# Vorlesung
# Software aus Komponenten

## 3. Komponentenmodelle

Prof. Dr. Hans-Gert Gräbe
Wintersemester 2015/16

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Entwicklung einer CORBA–Anwendung unter JAVA
(aus "Lehrbuch der Softwaretechnik" von Helmut Balzert)

1) Spezifikation der Schnittstelle in CORBA–IDL
2) Übersetzung mittels IDL–Compiler
   - Stummel- und Skelettklassen werden erzeugt
3) Implementierung der Operationen der Schnittstelle
4) Rahmenanwendung entwickeln
   - Erzeugt Objekt der Klasse
   - Objekt wird für Clients zugreifbar gemacht
5) Entwickeln des Clients

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

# Definition der Schnittstelle mit OMG IDL

```
module SemOrg { // Schnittstelle der Klasse Firma
    interface Firma {
        attribute string Name;
        attribute float Umsatz;
        // Operationssignatur
        float berechneGewinn( in float Kosten );
    };
};
```

SemOrg.idl

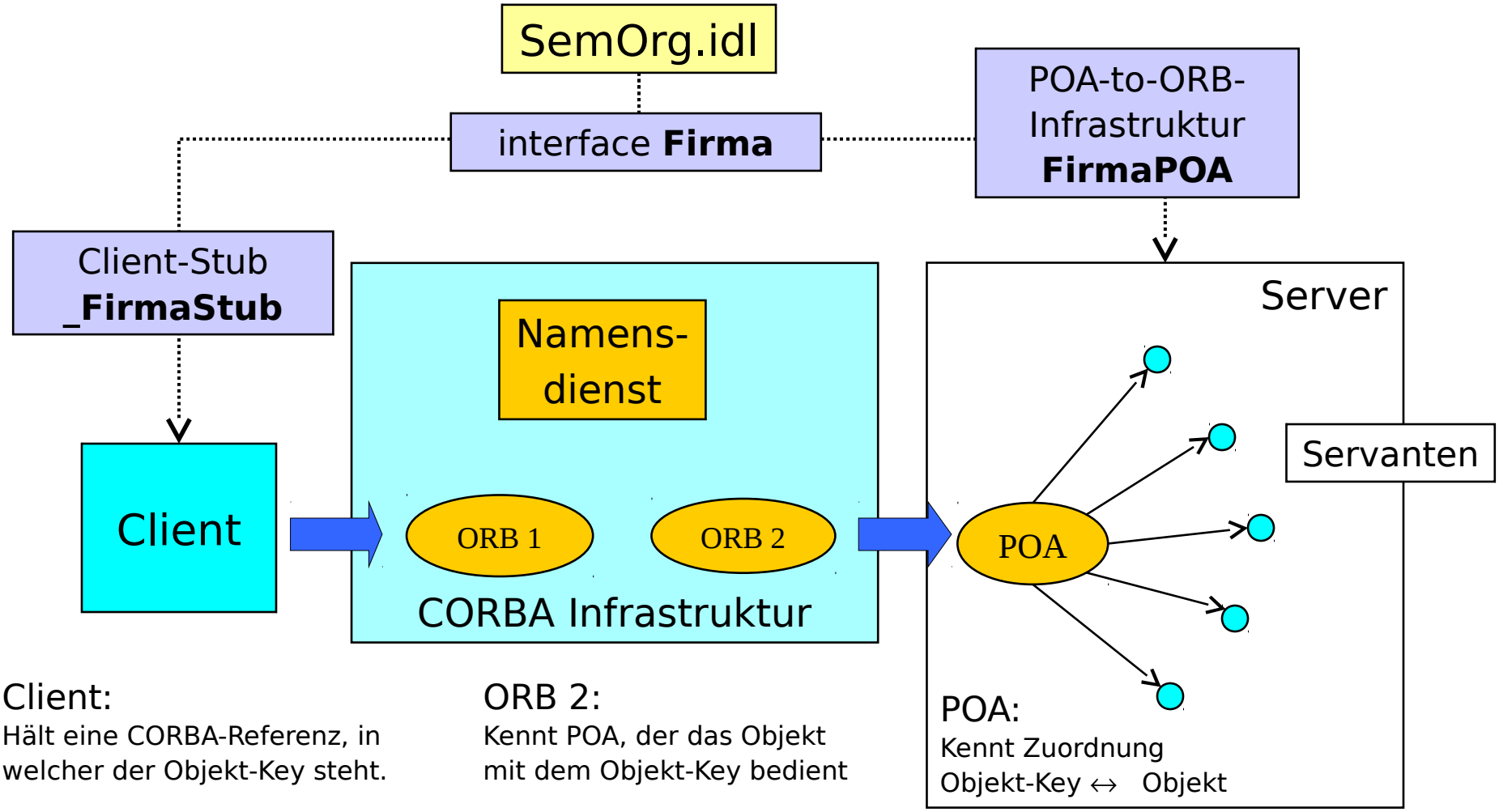
**‘idlj –f all SemOrg.idl‘** erzeugt daraus Java-Package **SemOrg**

