Vorlesung
Software aus Komponenten

3. Komponentenmodelle

Prof. Dr. Hans-Gert Gräbe
Wintersemester 2014/15

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Objekte und Methoden

- RPC ist ein statisches Aufrufkonzept
- Besonderheit von Methoden gegenüber Prozeduren:
  - werden dynamisch an Hand der Charakteristika des Objekts (=Instanz seiner Klasse) ausgewählt
    - erst nach dieser Auswahl kann der RPC-Mechanismus greifen
    - Klassen müssen dazu genügend (binär kodierte) Information bieten, die durch Introspektion zur Laufzeit abgefragt werden kann
  - Objektreferenzen als Aufrufparameter
    - Keine automatischen Objektkopien

Der RPC-Ansatz ist deutlich aufzubohren, wenn mit Objekten und Methoden mit laufzeitabhängigem Verhalten umgegangen werden soll.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Weitere Fragen, die ein objektorientiertes Kommunikationskonzept beantworten muss

1. Wie werden Schnittstellen spezifiziert? Hier interessiert vor allem die programmiersprachen-unabhängige technisch-funktionale Spezifikation der Aufrufcharakteristika
2. Wie werden Dienste aufgefunden und bereitgestellt?
3. Wie wird die Evolution von Komponenten gehandhabt? (Versionsmanagement)

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Die OMG und CORBA

Geschichte, Zielstellungen, Entwicklungsetappen

Architektur

- Objekte, Servanten, Anwendungen
- Schnittstellensprache OMG IDL
- Dynamische Methodenaufrufe (DII)
- Symmetrie des CORBA-Modells
- Der Object Request Broker (ORB)
- CORBA-Objekte und Objektreferenzen
- CORBA IDL und Datentypen

Literatur: CORBA Spezifikation 3.0 (Juni 2002),

- 1154 Seiten pdf-Dokument,
  siehe http://www.omg.org/spec/CORBA/index.htm

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Zur Geschichte der OMG (Object Management Group)

- **Ausgangspunkt 1989:** Wie kommuniziert man in einem verteilten OO-System über Sprach- und Plattformgrenzen hinweg?
  - selbst auf derselben Plattform lieferten C++-Compiler inkompatiblen Bytecode, verschiedene Objektmodelle in verschiedenen Programmiersprachen, Plattformunterschiede bei Socket-Kopplung
  - „Deep gaps everywhere“
- im April 1989 von 11 Firmen gegründet
- heute mit ca. 800 Mitgliedern eines der größten Konsortien der Computer-Industrie
  - vor allem Systemanbieter und Anwender objektorientierter Techniken

Zielstellung: „Standardisierung, koste es, was es wolle“, um Interoperabilität auf allen Ebenen in einem offenen Markt für „Objekte“ zu erreichen.

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Zielstellungen der OMG

- Offene Interoperabilität zwischen einer Vielzahl von Sprachen, Implementierungen und Plattformen
- mehr standardisieren als „binäre“ Standards
- Flexibilität statt Binärkompatibilität
  - „teure“ Hochsprachenprotokolle
- **Nichtkommerzielle Vereinigung** zur Entwicklung von technisch exakten und in der Praxis realisierbaren Spezifikationen
- **Vereinbarung von Standards und Spezifikationen** der Infrastruktur für verteilte, objektorientierte Anwendungen
- **Aufstellung von Richtlinien** (guidelines) zur Entwicklung von Umgebungen, in denen heterogene Systemen (verschiedene Plattformen, Betriebssysteme u.ä.) zusammenarbeiten können
- Durch standardisierte, objektorientierte Softwarekonzepte die **Entstehung eines Marktes für Komponentensoftware forcieren**

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Etappen der Entwicklung von CORBA

CORBA 1 (seit 1991) : **Standardisierung des ORB**

- erste Lösungen, um das Wirrwarr zu entflechten
- Ansatz: Vermittlung zwischen Anfragen und Diensten durch einen Object Request Broker (ORB)
- CORBA = Common Object Request Broker Architecture
- **Meilenstein**: Schnittstellen-Definitionssprache (OMG IDL)

