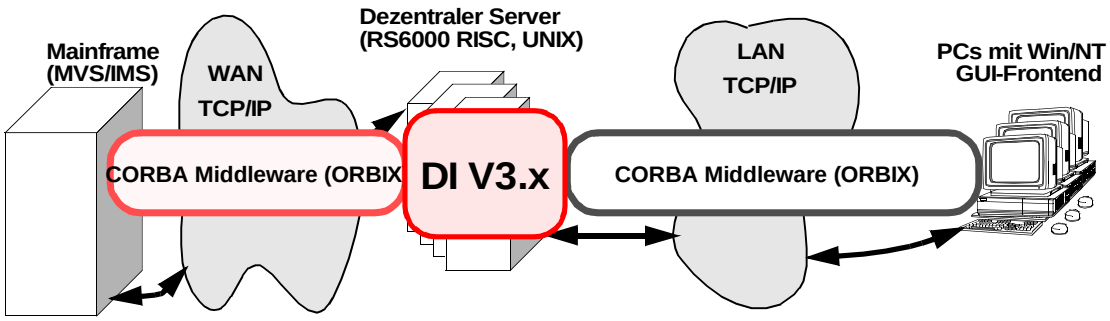
Praxis: EAI als Teil von Legacy-Ablösung



- Terminalsubsysteme wurde durch PCs mit der üblichen graphischen Oberfläche abgelöst.
- Die Hostapplikation wurde durch eine CORBA-konforme Schnittstelle für beliebige Applikationen zugänglich gemacht.
- Terminals verschwinden nach der Einführungsphase des GUI

Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

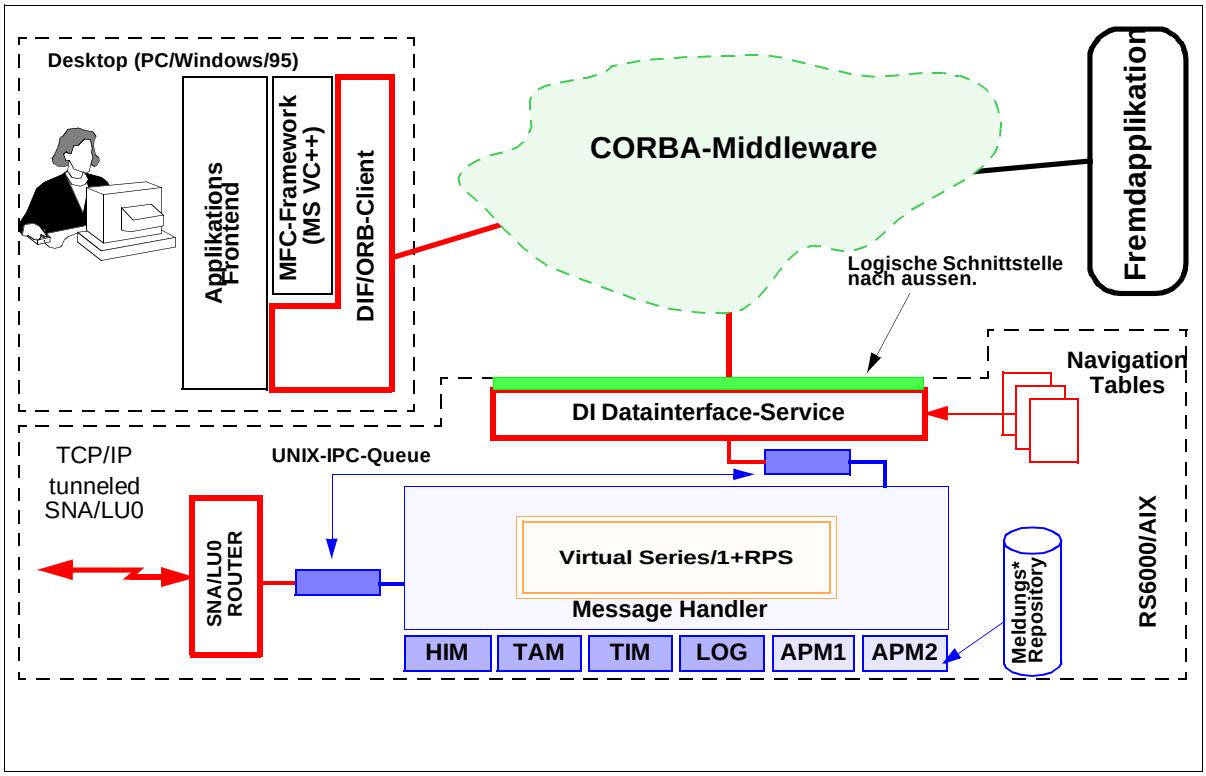
Praxis: EAI als Teil von Legacy-Ablösung



