

CORBA als Ordnung in verteilten Anwendungen

«Java Programming with CORBA» von Brose, ...
«Essentials CORBA-Buch» von Andreas Sayegh
und div. «Internetquellen»

Anton Böhm

anton.boehm@itServe.ch

itServe AG

P.O.Box

Länggass-Str.26

CH-3000 Bern 9

Tel. +41 31 305 16 16

Client / Server

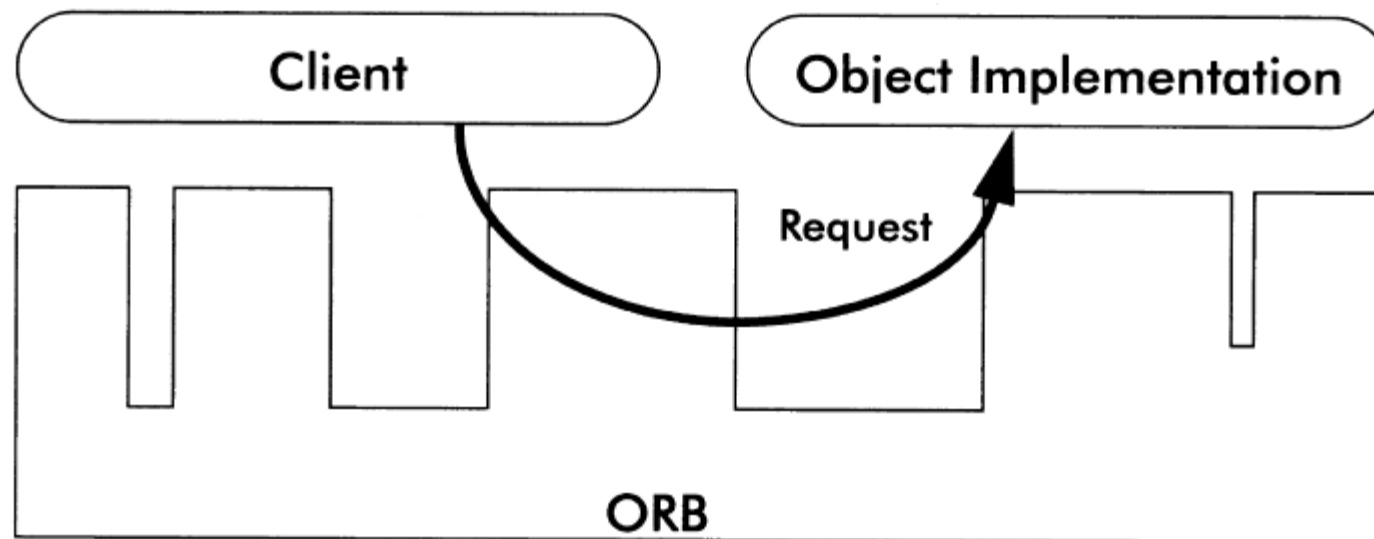


Abbildung 2-1: Objektorientierte Client/Server-Kommunikation.
Quelle: CORBA-Standard

Verteilte objektorientierte Anwendung

keine revolutionäre Technologie
sondern
eine «Spezifikation» zur Ordnung der
«intergalaktischen» Objekt-Welt
Homogenes Client/Server Konzept
Integration anderer Objekt-Modelle
(z.B. DCOM Distributed Component Object Model
oder J2EE / EJB)

Verteilte objektorientierte Anwendung (2)