CORBA 2 (seit 1995–96) : **Interoperationsstandards zwischen ORBs**

- **Meilenstein**: Internet Inter-ORB Protokoll (IIOP)
- muss von jeder ORB-Implementierung unterstützt werden
- Für CORBA 2 existiert Vielzahl von Realisierungen verschiedener Anbieter und für verschiedene Plattformen

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Etappen der Entwicklung von CORBA (Fortsetzung)

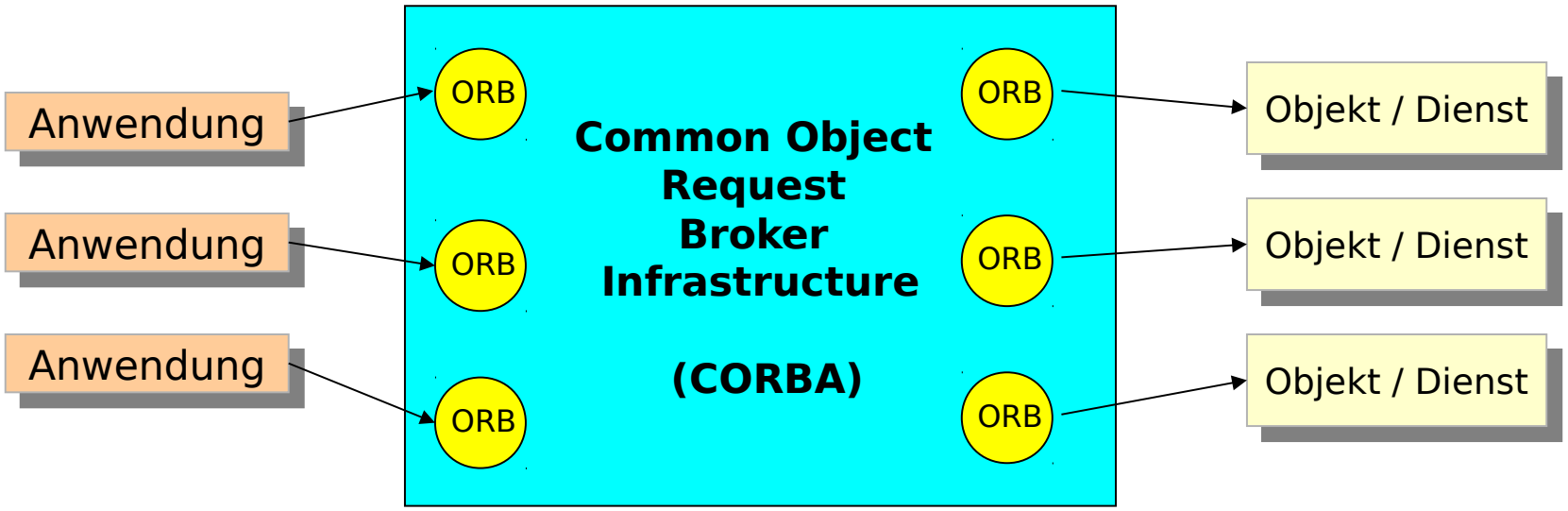
CORBA 3 (12/2002) : **Komponenten- und Systemintegration**

- Höhere Abstraktionsebene
- neue Sprachebenen zur Beschreibung von Komponenten-Eigenschaften
- seit 1998 in der Entwicklung, aber als Ganzes erst Ende 2002 freigegeben
  - CORBA 2.3 ... 2.6 (2001) : Freigabe verschiedener Standards, auf die man sich auf dem Weg zu CORBA 3 zwischenzeitlich geeinigt hatte
- **Meilenstein**: CORBA Komponentenmodel (CCM)
  - Version 4.0, April 2006
- aktuelle Version CORBA 3.3, November 2012 (http://www.omg.org/spec/CORBA)

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

CORBA besteht im Grunde aus drei wichtigen Teilen:

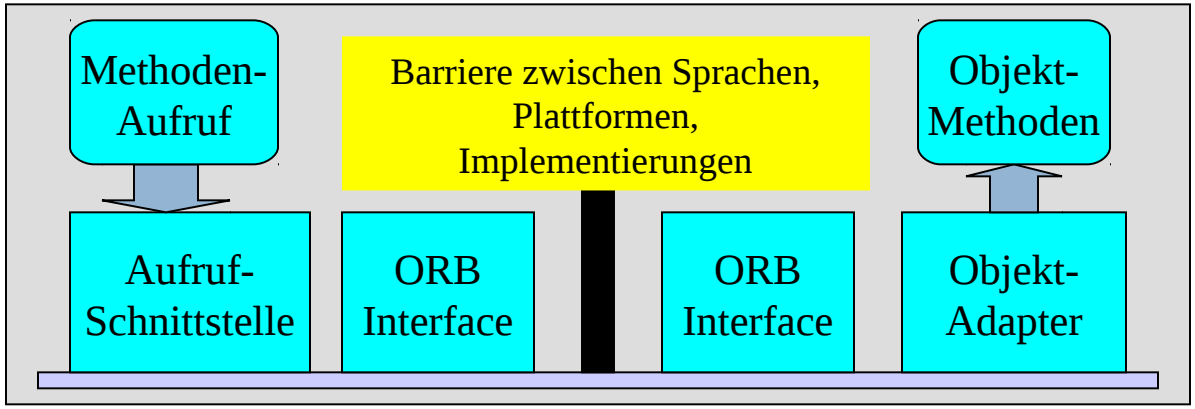
- einer Menge von Aufrufschnittstellen (Invocation Interfaces)
- den Vermittlern (Object Request Brokers – ORBs) als den Schaltstellen der Kommunikation
  - mit einem spezifizierten Protokoll, dem Internet Inter-ORB Protocol IIOP
- einer Menge von Objekt-Adaptern



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Laufzeitbindung von Methodenaufrufen

- Aufrufschnittstelle serialisiert Aufrufargumente
- ORBs suchen Zielobjekt, -methode, organisieren Transport der Argumente
- Objektadapter: dient der Aktivierung des Diensts im Objekt. Deserialisiert außerdem Argumente und ruft entsprechende Methode des Zielobjekts auf (Skeleton-Funktion).



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Wichtige Voraussetzungen

- Schnittstellen müssen in einer einheitlichen Sprache **definiert** werden (**Interface Definition Language - OMG IDL**)
  - wesentlicher Bestandteil des CORBA-Standards
  - ermöglicht generisches Serialisieren / Deserialisieren

```
module Example {
  struct Date {
            unsigned short Day;
            unsigned short Month;
            unsigned short Year; }

interface Ufo {
            readonly attribute unsigned long ID;
            readonly attribute string Name;
            readonly attribute Date FirstContact;
            unsigned long Contacts ();
            void RegisterContact (Date dateOfContact); }
}
```

Datentyp-Definition

Attribute der Schnittstelle

Methoden der Schnittstelle

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Wichtige Voraussetzungen (Fortsetzung)

alle Programmiersprachen, die den CORBA-Standard unterstützen, müssen **an OMG IDL gebunden** werden.

- Mapping von Datentypen
- Übersetzung des OMG IDL Operationsformats in das sprachspezifische Aufruf-Format
- Fehlerbehandlung
- existieren Anbindungen für C, C++, SmallTalk, Cobol, Java, ...

In OMG IDL beschriebene Schnittstellen werden dann

- mit einem **OMG IDL Compiler** übersetzt
- im **Interface Repository** abgelegt
- durch **Methoden der ORB-Schnittstelle** gefunden