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

## Vom IDL–Compiler erzeugte Klassen

- interface **FirmaOperations**
  - interface Firma als Java-Interface
- interface **Firma** extends SemOrg.FirmaOperations, ...
  - Schnittstelle mit speziellen Firma-Operationen und generischen CORBA.Object-Operationen
- class **_FirmaStub** - voll generierte Stummel–Klasse für den Client
- final class **FirmaHolder** implements CORBA.portable.Streamable
- class **FirmaHelper**
  - Statische Methoden, um CORBA.Any auf Semorg.Firma zu casten
- abstract class **FirmaPOA** extends PortableServer.Servant implements SemOrg.FirmaOperations ....
  - portabler Objekt-Adapter
  - generierte Implementierung der ORB-seitigen Kommunikation zur Implementierung der Fachlogik in einer von FirmaPOA abgeleiteten Klasse (auf diese Methoden wird „abstract“ Bezug genommen)

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme


**Client:**
Hält eine CORBA-Referenz, in welcher der Objekt-Key steht.

**ORB 2:**
Kennt POA, der das Objekt mit dem Objekt-Key bedient

**POA:**
Kennt Zuordnung Objekt-Key ↔ Objekt

Dienst nutzen ⇐ Dienst finden ⇐ Dienst bekannt machen ⇐ Dienst als Subinstanz des POA generieren

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

## Implementierung des Servanten

```
package SemOrg;
public class FirmaImpl extends FirmaPOA {
    private String derName;
    private float derUmsatz;
    public FirmaImpl() {} // Konstruktor
    public float berechneGewinn(float Kosten)
        { return this.derUmsatz – Kosten; }
    // get-/set-Operation für Attribut Name
    public String Name() { return this.derName }
    public void Name(String neuerName)
        { this.derName = neuerName; }
    //analog für Umsatz ...
}
```

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

## Default-Server-Modell: Portable Servant Inheritance Model

### Aufgaben des Servers

- Verbindung zur ORB-Struktur herstellen über eigenen Broker
- Instanz des Servanten **FirmaImpl** anlegen und in der CORBA-Struktur verankern: als POA registrieren und den POAManager aktivieren.
- CORBA-Objektreferenz des Servanten besorgen
- Objektreferenz auf einen Namens-Kontext von Namens-Server der ORB-Infrastruktur holen und den Servanten als „eineFirma“ registrieren.
- Auf Anfragen warten

### Aufgaben des Client

- Verbindung zur ORB-Struktur herstellen über eigenen Broker
- CORBA-Objektreferenz auf den Servanten „eineFirma“ über den Namens-Kontext besorgen
- Dienste des Servanten in Anspruch nehmen

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Serverkomponente

```
package SemOrg;
import java.io.*;
import org.omg.CosNaming.*;
import org.omg.PortableServer.*;

public class SemOrgServer {
  public static void main(String[ ] args) {
    try {
      // Kontakt zur ORB-Infrastruktur herstellen: ORB-Referenz holen
       org.omg.CORBA.ORB orb = org.omg.CORBA.ORB.init(args, null);
      // POA-Referenz vom ORB besorgen und POA aktivieren
       POA poa=POAHelper.narrow(orb.resolve_initial_references("RootPOA"));
       poa.the_POAManager().activate();
       // Servanten anlegen als reales Java-Objekt
       FirmaImpl impl = new FirmaImpl();
```