- Schnittstelle zu den Client-PCs identisch mit vorangehender Phase
- CORBA als Middleware bis hinauf zum Mainframe mit hoher Performanceverbesserung
- Assembler- und PL/1-Programme auf Zwischenserver eliminiert
- Host-Transaktionsprogramme bleiben weitgehend unverändert.

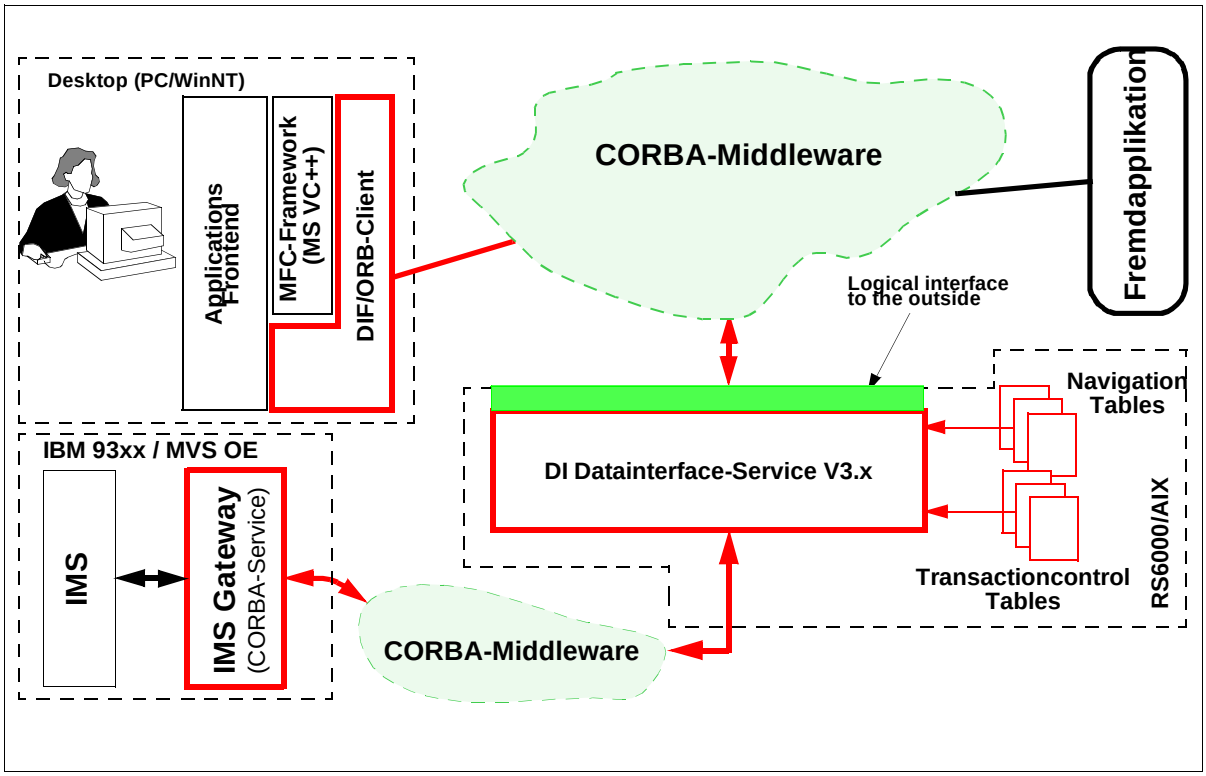
Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