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

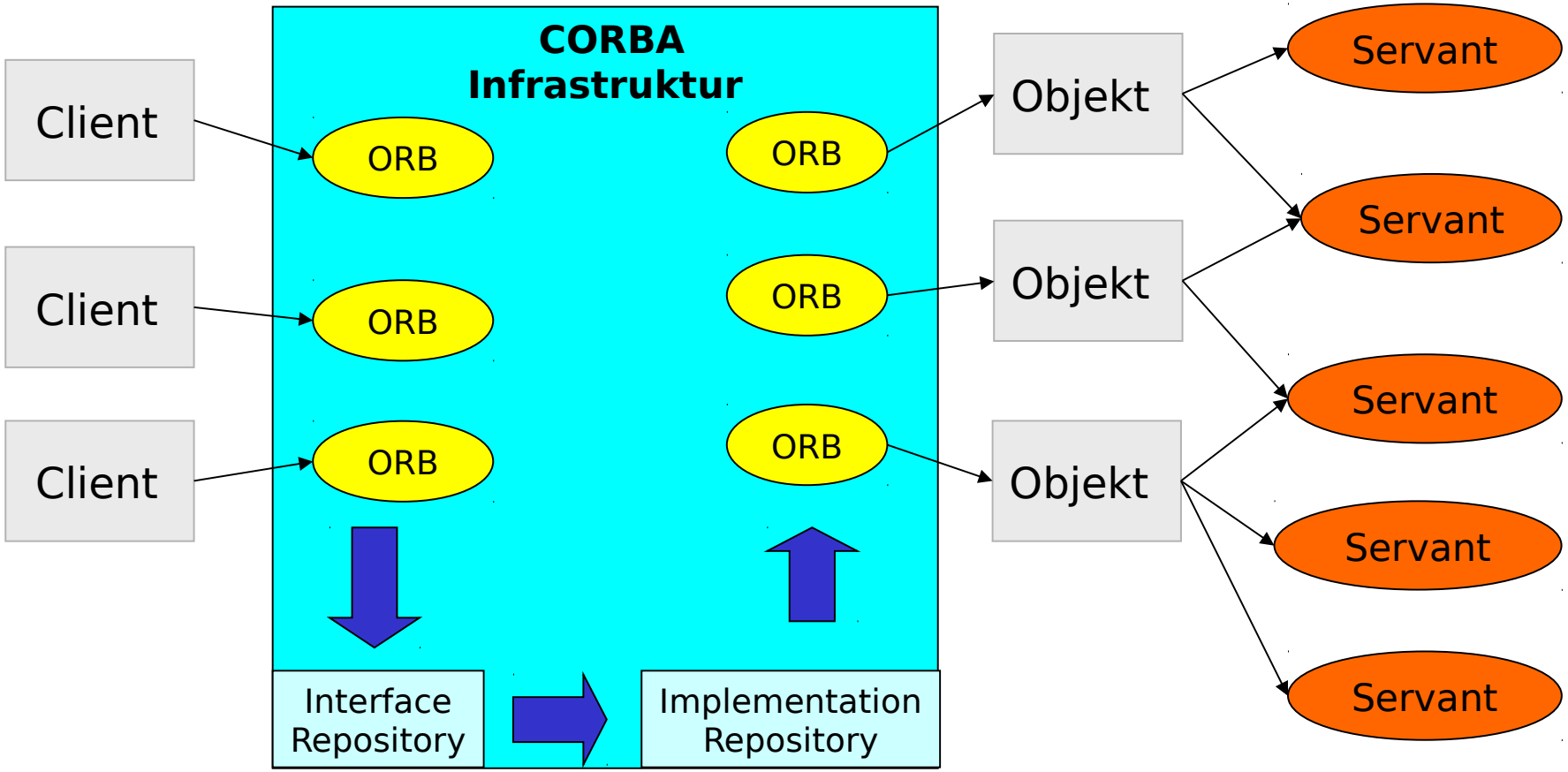
Wichtige Voraussetzungen (Fortsetzung)

Programmfragmente stellen **Implementierungen** für solche Schnittstellen (oder Teile davon) bereit

- Heißen **Servanten** (object servant)
- Werden über ihren Objekt-Adapter im **Implementation Repository** registriert und von diesem bei Bedarf geladen und/oder gestartet
- Objektadapter teilen dem ORB mit, welche (leichtgewichtigen) Objekte von welchen Servanten bedient werden.
- Eine Serverumgebung (typ. Prozess) kann mehrere Servanten bedienen.
- *:* - Beziehung zwischen Objekten und Servanten
- Objektbegriff hat damit leicht anderen Fokus als in der Vorlesung