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

```
    // CORBA-Objektreferenz auf den Servanten besorgen und „casten“
    org.omg.CORBA.Object implObjServer = poa.servant_to_reference(impl);
    Firma eineFirmaS = FirmaHelper.narrow(implObjServer);
    // vom ORB Referenz auf Namensdienst-Server besorgen und „casten“
    org.omg.CORBA.Object ncObj =
          orb.resolve_initial_reference("NameService");
   NamingContextExt ncRef = NamingContextExtHelper.narrow(ncObj);
    // Servanten unter dem Namen "eineFirma" im Namensdienst bekannt
    // machen. Verbindung zwischen "eineFirma" und dem Java-Objekt impl
    // kennt nur poa
    ncRef.rebind(ncRef.to_name("eineFirma"), eineFirmaS);
    System.out.println(„SemorgServer gestartet ...“);
  } // end try
  catch (Exception e) { ... }
 } // end main
}
```

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Client-Komponente

```
package SemOrg;
import org.omg.CosNaming.*;

public class SemOrgClient {
  public static void main(String[ ] args) {
    try {
      // Verbindung zur ORB-Infrastruktur
       org.omg.CORBA.ORB orb = org.omg.CORBA.ORB.init(args, null);

      // vom ORB Referenz auf Namensdienst besorgen
      org.omg.CORBA.Object ncObj = orb.resolve_initial_reference("NameService");
      NamingContextExt ncRef = NamingContextExtHelper.narrow(ncObj);
```

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

```
    // CORBA-Objektreferenz auf das Servanten-Objekt „eineFirma“ besorgen
    org.omg.CORBA.Object firmaObj = ncRef.resolve_str("eineFirma");
    Firma eineFirmaC = FirmaHelper.narrow(firmaObj);

    System.out.println(“Handle auf das Server-Objekt”+eineFirmaC);

    //Aufrufe durchführen
    System.out.println(eineFirmaC.Name());
        ...
    }
    catch (Exception e) { ... }
  }
}
```

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

## CORBA in der industriellen Anwendung

- Hauptanwendungsfeld: Ersetzen von Sockets und RPC in Anwendungen, die über mehrere Server verteilt sind
- höhere Abstraktionskonzepte wurden vor CORBA 3 kaum bedient
  - Kooperation beim Entwickeln verteilter Anwendungen über Teamgrenzen bisher kaum unterstützt
- Weitergehende Konzepte hat OMG schon lange im Visier