CORBA 1 : Verteilte Objekte
durch ORB

CORBA 2 : Interoperabilität der ORB's
durch Netzwerkprotokoll

Interoperabilität schafft

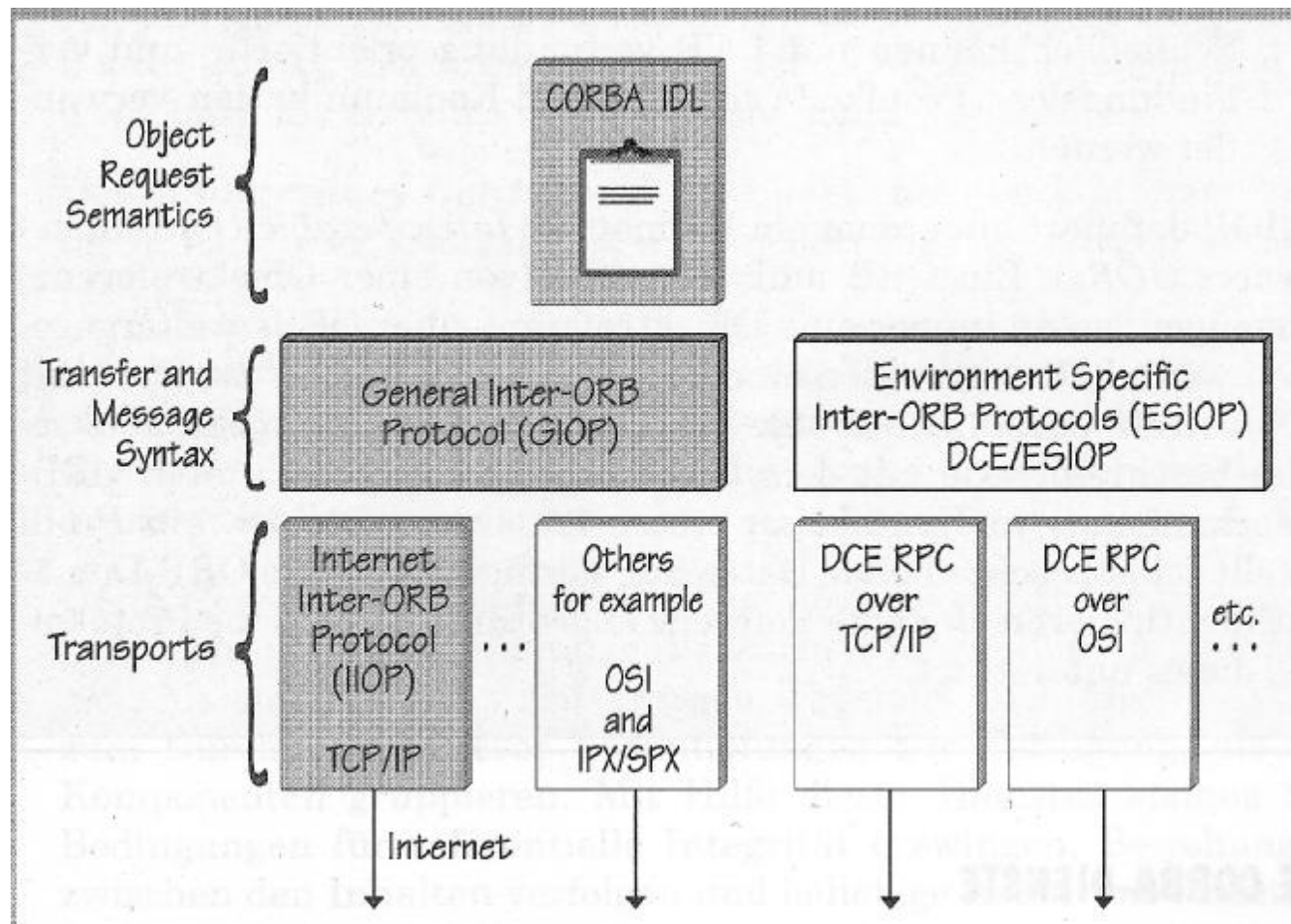
– Konkurrenz



– Markt



Inter-ORB-Architektur



Entfernte Objekte

Differenz der «lokalen» und «entfernten» Objekte

- *Ort und «life cycle»* (spez. Folie)
 - lokal: innerhalb eines Prozesses (Adressraumes) /CPU
 - entfernt: nicht innerhalb eines Prozesses /CPU
- *Zugriff*
 - lokal durch Memory-Pointer
 - entfernt durch Kommunikationsprotokolle
- *Protokoll - Operationen*
(im single Task/Thread-Bereich)
 - lokal: sequentielles Verhalten
 - entfernt: nicht sequentielles Verhalten

Client/Server Modell

Client «operiert» auf Server

Anfrage (Request) senden

und Antwort (Reply) empfangen

Operation wird als «Dienst» (Service) erbracht

Entfernter Zugriff soll für die Anwendung

«transparent» erfolgen

Lösung:

Lokaler Zugriff auf ein entferntes
Objekt

Entfernte Objekte und Proxies

Zu jeder Implementierung eines entfernten Objekts muss es eine korrespondierende Implementierung eines lokalen **Proxy**-Objekts geben.