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Architektur im Überblick



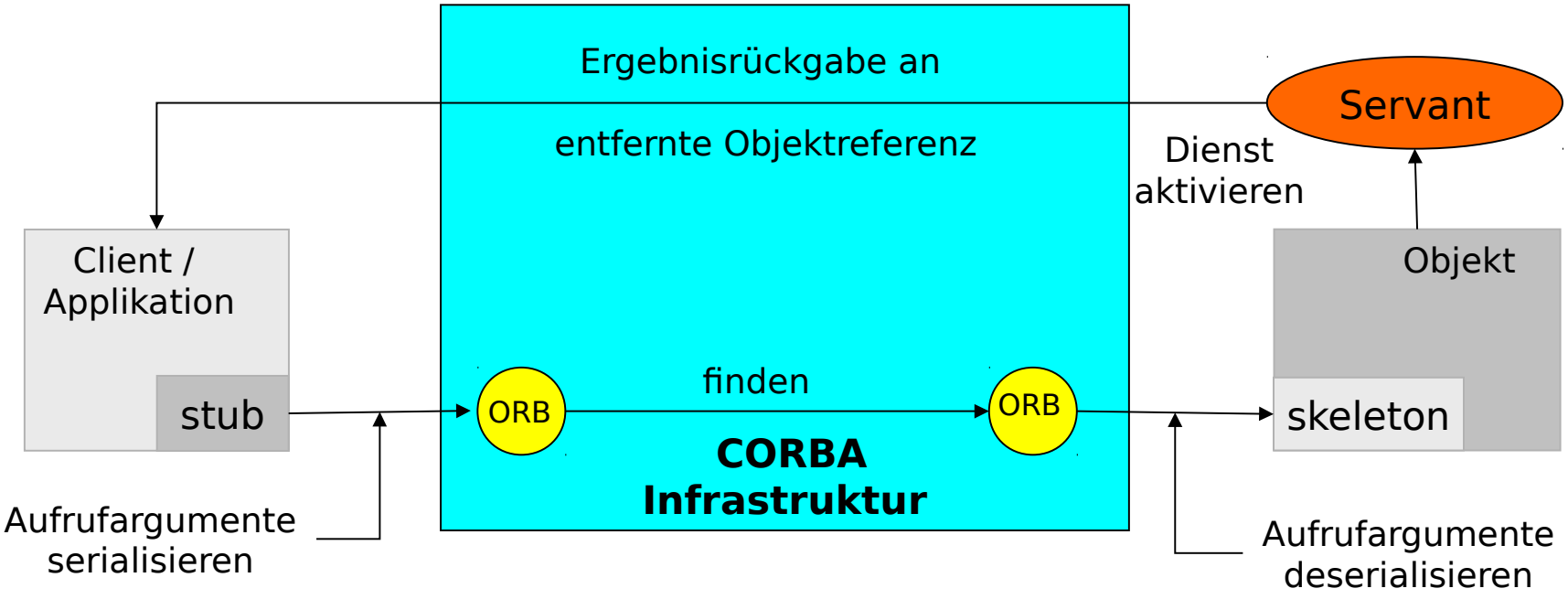
UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Stummel (stubs) und Skelette (skeletons)

