Hasso Plattner Institut

IT Systems Engineering | Universität Potsdam

Informationsintegration
Architekturen

22.5.2008

Felix Naumann

Überblick

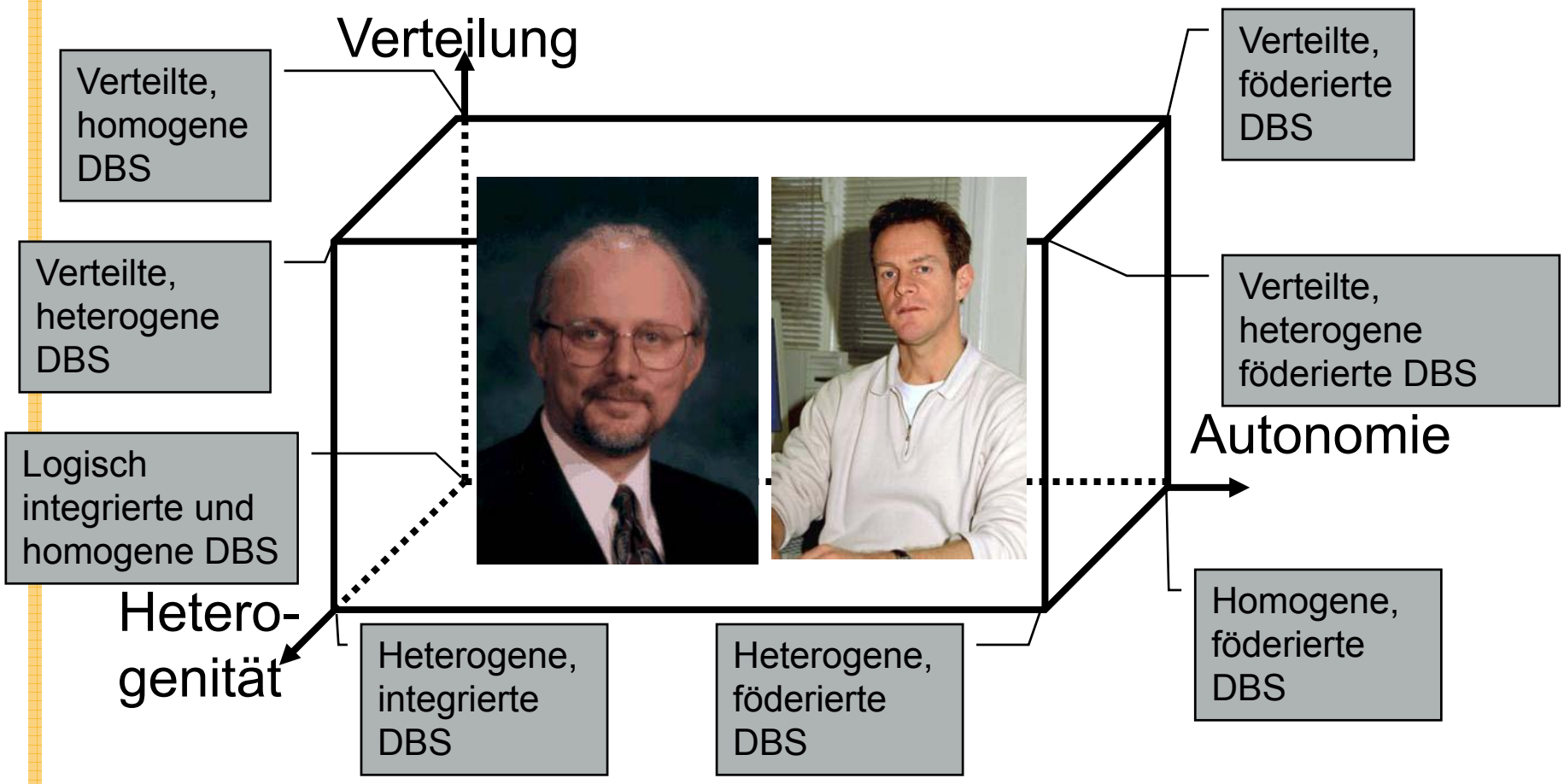
- Überblick über Informationssysteme
  - Klassifikation
  - Weitere Kriterien
- Architekturen
  - 3 Schichten Architektur
  - 4 Schichten Architektur
  - 5 Schichten Architektur
- Mediator-Wrapper Architektur
  - Gio Wiederholds Definitionen
  - Konfigurationen
  - Mediatoren
  - Wrapper
- Peer-Data-Management
  - Architektur
  - Anwendungen

Klassifikation von Informations-systemen nach [ÖV99]

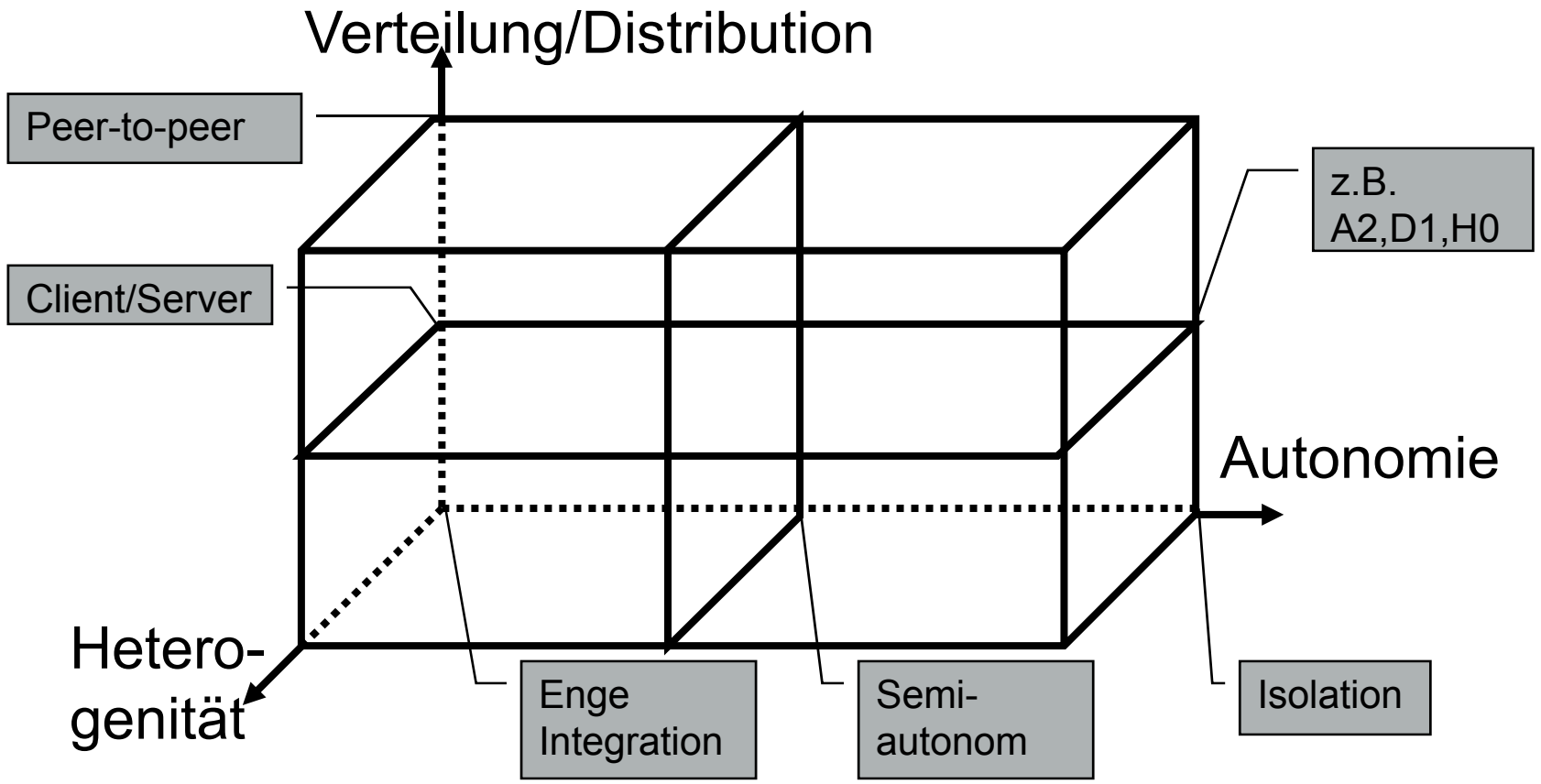
- Orthogonale Dimensionen
  - Verteilung
  - Autonomie
  - Heterogenität