Praxis: EAI als Teil von Legacy-Ablösung



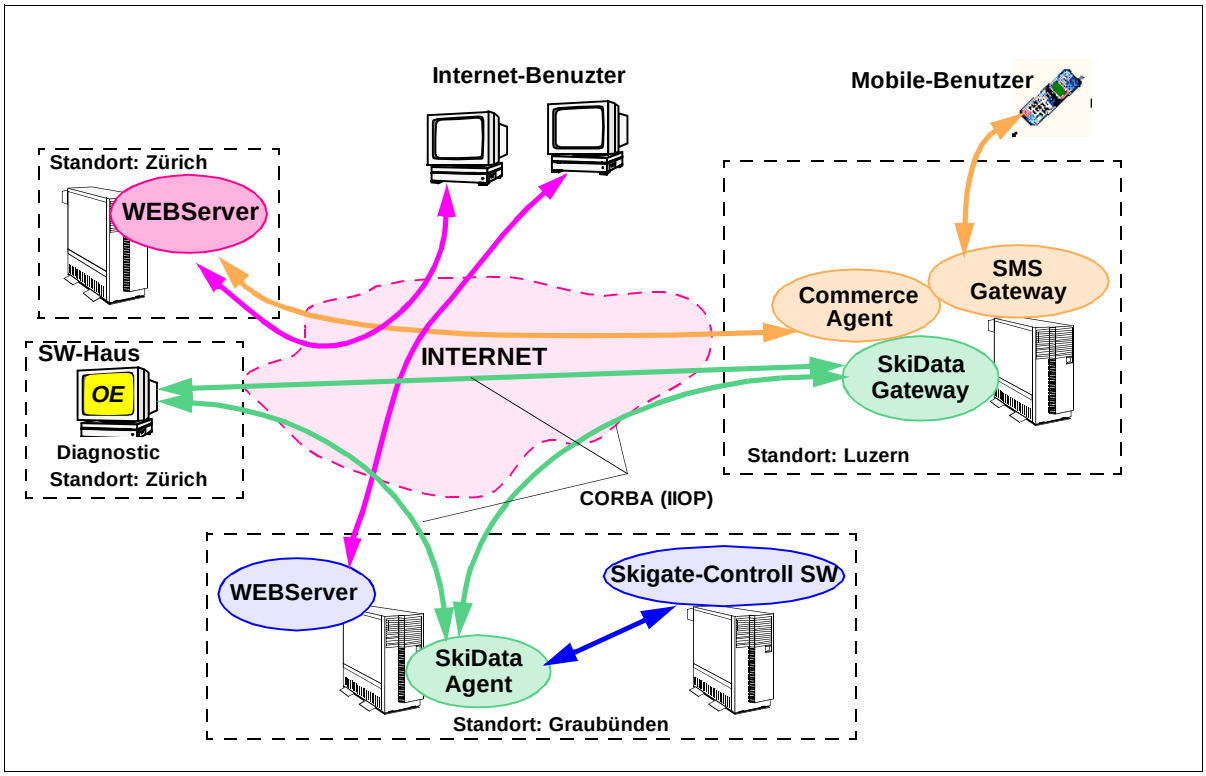
Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