Methodenaufrufe erfolgen über Stummel-Skelett-Prinzip des RPC

- mit OMG IDL Compiler aus Schnittstellenbeschreibung generierbar

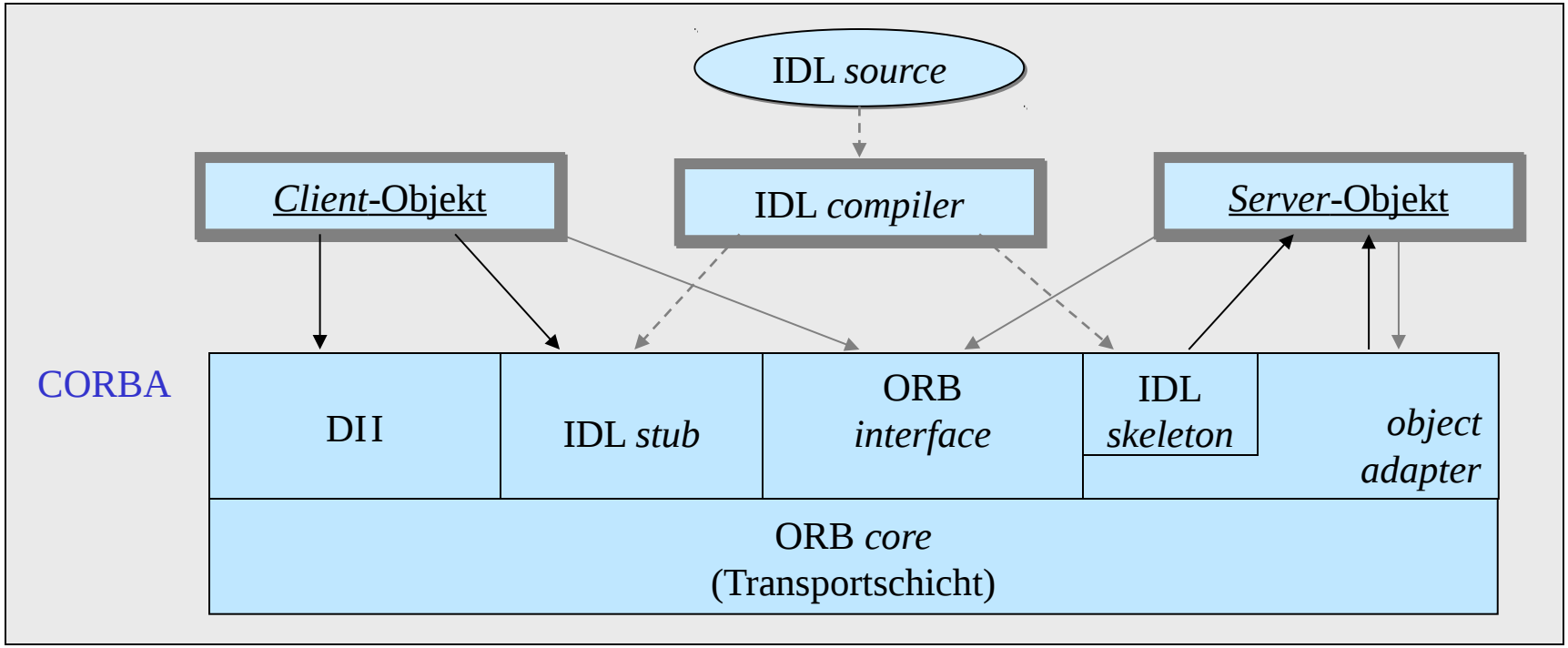
Direkt nur für statische Methodenaufrufe einsetzbar (static invocation interface SII / static skeleton interface SSI)



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Dynamische Methodenaufrufe (dynamic invocation interface - DII)

- Erforderlich, um Methoden zur Laufzeit binden zu können
- Seit CORBA 2.0 auch Dynamik auf der Serverseite (dynamic skeleton interface – DSI)
- Verwenden eine **universelle Datenstruktur für Argumente**, um Methoden mit (statisch) unbestimmter Signatur zu behandeln
- Aus Geschwindigkeits-Gründen wird SII / SSI zusätzlich bereitgestellt.



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Symmetrie des CORBA-Modells

Keine Asymmetrie wie im Client-Server-Modell: Jeder Prozess kann sowohl Methodenaufrufe absetzen als auch empfangen.

Einzige Asymmetrie kommt durch den **Objektadapter.**

- Programme, die als Servant eingesetzt werden, müssen sich beim ORB durch einen Objektadapter registrieren
- erster Standard: basic object adapter (BOA), deprecated seit 1998
  - war unterspezifiziert, deshalb Wildwuchs von Erweiterungen
- heute: portable object adapter (POA)
  - aktueller Standard ist Teil von CORBA 3.0.3, März 2004
- Mit der Registrierung „weiß“ der Objektadapter (und nur dieser), wie der Servant aktiviert wird
- jedes Objekt hat eine „Hausmaschine“, auch wenn ein Servant über mehrere Maschinen „verfügen“ kann
- Reine Applikationen, die keine Dienste zur Verfügung stellen, sondern nur welche nutzen, werden auch nicht registriert. Können damit auch nicht durch CORBA gestartet werden.

Aufgaben des ORB

- Schlüsselkomponente und Kommunikationszentrale der Architektur
- vermittelt Methoden-Aufrufe zwischen Applikationen und Servanten
  - arbeitet nur mit den Schnittstellen (stub, skeleton)
  - Schnittstellen-Definition über OMG IDL

Verwendet dazu Informationen aus dem **Schnittstellen-Repository** (interface repository - IR)

CORBA Begriffe: **Dienste** werden über **CORBA-Objekte** angesprochen, deren Methoden mit den entsprechenden **Servanten** verbunden sind.