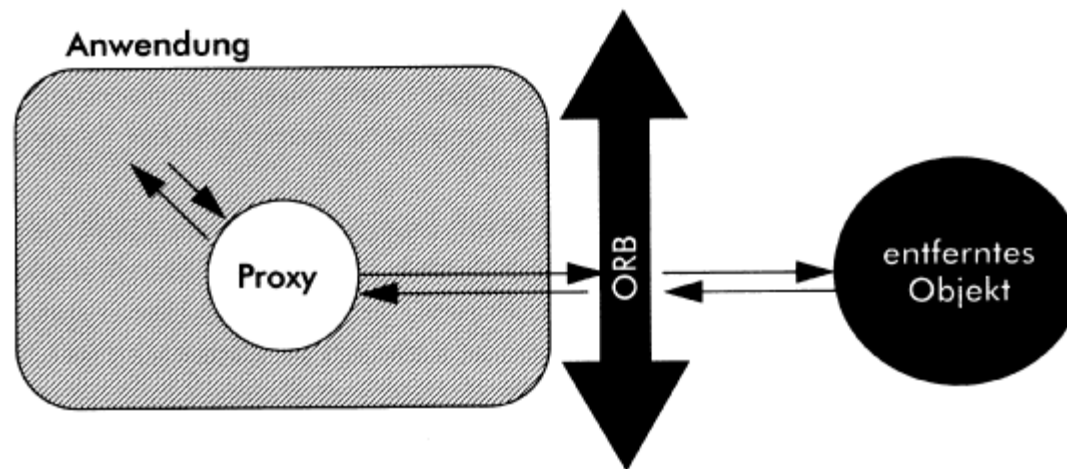
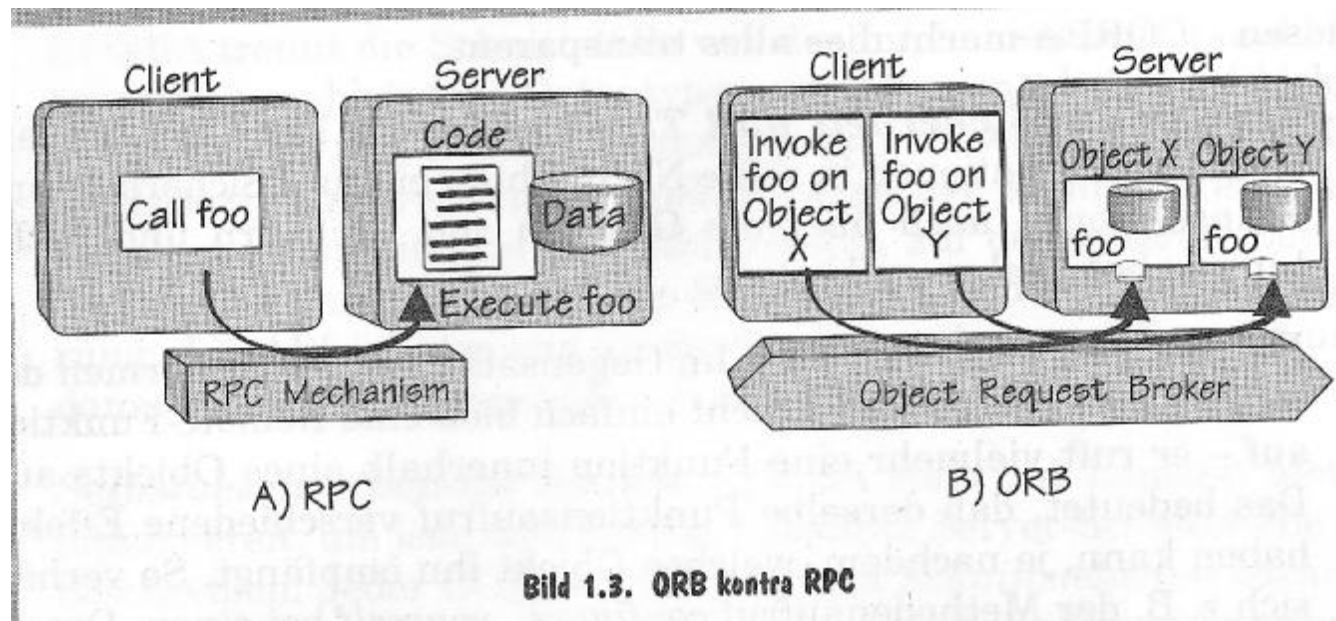