- Orthogonal in der Lösung der jeweiligen Probleme
- Nicht unbedingt orthogonal in ihrer Ursache

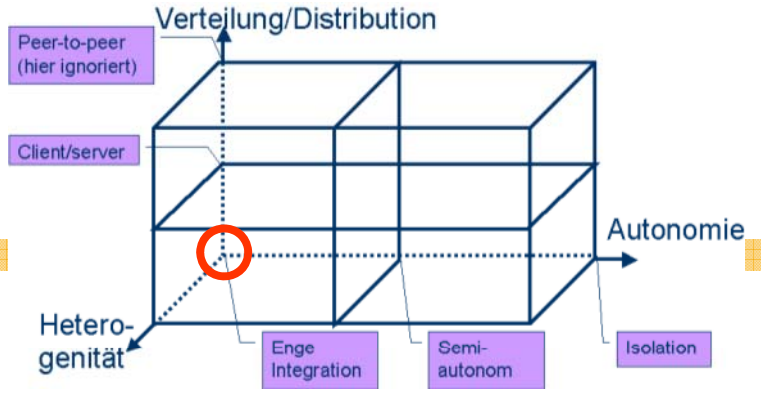
Klassifikation von Informations-systemen nach [ÖV91]



Erweiterung der Klassifikation nach [ÖV99]

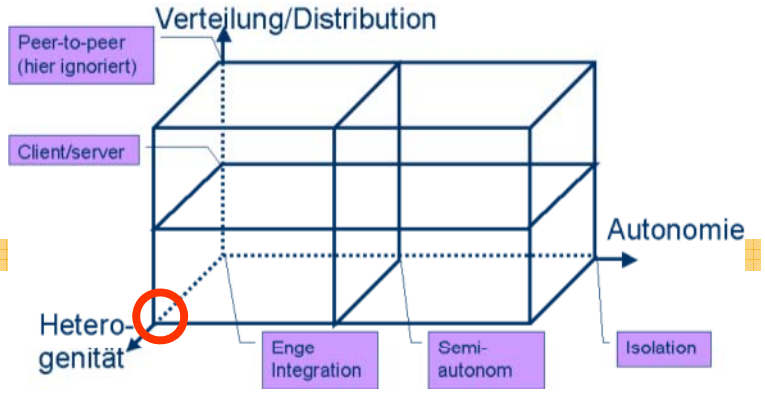


Aut0, Dist0, Het0



- Zwar nicht verteilte, aber dennoch mehrere DBMS.
- Logisch integrierte, homogene Datenbanken
- „Composite System“
- Nicht üblich
- Höchstens für Shared-Everything Multiprozessor Systeme

Aut0, Dist0, Het1

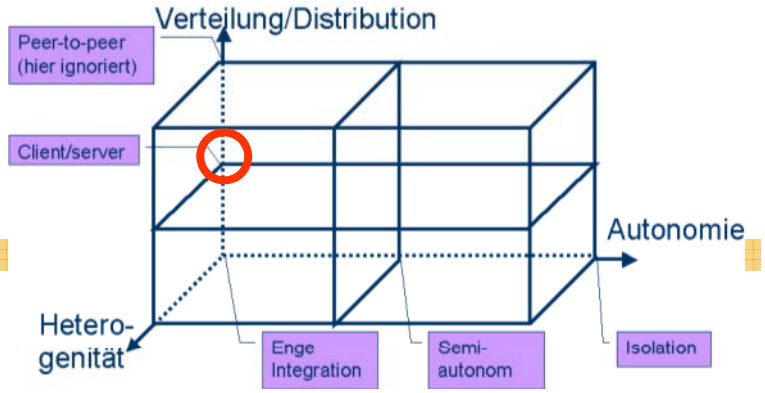


- Heterogenes, integriertes DBS
- Mehrere DBMS, einheitliche Sicht für Nutzer
- Anwendungsbeispiel
  - Integrierter Zugriff auf mehrere
    - verschiedene DBMS (hierarchisch, relational,...)
    - auf einer Maschine.