ORB verantwortlich für Suche von CORBA-Objekten, über deren Methoden der jeweilige Dienst erbracht wird

- Verwendet dazu Informationen aus dem **Repository der Implementierungen** (implementation repository)

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Aufgaben des ORB (Fortsetzung)

- Zusammenspiel von ORB, Objektadapter (Dienst) und Stummel (Nutzer) beim Auflösen dynamischer Methodenbindungen
  - ORB liefert relevante Interfacedefinitionen aus dem IR
  - Stummel löst dynamische Bindung von Aufrufen auf (DII)
  - Objektadapter realisiert Auflösung von Objektreferenzen
- Verwaltung der CORBA-Objekte
  - Aktivierung und Deaktivierung
  - Policy-Operationen
  - Verwaltung zugeordneter leichtgewichtiger Prozesse (Threads)

- **Unterscheide zwischen ORB als Klasse (Konzept der Architektur) und ORB-Instanzen (Laufzeitobjekte)**
  - Aufbau und Organisation einer Implementierung des ORB als Klasse ist von Anbieter und Einsatzgebiet abhängig.
  - Kann als einzelner Prozess oder verteilte Anwendung implementiert sein

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

CORBA–Objekte

CORBA-Objekte sind Programmfragmente mit Eigenleben, charakterisiert durch

- Objektzustand (aktuelle Werte der Attribute)
- Funktionalität (verfügbare Methoden)

Dienste eines CORBA-Objekts können von Applikationen nur über den ORB in Anspruch genommen werden.

ORB organisiert Aktivierung und Deaktivierung von CORBA-Objekten und Diensten

- Anforderung entsprechender Ressourcen (CPU, Speicher)
- Sicherung der persistenten Bestandteile bei Deaktivierung
- Aktivierungs- und Deaktivierungsmuster können durch entsprechende Regeln (policies) festgelegt sein

CORBA-Objekte sind damit Zwischending zwischen Komponenten und Objekten im Sinne der Vorlesung

Jedes CORBA-Objekt hat einen Typnamen (= Klasse in Java)

- Typname entspricht dem Schnittstellennamen in der IDL Deklaration
- Typname steht für einen abstrakten Datentyp als Menge von Methoden und deren Signaturen sowie Variablen (Attributen) und deren Typen

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

CORBA Objektreferenzen

Statt Objekten werden normalerweise nur Referenzen übergeben

- Seit CORBA 2.3 können Objekte auch als Wertparameter übergeben werden
- Verwendet eine **standardisierte Serialisierung**

Referenz über Programmgrenzen statt Referenz innerhalb eines Programms

- Kann Objektänderungen durch die Evolution des Programmstatus im Ursprungsprogramm nicht verfolgen
- Damit teurer in der Handhabung als physische Referenzen und eher vergleichbar mit einer URL
  - Seit CORBA 2.3 existiert Standard zur Darstellung von Objektreferenzen als URL
- ORB Schnittstelle enthält **Methoden zum Umwandeln** zwischen physischen Referenzen und CORBA Objektreferenzen
- Problem des Call Backs – physische und CORBA Objektreferenz können im lokalen Kontext koexistieren.

Lebensdauer per Definition **unbestimmt**

- Wiederverwendung einer Referenz kann einen Fehler auslösen
- referenziertes Objekt muss nicht mehr existieren

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

OMG IDL und Datentypen

OMG IDL unterscheidet primitive Datentypen und CORBA Objektreferenzen

- Basistypen (integer, float, char, string)
- zusammengesetzte Datentypen
  - Strukturen, Sequenzen, Aufzählungstypen
  - multi-dimensionale Felder fester Größe

Parameter eines primitiven Datentyps werden als Wertparameter übergeben

Umfang der Unterstützung ist von eingesetzter Programmiersprache abhängig

- CORBA Standard: Aufruf eines nicht unterstützten Typs erzeugt einen Fehler zur Übersetzungszeit
- Damit kann Sprache verwendet werden, die nur einen Teil des Standards implementiert, wenn auch nur dieser Teil verwendet wird.