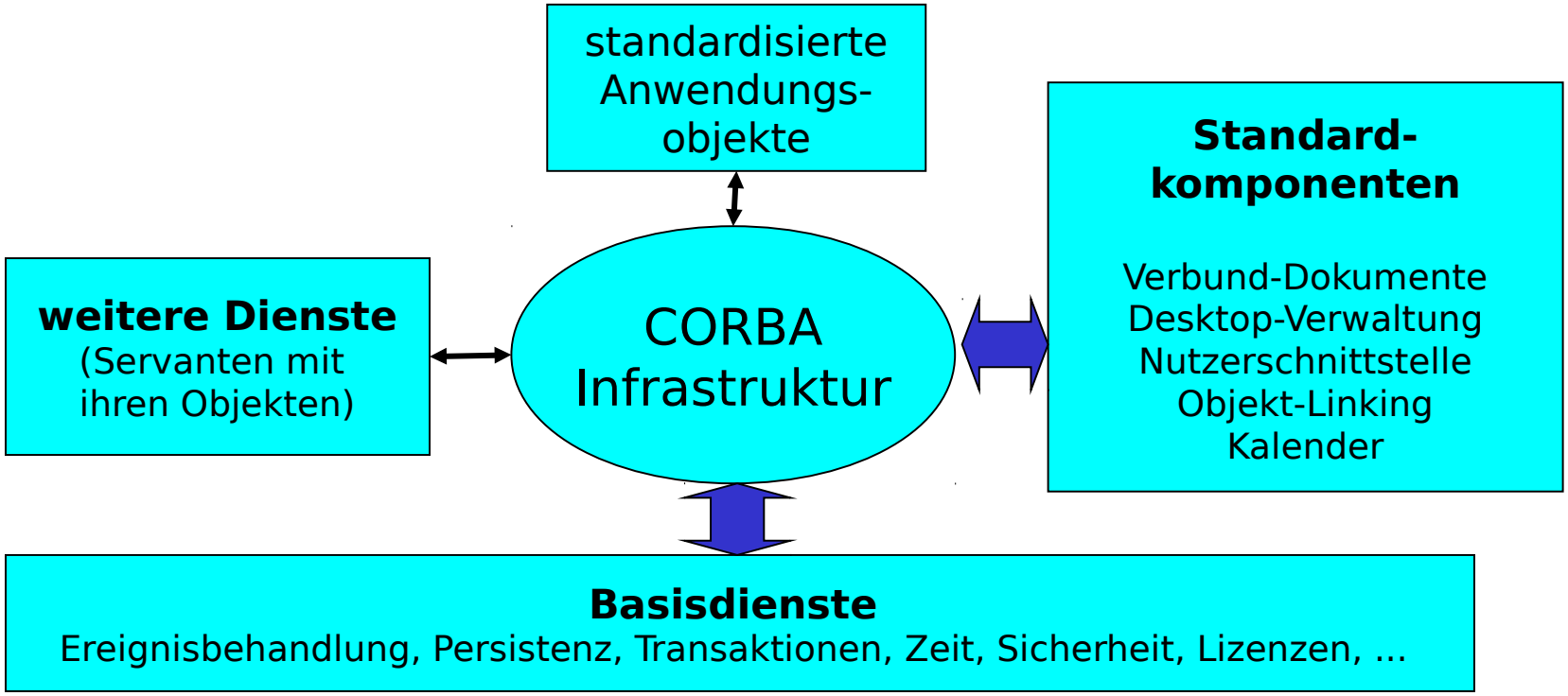
## Object Management Architecture (OMA)

- erste Standardisierungen mit CORBA 2, seit 1997 im Focus der OMG
- mit CORBA 3 ins Zentrum gerückt
- 3 neue Standardisierungsfelder
  - Spezifikation von (grundlegenden) Objektdiensten (CORBAservices)
  - Spezifikation von (häufig benötigten) Anwendungsbestandteilen (CORBAfacilities)
  - Spezifikation von Anwendungsobjekten
- CORBA Komponentenmodell (CCM)

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Idee: Unterteilung der Dienste in mehr oder weniger wichtige

- Bereitstellung von Dienst-Servanten für wichtige Funktionalitäten, die in einen **Komponentenrahmen** (component framework) „eingesteckt“ werden können
  - Bsp: Geschäftsfeld-Objekte (business objects) = Objekte, die direkte Geschäftsprozessabstraktionen repräsentieren
  - steht erst ganz am Anfang der Entwicklung



UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

# Auf dem Weg zu CORBA 3

- Als Ganzes formal erst Ende 2002 freigegeben
- Teilstandards Stück für Stück bereits in CORBA 2.3 ... 2.6 integriert
- CORBA 2.3
  - Objekte als Wertparameter
  - XML-Abbildungen
  - Java-RMI als Schnittstellenmodell in CORBA
    - Java-CORBA-Koevolution
    - Java als wichtigste Plattform für Implementierung der Standards
- CORBA 2.4
  - Objektreferenzen als URL
  - asynchrone Botschaften
  - Minimal- und Realzeit-CORBA
- CORBA 2.5: Fehlertoleranz- und Abbruch-Standards
- CORBA 2.6: Sicherheitsstandards
- CORBA 3: **Meilenstein** ist Komponentenmodell (CCM)
  - im Wesentlichen fertig seit Ende 2001

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

# Basisdienste (CORBAservices)

- Fokus auf fundamentale Bausteine **jeder** Implementierung, welche die Komponenten-Infrastruktur zur Verfügung stellen sollte
  - z.B. Ereignisbehandlung, Transaktionsabwicklung, Namensverwaltung
  - derzeit 16 Basisdienste spezifiziert
- 2 Kategorien:
  - Dienste zur Unterstützung unternehmensweiter verteilter Anwendungen
    - nutzen typischerweise CORBA-Objekte als Moduln und CORBA als Kommunikations-Middleware
    - grob granulare Ebene, CORBA Kommunikations-Infrastruktur als „Objektbus“
  - Dienste zur Unterstützung fein granularer verteilter Anwendungen
    - Bedeutung nimmt ab, da zu hoher Komplexitätsgrad
    - Ausnahme: persistent state service (PSS) als einer der Pfeiler des CCM
- große CORBA-Anwendungen nutzen oft nur wenige Basisdienste
  - verfügbare CORBA-Plattformen bieten deshalb oft nur Implementierungen einiger Basisdienste an