Keine Autonomie und keine Verteilung

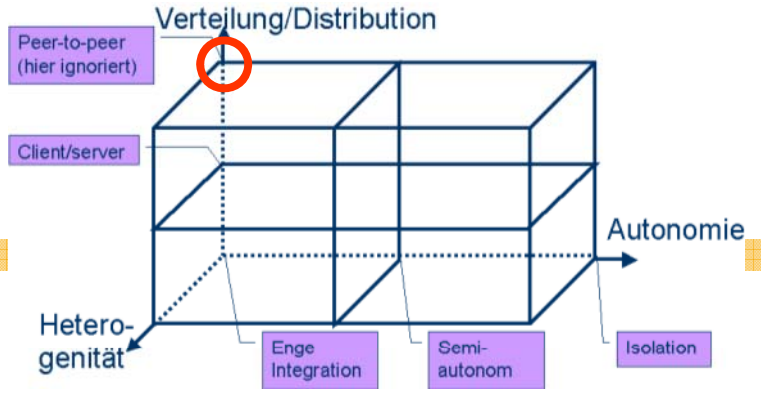
Heterogenität

Aut0, Dist1, Het0



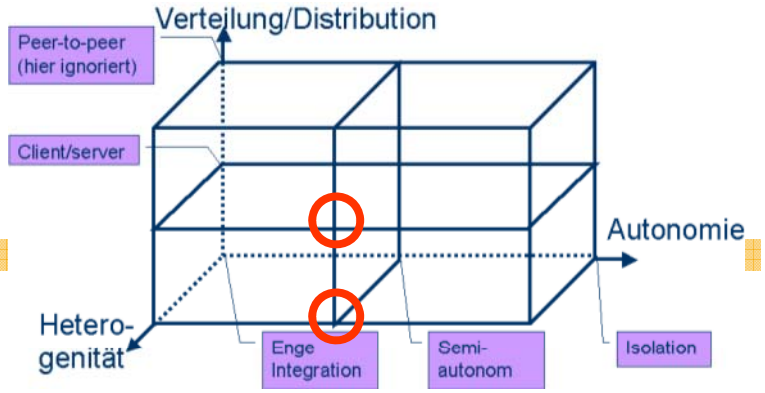
- Client/Server Verteilung
- Physische Verteilung der Daten
- Einheitliche Sicht für Nutzer
- Server
  - Datenmanagement, Optimierung, Transaktionen
- Client
  - Anwendung, GUI, Cache Management
- Kommunikation mittels SQL
- Auch
  - Multiple Client / Single Server
  - Multiple Client / Multiple Server

Aut0, Dist2, Het0



- P2P Verteiltes Informationssystem
- Wie A0,D1,H0, aber keine Unterscheidung zwischen Client und Server
- „PDMS“ ohne Probleme der Heterogenität und Verteilung

Aut1, Dist0, Het1

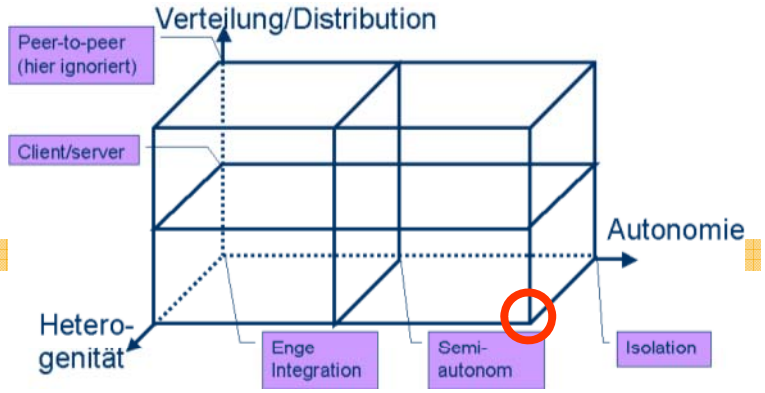


- Heterogene, föderierte DBMS
- Z.B. Katalog DBMS und Image DBMS auf einer Maschine, die gemeinsamen Zugriff erlauben.
- Typisches Szenario

Und

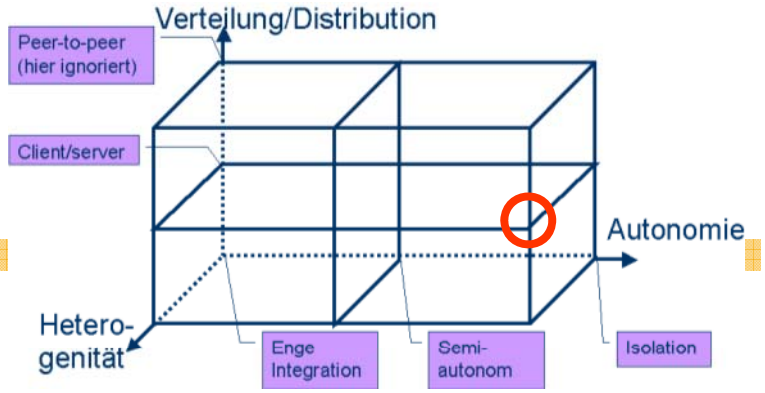
- Verteilte, heterogene, föderierte DBMS
- Verteilung birgt nur wenige neue Probleme

Aut2, Dist0, Het1



- Wie A1, D0, H1
- Noch realistischer
- Keine Interoperation untereinander möglich
- Integration nur in neuer, integrierender Komponente.
- Z.B. DBMS und WWW Server auf einer Maschine
  - Nicht zur Interoperation entwickelt
    - DBMS „spricht" kein http, WWW „spricht" kein SQL