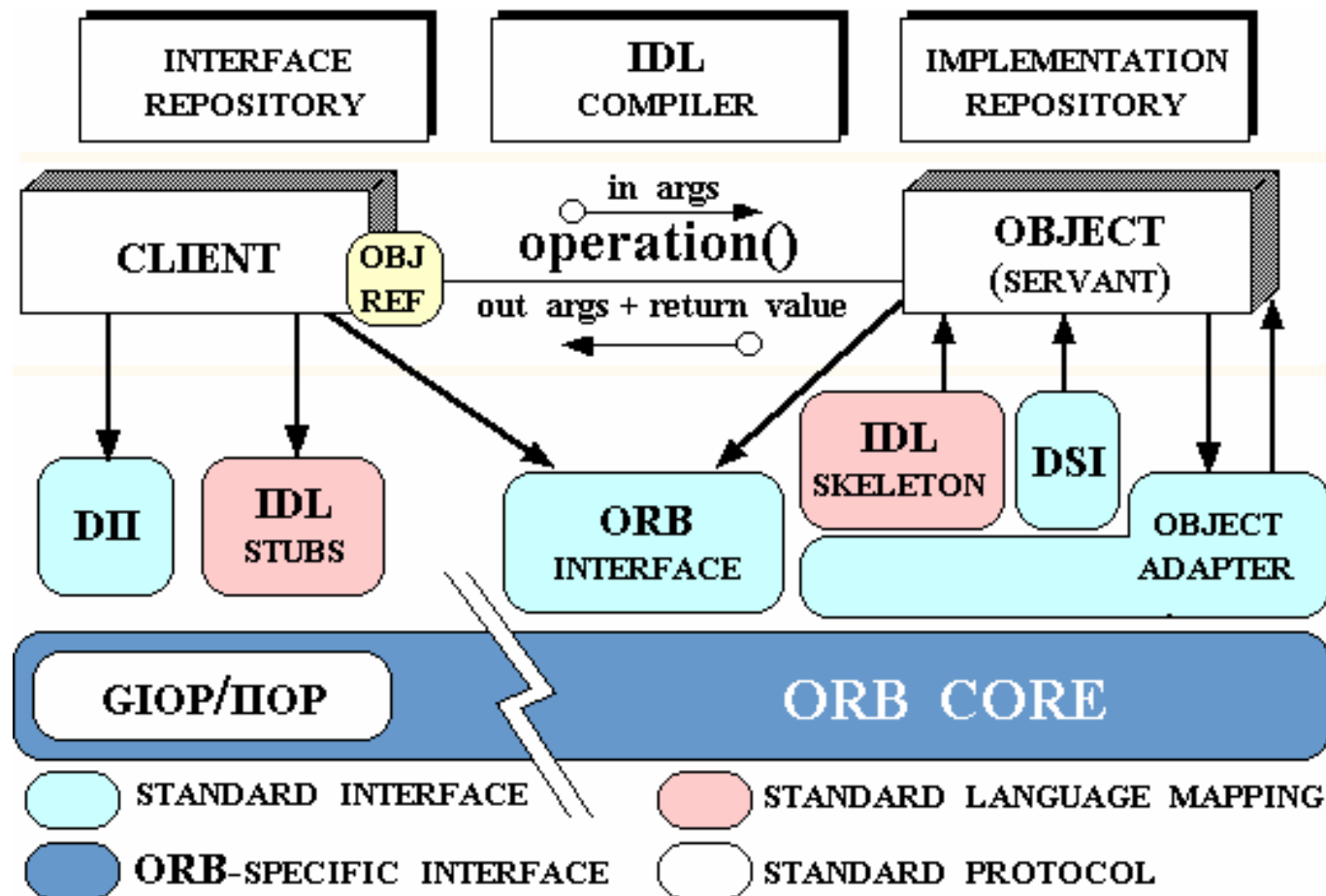
Abbildung 2-2: Schema der Proxy-Kommunikation

ORB kontra RPC

Funktionsaufruf \Leftrightarrow polymorphe Methode
(verschiedengestaltig)



ORB Architektur-Skizze



<http://www.cs.wustl.edu/~schmidt/corba-overview.html>

Attribute

Ein gelungenes objektorientiertes Konzept sollte nicht den direkten Zugriff auf Attribute anderer Objekte vorsehen

Automatische Generierung

- Selektoren für lesenden Zugriff
- Modifikatoren für schreibenden Zugriff

Lebenszyklen

Die Erzeugung eines entfernten Objekts ist, gemäss CORBA, nur mit einem «Life-Cycle-Service» möglich.