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

# Dienste zur Unterstützung grob granularer verteilter Anwendungen

## Namensdienst (name service)

- Abbildung intern verwendeter UUID auf externe Bezeichner
- Namens**-Kontexte** und Kontext-**Hierarchien**
  - vergleichbar zu Verzeichnisstrukturen

## Händlerdienst (trader service)

- Verfeinerung des Namensdiensts (white vs. yellow pages)
- Anbieter veröffentlichen Dienstangebote per **Registrierung**
- Nutzer finden Angebote über Händlerdienst per **Beschreibung**
- Händlerdienste organisieren Angebote in Handels**-Kontexten**
- Standardisierte Methoden zur Suche in den Angeboten

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

## Ereignisdienst (event service)

- Verteilung der E.-**Objekte** von E.-**Erzeugern** (event supplier) an E.-**Konsumenten** (event consumer)
- E.-Objekte sind unveränderbar, wenn einmal erzeugt
  - strikt unidirektionaler Informationsfluss
- E.-**Kanäle** (event channel) entkoppeln Erzeuger und Konsument
- E. können getypt sein (OMG IDL)
- Kanäle können Ereignisse nach ihrem Typ filtern
- Push- und Pull-Methoden werden unterstützt

## Benachrichtigungsdienst (notification service)

- Erweiterung des Ereignisdiensts um einige kritische Merkmale
  - Dienstqualität, Administration
  - dynamische E.-Filterung, Filterung auf verschiedenen Ebenen
- technisch kein Basisdienst, sondern Standardkomponente
  - gemeinsamer Standard mit Telecomm. Domain Task Force

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

Transaktionsdienst (object transaction service, OTS)

- einer der wichtigsten Bausteine für verteilte Anwendungen
- standardisiert seit 1994
- wird von den meisten ORB-Produkten und J2EE-Servern unterstützt
- **Eingebettete Transaktionen** nur optional
  - Transaktionshülle um Folge von Operationen
  - erforderlich zur unabhängigen Entwicklung auf verschiedenen Hierarchie-Ebenen
  - (noch) nicht standardisiert, weil kaum eines der heute ex. Transaktionssysteme so etwas vorsieht
- Verwaltung eines (objektspez.) aktuellen Tr.-**kontexts** durch OTS
  - Objekte müssen dazu die Schnittstelle *TransactionalObject* implementieren
  - Methoden *begin*, *commit*, *rollback* operieren auf dem Kontext
- Objekte unter Transaktionskontrolle registrieren sich beim OTS-Koordinator-Objekt

UNIVERSITÄT LEIPZIG
Institut für Informatik
Betriebliche Informationssysteme

## Transaktionsdienst (Fortsetzung)

- Ressourcen müssen die Schnittstelle *Resources* implementieren
  - Koordinator wickelt darüber 2-Phasen-commit-Protokoll ab
  - bekanntes Problem der Deadlock-Gefahr
    - 3-Phasen-Protokoll vermeidet diese, ist aber teurer
- heute weit verbreitet: Transaktionskontrolle nicht als separater Dienst, sondern als Kontextkontrolle **innerhalb** eines Anwendungsservers
  - Diese Abstraktion wird vom CCM abgedeckt

## Sicherheitsdienst (security service)

- erforderlich, wenn sich verteilte Anwendung über mehrere Vertrauensbereiche (trusted domains) erstreckt
- spezifiziert in **CORBAsecurity**
- Authentifizierung, sichere Kommunikation, Zertifizierung
- volles Spektrum wird derzeit von kaum einem Produkt unterstützt
  - meist nur SSL-basierte Sicherheit
  - unterstützt einfache Sicherheit, aber keine Zertifikate