Aut2, Dist1/2, Het1

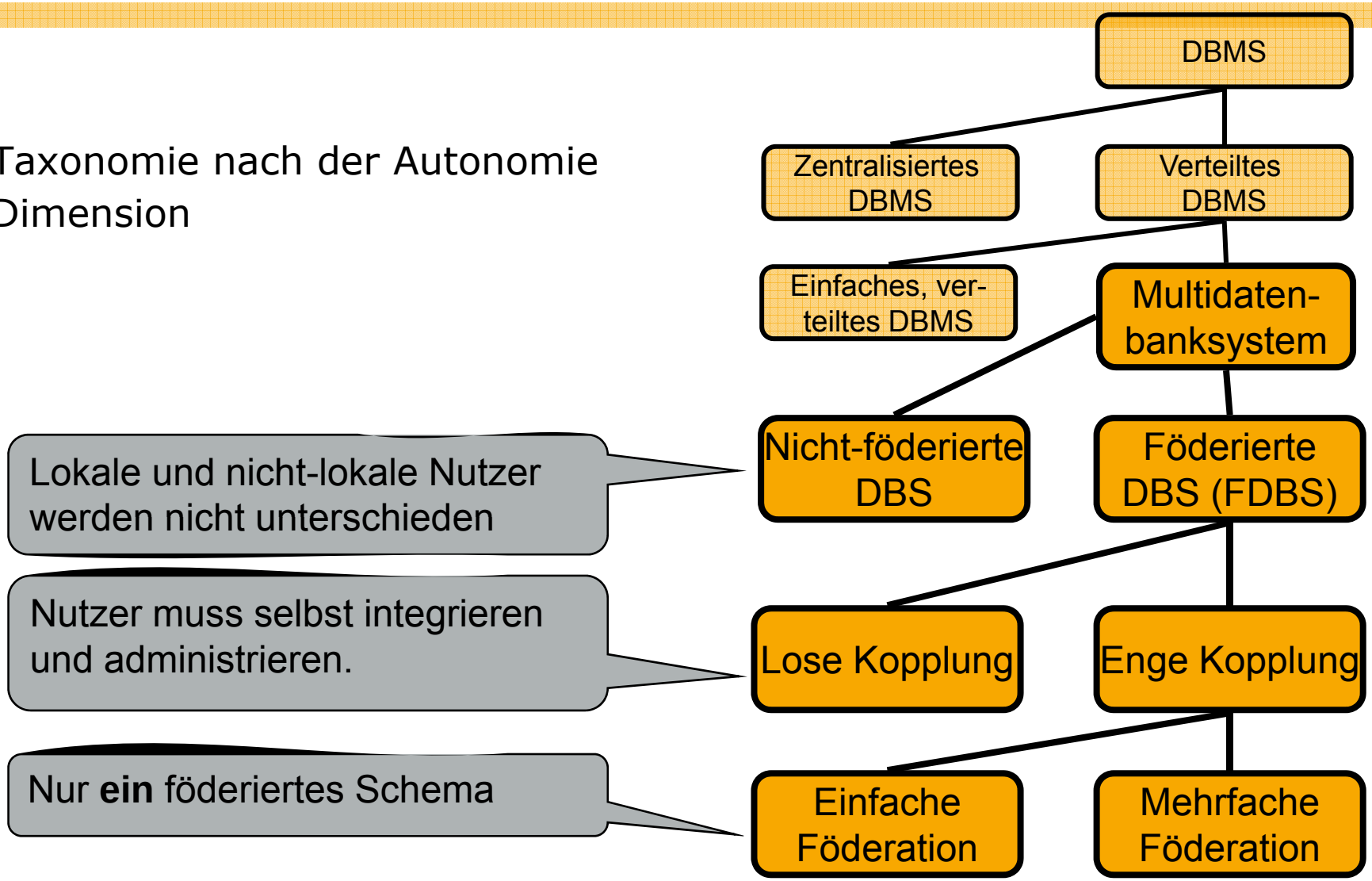


- Verteilte MDBMS
- Schwierigster Fall
- Wie A2,D0,H1
  - Verteilungsprobleme eher technisch

- Auch: Mediator-basierte Systeme

Taxonomie nach [SL90]

Taxonomie nach der Autonomie Dimension

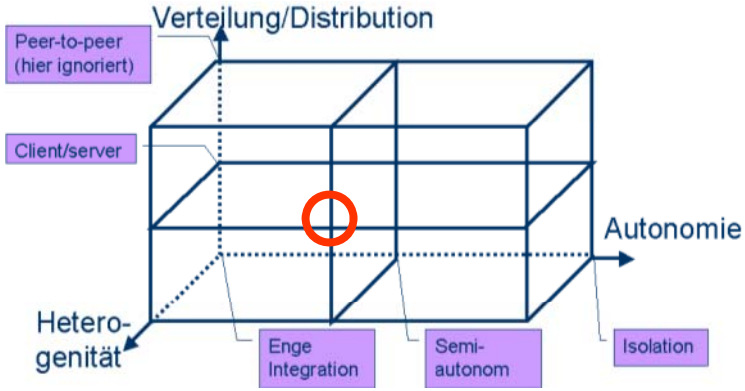


Überblick

- Überblick über Informationssysteme
  - Klassifikation
  - Weitere Kriterien
- Architekturen
  - 3 Schichten Architektur
  - 4 Schichten Architektur
  - 5 Schichten Architektur
- Mediator-Wrapper Architektur
  - Gio Wiederholds Definitionen
  - Konfigurationen
  - Mediatoren
  - Wrapper
- Peer-Data-Management
  - Architektur
  - Anwendungen

Kriterien föderierter Informationssysteme nach [BKLW99]

- Weitere (nicht-orthogonale) Kriterien
  - Strukturierung der Komponenten
  - Enge und lose Kopplung
  - Datenmodell
  - Art der semantischen Integration
  - Transparenz
  - Anfrage-Paradigma
  - Bottom-up oder Top-down Entwurf
  - Virtuell oder materialisiert
  - Read-only oder read-&-write
- Gelten zumeist auch für MDBMS



1. Strukturierung der Komponenten

- Strukturiert
  - Festes Schema, festes Format
  - Beispiel: Datenbanken
- Semi-strukturiert
  - Struktur vorhanden, aber nur teilweise bekannt
  - Daten sind mit Semantik gelabelt
  - Beispiel: XML Dokumente (und OEM Daten)
- Unstrukturiert
  - Keine Struktur
  - Beispiel: Textuelle Daten, Abstracts

2. Enge und lose Kopplung

- Enge Kopplung
  - Festes, integriertes/föderiertes Schema
    - Modelliert mit Korrespondenzen
  - Feste Anfragesprache
- Lose Kopplung
  - Kein festes Schema
    - Nutzer müssen Semantik der Quellen kennen
    - Integrierte Sichten helfen
  - Feste Anfragesprache
  - Multidatabase query language (MDBQL) [LMR90]
  - SchemaSQL

3. Datenmodell

- Kanonisches Datenmodell (im integrierten System)
  - Objektorientiertes Modell
  - Relationales Modell
  - Hierarchisches Modell
  - Semistrukturiertes Modell
  - XML Datenmodell
- Verlustfreie Integration ist schwierig
  - Beispiel: OO to Relational Mapping
    - Datenquelle: OO, kanonisches Datenmodell: Relational
    - Schlüssel erfinden
    - Nicht-Atomare Attribute müssen untergebracht werden.
      - struct, set, bag, list, array
    - Semantische Beziehungen gehen verloren (Schlüssel/Fremdschlüssel)

4. Art der Semantischen Integration

- Vereinigung
  - Simple „Konkatenation“ von Objekten
- Anreicherung
  - Mit Metadaten
- Fusion
  - Objektidentifizierung
  - Re-Strukturierung
  - Komplementierung
    - Mehrere Objekte werden zu einem integriert
  - Aggregation
    - Konfliktlösung

5. Transparenz

- Speicherorttransparenz
  - Physischer Ort der Daten unbekannt
  - IP, Quellenname, DB Name
- Schematransparenz
  - Nutzer sehen nur integriertes Schema
  - Strukturelle Konflikte werden verborgen
- Sprachtransparenz
  - Nutzer muss nur eine Sprache beherrschen
  - Integriertes System übersetzt.

6. Anfrage-Paradigma

- Strukturierte Anfragen
  - Struktur ist Nutzern bekannt .
  - Struktur kann in Anfrage verwendet werden.
  - Z.B. DBMS + SQL
- „Canned queries“
  - Vordefinierte Anfragen (parametrisiert)
- Such-Anfragen
  - Struktur unbekannt
  - Information Retrieval
  - Z.B. Suchmaschinen auf Texten
- Browsing
  - Kein Such-Interface
  - WWW

7. Bottom-up oder Top-down Entwurf

- Beim Entwurf des integrierten Systems
- Bottom-up:
  - Ausgelöst durch den Bedarf mehrere Quellen integriert anzufragen
  - Schemaintegration ist wichtig.
  - Änderungen schwierig, da neu integriert werden muss.
  - Typisches Szenario: Data Warehouse
- Top-down
  - Ausgelöst durch globalen Informationsbedarf
  - Vorteilhaft bei labilen Quellen
  - Schemaintegration nicht nötig, bzw. leichter
  - Typisches Szenario: Virtuelle Integration