Protokoll mit folgenden Operationen

- Erzeugen
- Kopieren
- Verschieben (Migration in andere CORBA-Domäne)
- Löschen

Proprietärer «Teil-Ersatz» mit `_bind()`

Anhang

Architekturen

WWW

Java

Client/Server (3-Tiered)

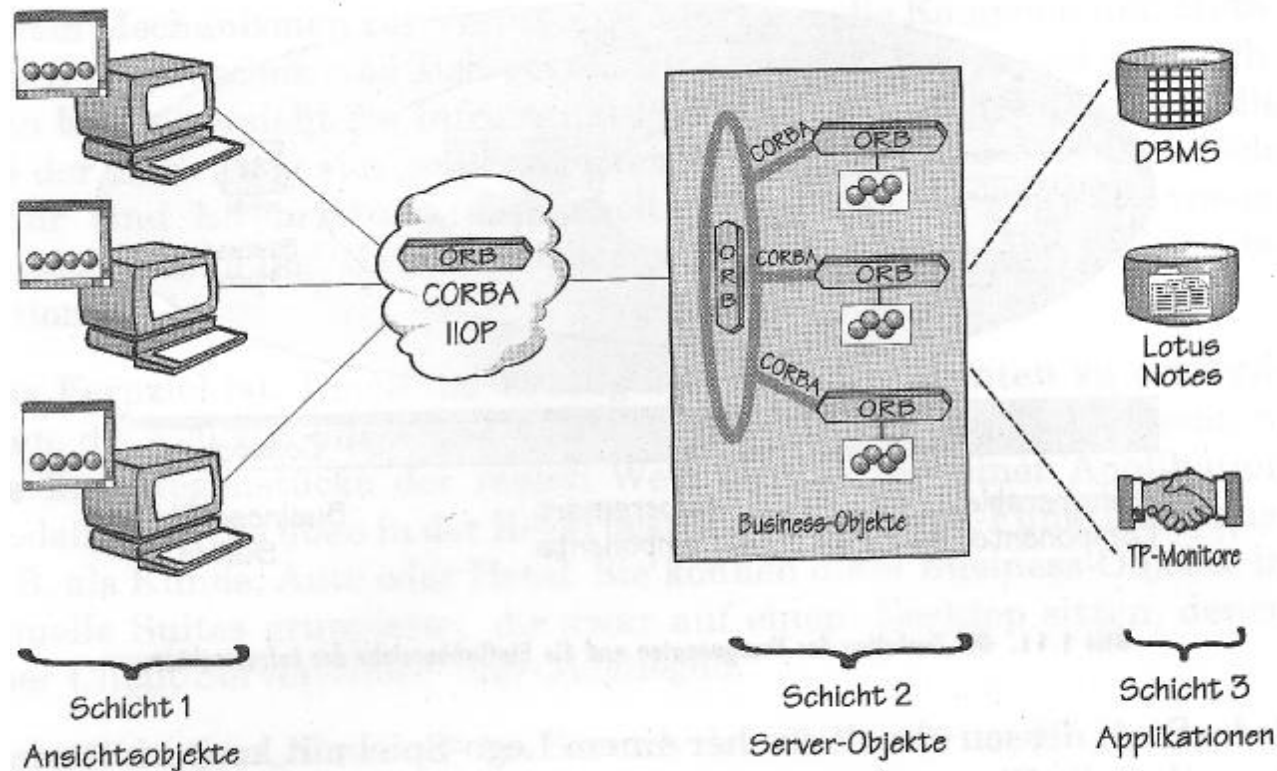


Bild 1.12. Dreischichtige Client/Server-Verarbeitung mit Objekten

WWW

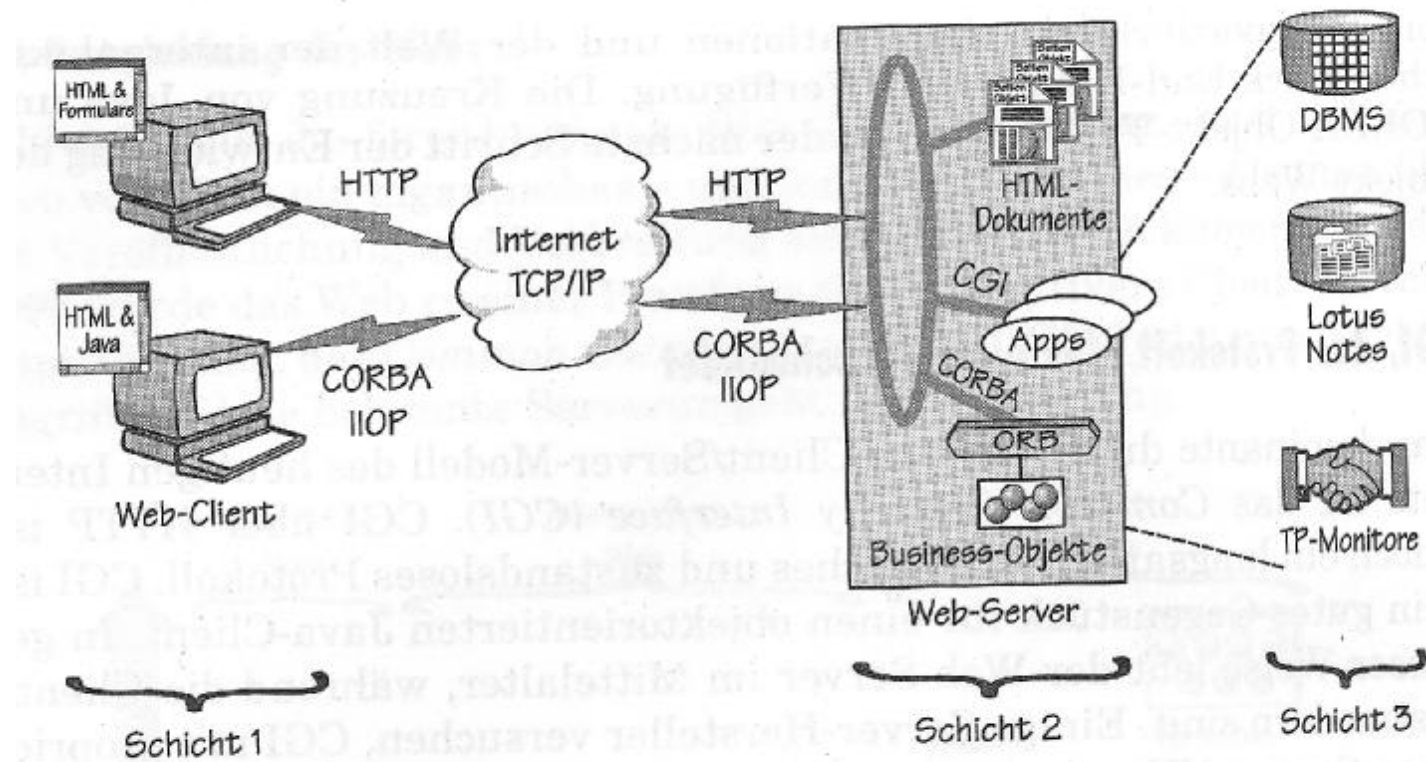


Bild 2.2. Ein Objekt-Web-Modell, das auf Java-Clients und CORBA-ORBs basiert

WWW und Java

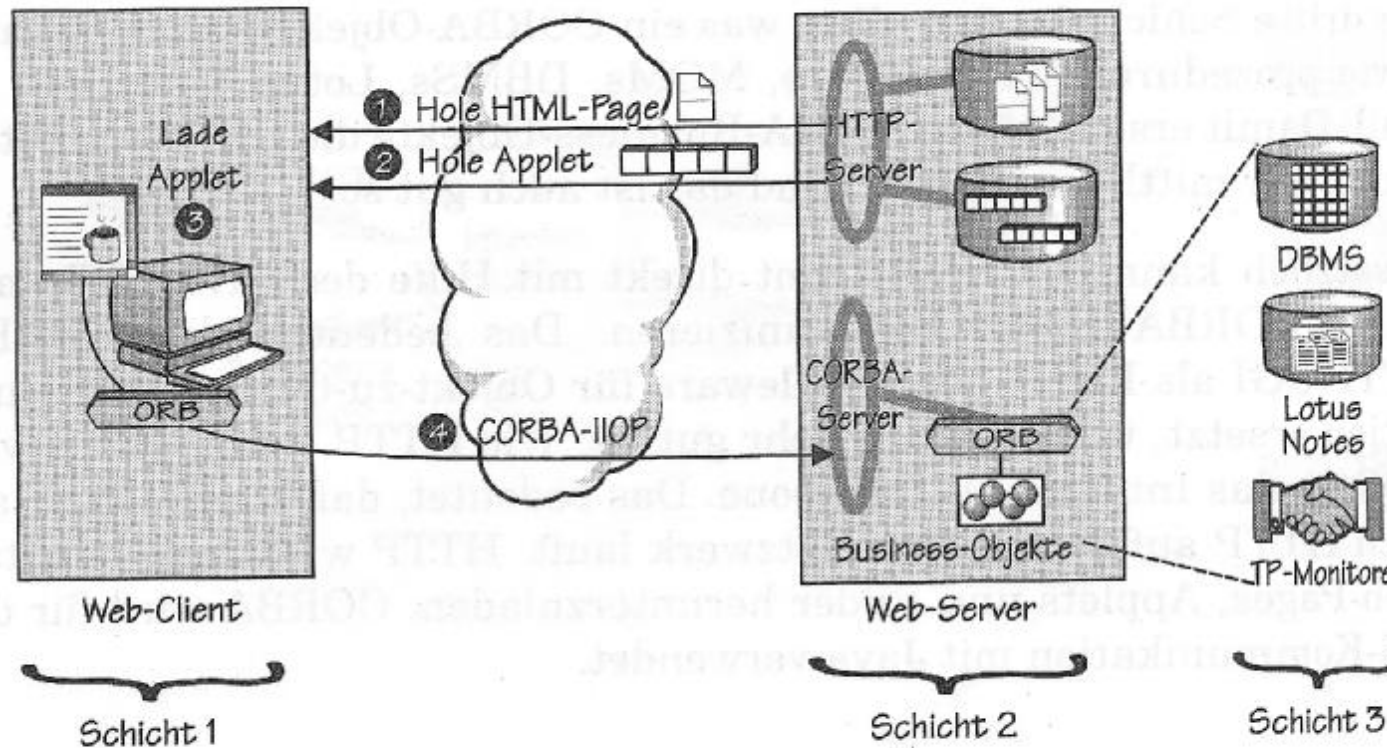