Praxis: EAI als Teil von Legacy-Ablösung



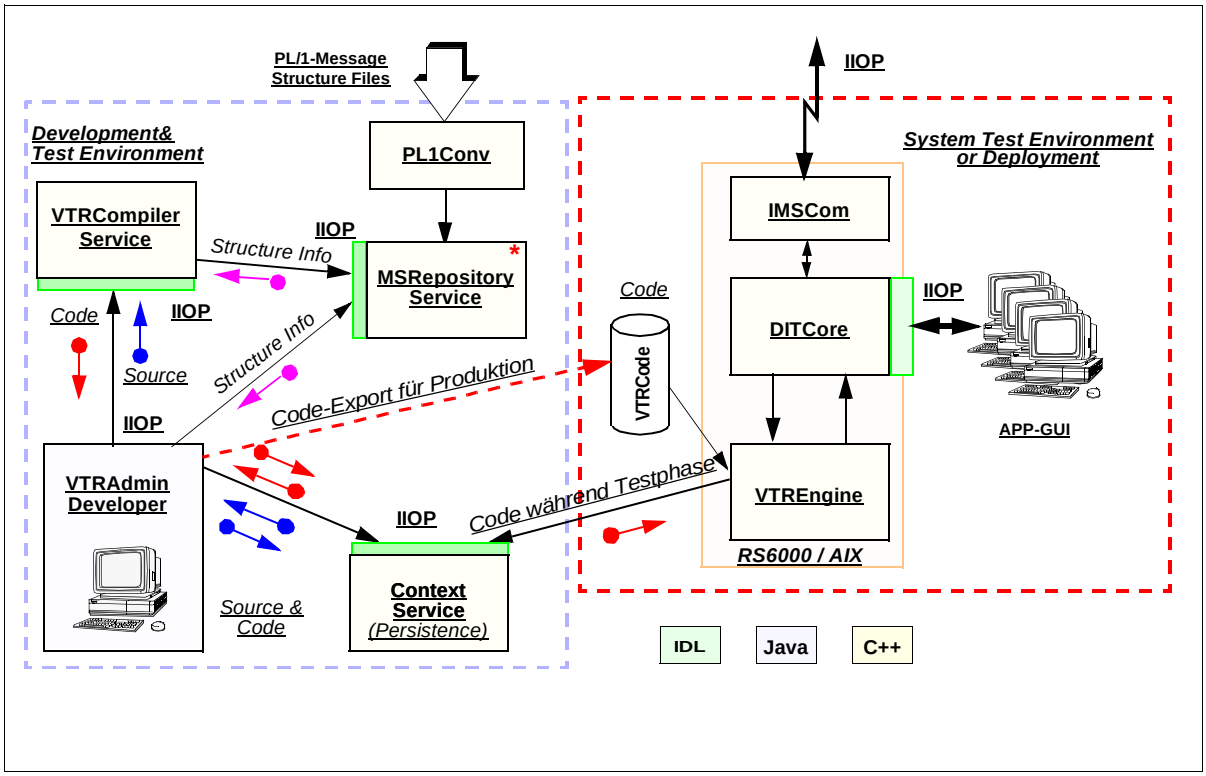
Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

Praxis: B2B über Internet



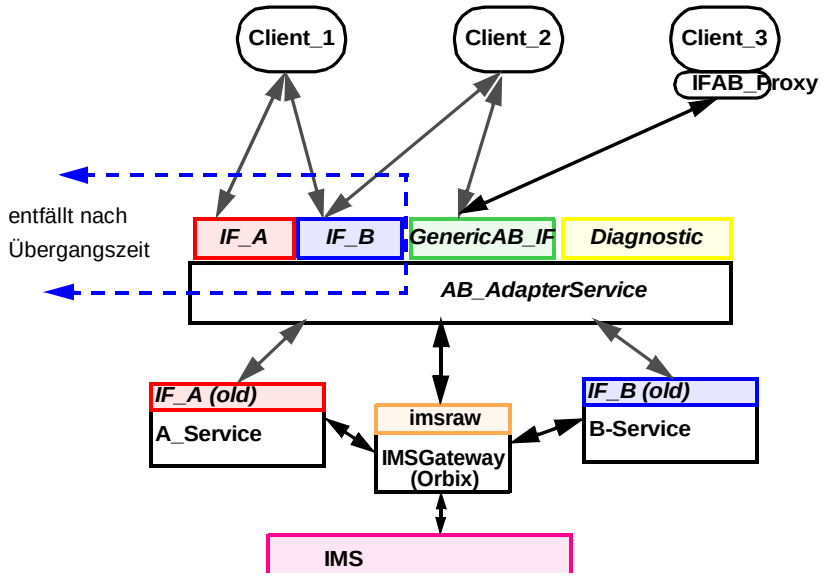
Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

Praxis: EAI-"Extremfall"



Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

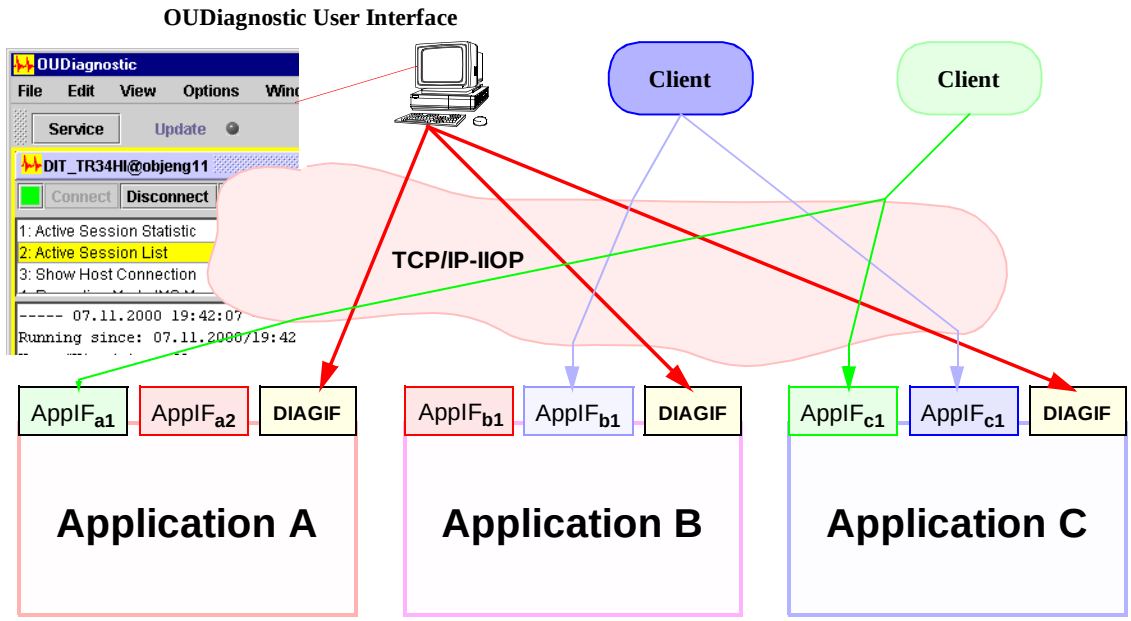
Praxis: Migration von Schnittstellen



- Zusammenführung “alter” Schnittstellen und Migration auf eine generische Schnittstelle (Notaktion zur Bewältigung alter Design-Sünden)

Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

Praxis: Diagnose zur Laufzeit



- Jede Applikation hat identische Diagnostik-Schnittstelle.

Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

Zusammenfassung

- Schnittstellen sind in der Realisation (ausprogrammiert) um Faktoren komplexer, als es die Schnittstellen-Definition erkennen lässt.
- Schnittstellen-Design muss durch die Architektur vorgegeben sein.
- Auch wenn CORBA-IDL-Sprache sehr umfangreich ist, muss man trotzdem nicht alle Möglichkeiten davon nützen (Weniger ist oft mehr).
- Performance wiedersetzt sich in der Regel Flexibilität
- Nachträgliches Ändern von Schnittstellen, kann zu “broken Code” führen.
- Aufteilen und “Vererben” von Schnittstellen bringt Design-Flexibilität.
- Generische Schnittstellen erfordern teils aufwendige Interpretationen auf Client wie Service-Seite bieten aber Flexibilität.
- XML kann wesentlich zur Flexibilität beitragen š *Message Paradigma*
- Die IDeaLe Schnittstelle existiert nicht, kann aber angestrebt werden.
- Diagnose zur Laufzeit ist ein sehr wichtiger Aspekt für den Betrieb von verteilten Applikationen.

Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

Schlussfolgerung

- EAI hat in erster Linie mit IT-Architektur und Organisation und erst an zweiter Stelle mit Technologie zu tun.
- Architektur aus einer Hand erweist sich als vorteilhaft
- Gutes Teamwork und gute Kommunikation auf menschlicher Ebene ist für den Erfolg von EAI-Projekten unabdingbar.
- Technologie ersetzt auch bei EAI weder den guten Projektleiter noch den kreativen Software Ingenieur.
- EAI braucht etwas Mut, Kreativität und Pragmatik, ohne dabei Systematik und Professionalismus zu vergessen.
- EAI und IT-Architektur ist wie Qualitätssicherung ein fortwährender Prozess.

Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland

Schnittstellen: Das A&O für Enterprise Application Integration

Fragen?

Andres Koch
dipl. El. Ing. HTL / M. Math
Object Engineering GmbH,
Uitikon-Waldegg, Schweiz
Email: akoch@objeng.ch
www.objeng.ch

Copyright © 2002 by Object Engineering GmbH, 8142 Uitikon, Switzerland