7. Bottom-up Entwicklung

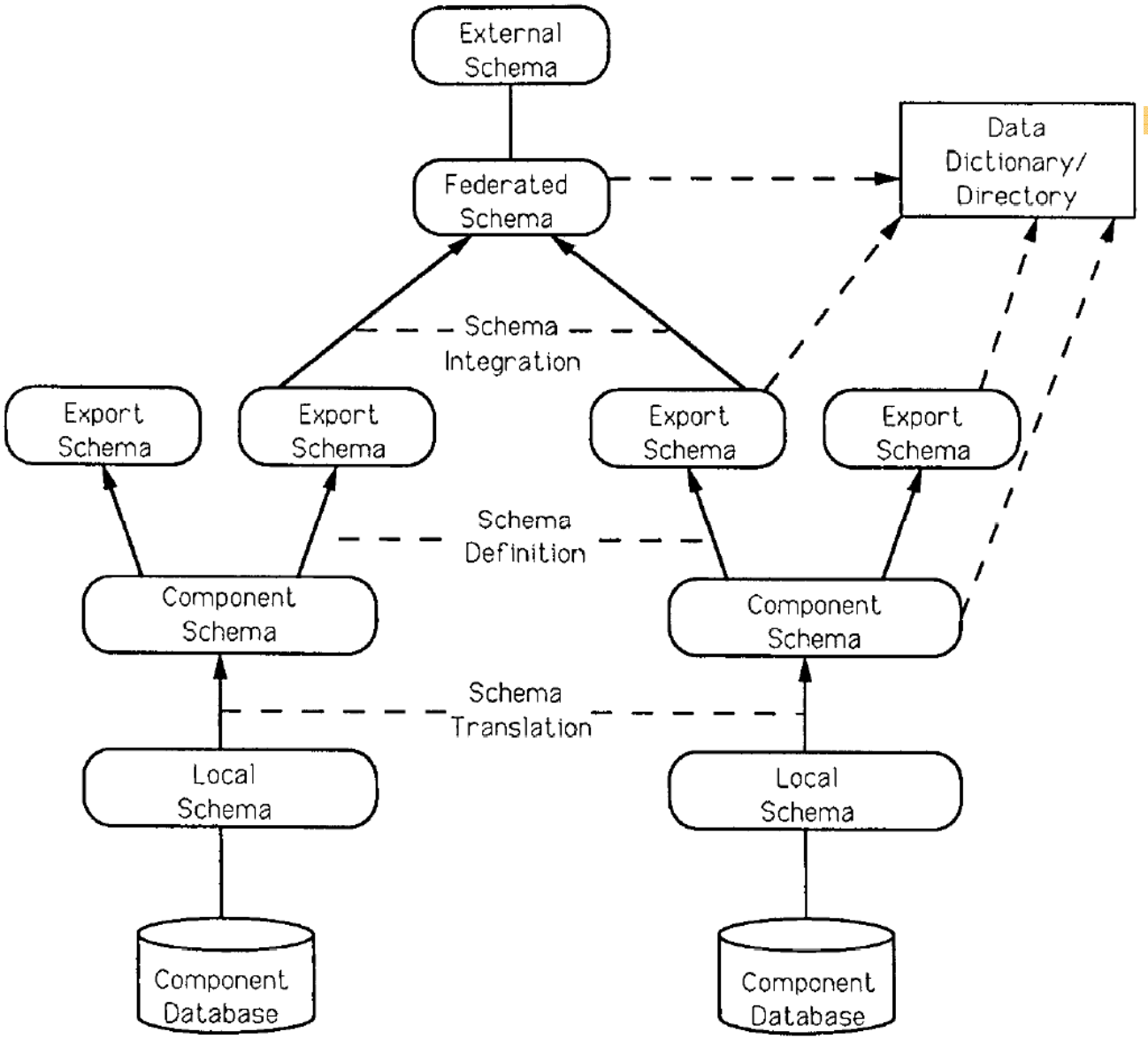
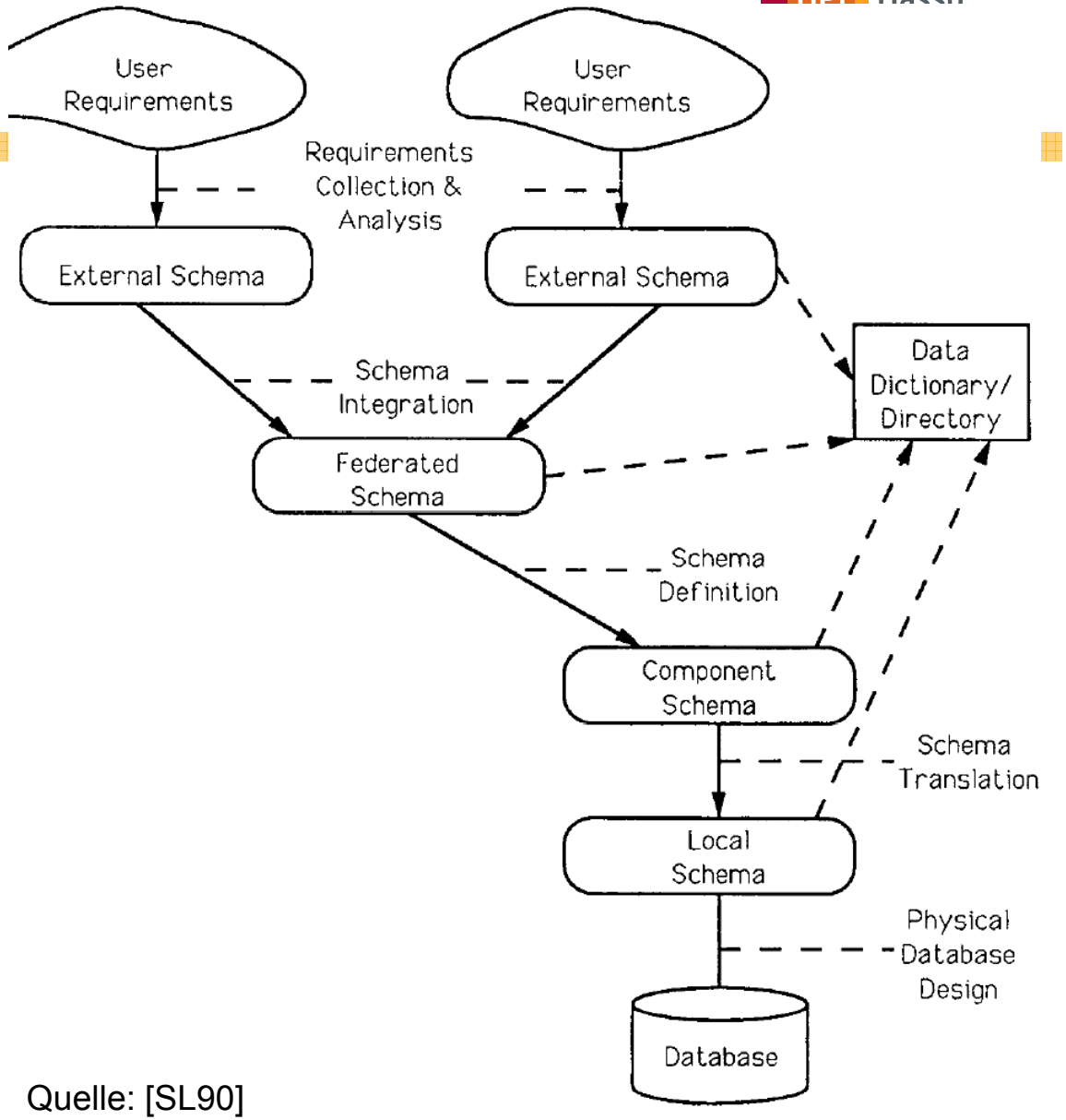


Figure 21. Bottom-up FDBS developing process. Quelle: [SL90]

7. Top-down Entwicklung



Quelle: [SL90]

Figure 22. Top-down FDBS developing process.