Bild 2.3 Wie HTTP, Corba und Java zusammenarbeiten

Vergleich

Tabelle 2.2: Vergleich von Java/CORBA ORBs und ihrer Konkurrenz

Funktion	CORBA/IIOP	DCOM	RMI	HTTP/CGI	Sockets
Abstraktionsebene	★★★★	★★★★	★★★★	★★	★
Nahtlose Java-Integration	★★★★	★★★	★★★★	★★	★★
OS-Plattform-Unterstützung	★★★★	★★	★★★★	★★★★	★★★★
All-Java Implementierung	★★★★	★	★★★★	★★★★	★★★★
Typed parameter-Unterstützung	★★★★	★★★★	★★★★	★	★
Einfache Konfiguration	★★★	0	★★★	★★★	★★★
Verteilte Methodenaufrufe	★★★★	★★★	★★★	0	0
Aufrufe zwischen Zuständen	★★★★	★★★	★★★	0	★★
Dynamische Entdeckung u. Metadaten-Unterstützung	★★★★	★★★	0	0	0
Dynamische Aufrufe	★★★★	★★★★	★	0	0
Übertragungsleistung (Remote Ping)	★★★★ 3,3 ms	★★★ 3,9 ms	★★★ 5,5 ms (extrapoliert)	0 603,8 ms	★★★★ 2,0 ms
Sicherheit auf Kabelebene	★★★★	★★★★	★★★	★★★	★★★
Transaktionen auf Kabelebene	★★★★	★★★	0	0	0
Langlebige Objektreferenz	★★★★	★	0	0	0
Benennung auf URL-Basis	★★★	★	★★	★★★★	★★★
Objektaufrufe in verschiedenen Sprachen	★★★★	★★★	0	★★★	★★★★
Sprachneutrales Kabelprotokoll	★★★★	★★★★	0	★★★★	0
Intergalaktische Skalierung	★★★★	★	★	★★	★★★★
Offene Standards	★★★★	★★	★★	★★★★	★★★★