8. Virtuell oder materialisiert

- Virtuell
  - Anfragen werden in Teilanfragen übersetzt.
  - Daten werden nur bei Bedarf übertragen und nur temporär gespeichert.
- Materialisiert
  - Daten werden transformiert und lokal gespeichert.
  - Anfragen werden direkt gegen die materialisierten Daten gestellt.

9. Read-only oder read-&-write

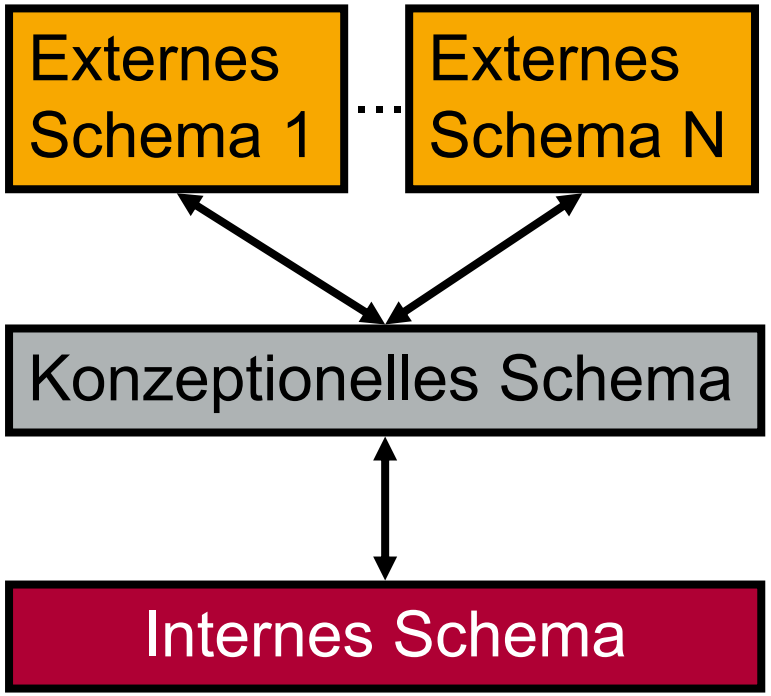
- Read-only die beliebtere Variante
- Write (insert & update) schwierig
  - Autonomie!
  - Viele Interfaces erlauben kein Schreiben
  - Update durch Sichten ist schwierig
  - Globale Transaktionen (komplexe Protokolle)
  - Bei Komplementierung: Welche Quelle?

Überblick

- Überblick über Informationssysteme
  - Klassifikation
  - Weitere Kriterien
- Architekturen
  - 3 Schichten Architektur
  - 4 Schichten Architektur
  - 5 Schichten Architektur
- Mediator-Wrapper Architektur
  - Gio Wiederholds Definitionen
  - Konfigurationen
  - Mediatoren
  - Wrapper
- Peer-Data-Management
  - Architektur
  - Anwendungen

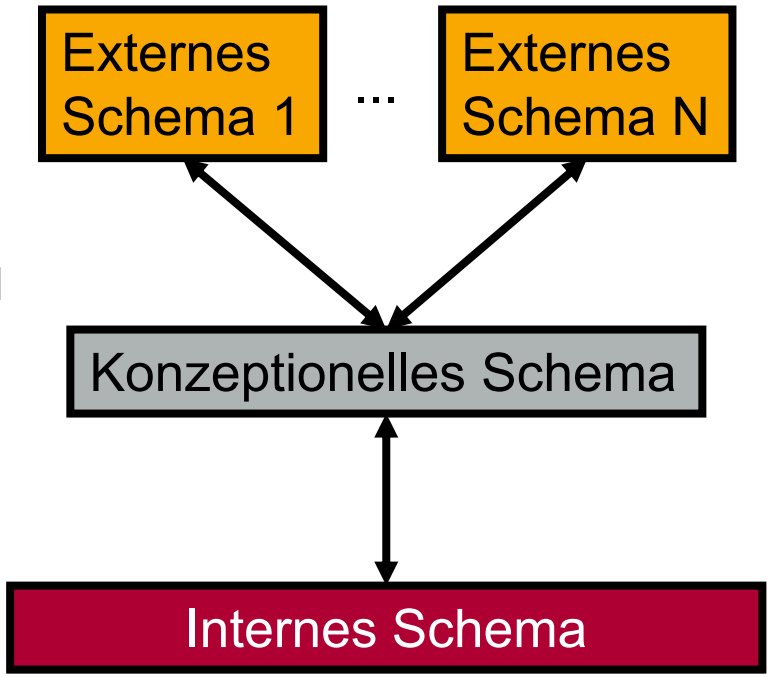
3-Schichten Architektur

ANSI/SPARC
3-Schichten Architektur für zentralisierte DBMS

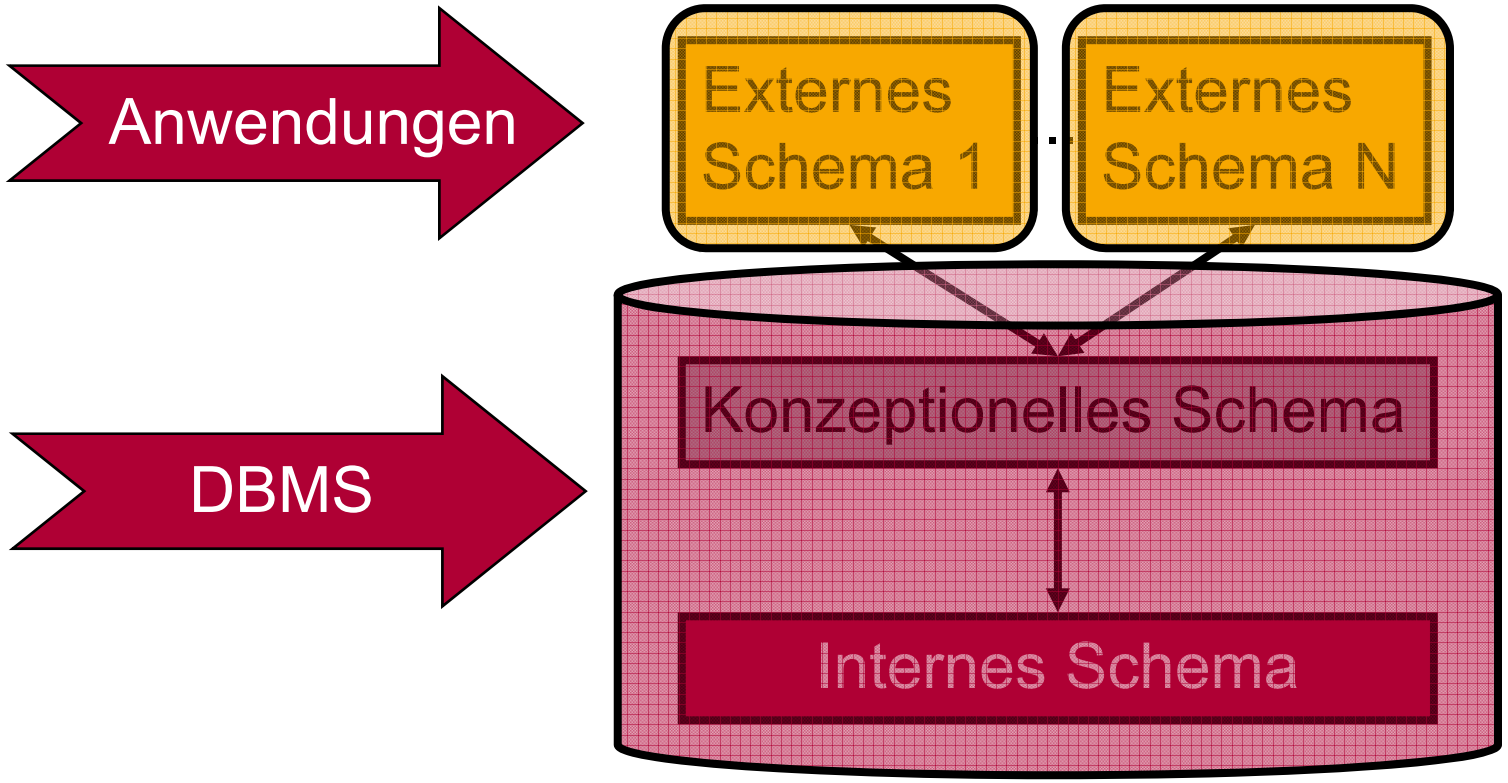


Wdh: Das Schichtenmodell

- Interne (physische) Sicht
  - Speichermedium (Tape, Festplatte)
  - Speicherort (Zylinder, Block)
- Konzeptionelle (logische) Sicht
  - Unabhängig von physischer Sicht
  - Definiert durch Datenmodell
  - Stabiler Bezugspunkt für interne und externe Sichten
- Externe (logische) Sicht
  - Anwendungsprogramme
  - Nur auf die relevanten Daten
  - Enthält Aggregationen und Transformationen

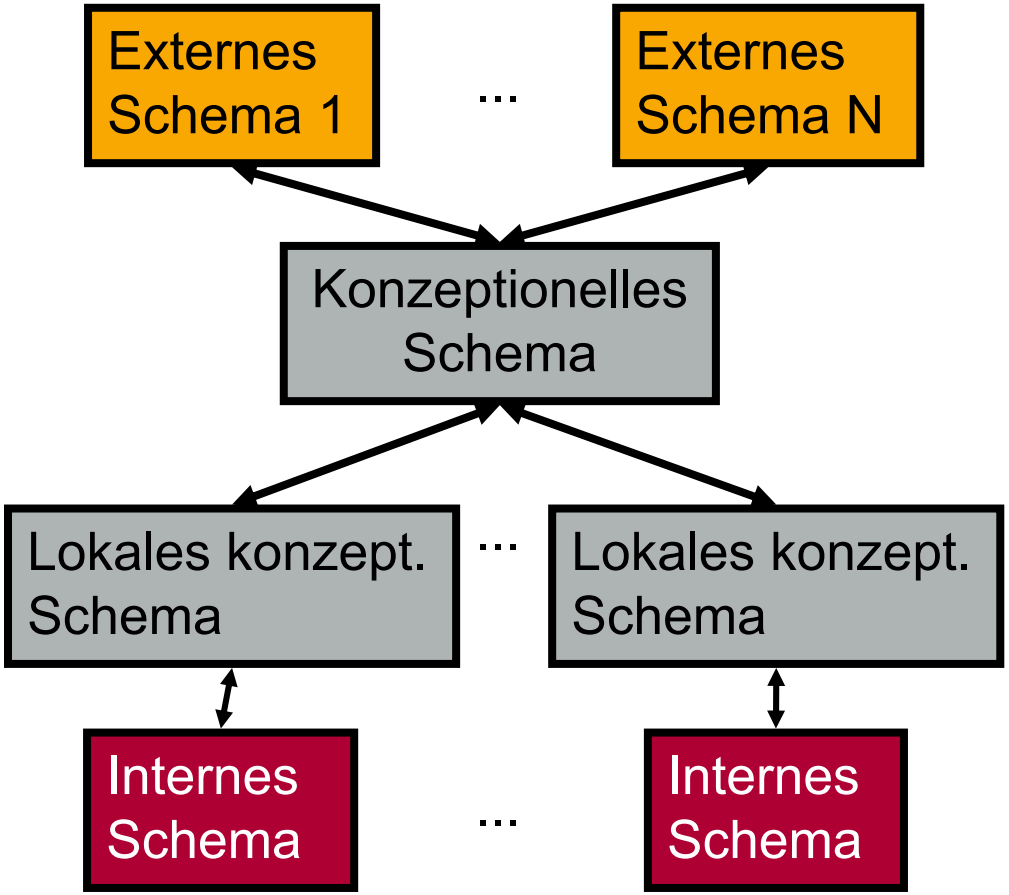


3-Schichten Architektur

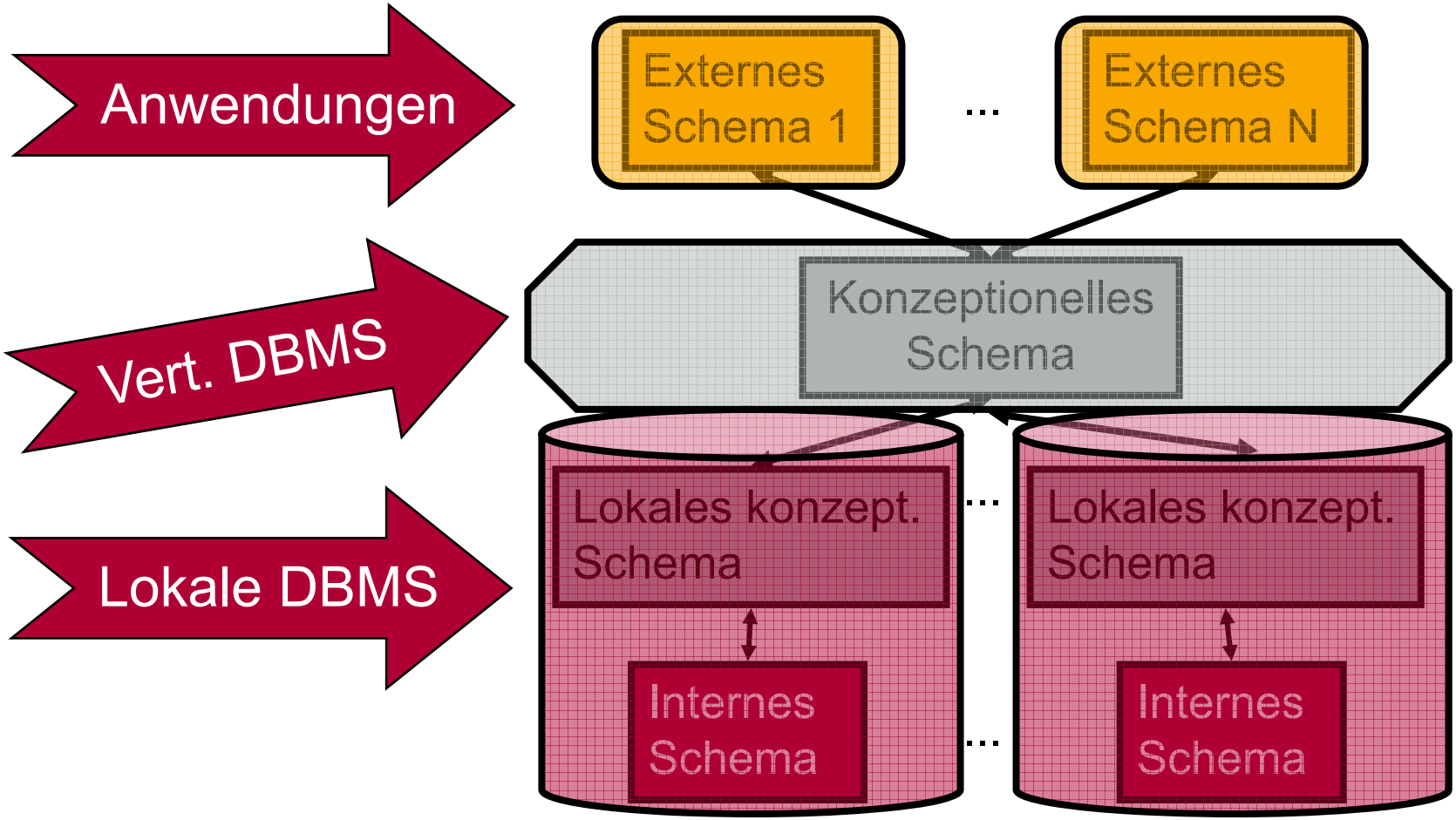


4-Schichten Architektur

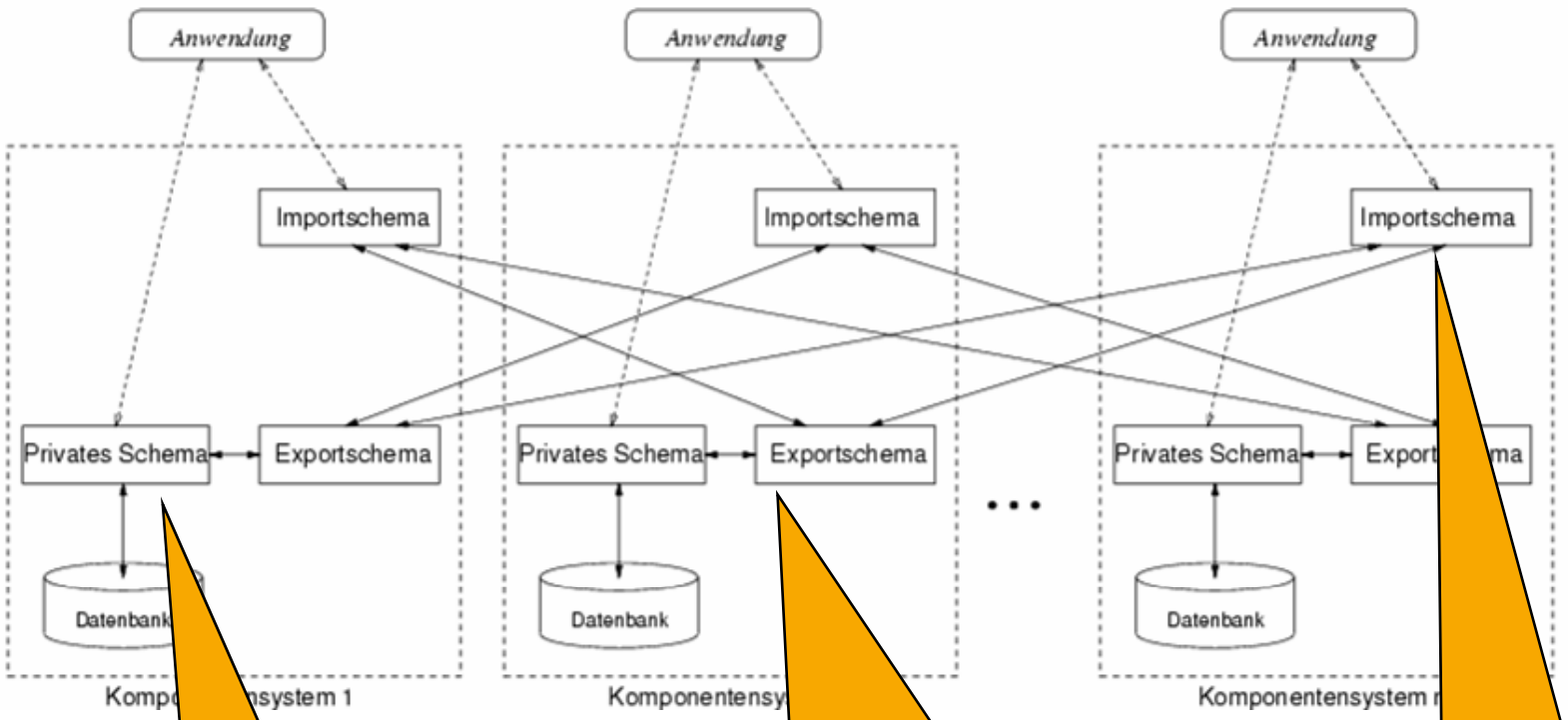
- Für verteilte DBMS
- Neu: Trennung lokales vs. globales konzeptionelles
- Globales Konzeptionelles Schema ist integriert aus den lokalen konzeptionellen Schemas.
- Lokales und globales konzept. Schema kann gleich sein.



4-Schichten Architektur



Import-/Export-Schema-Architektur nach [HM85]



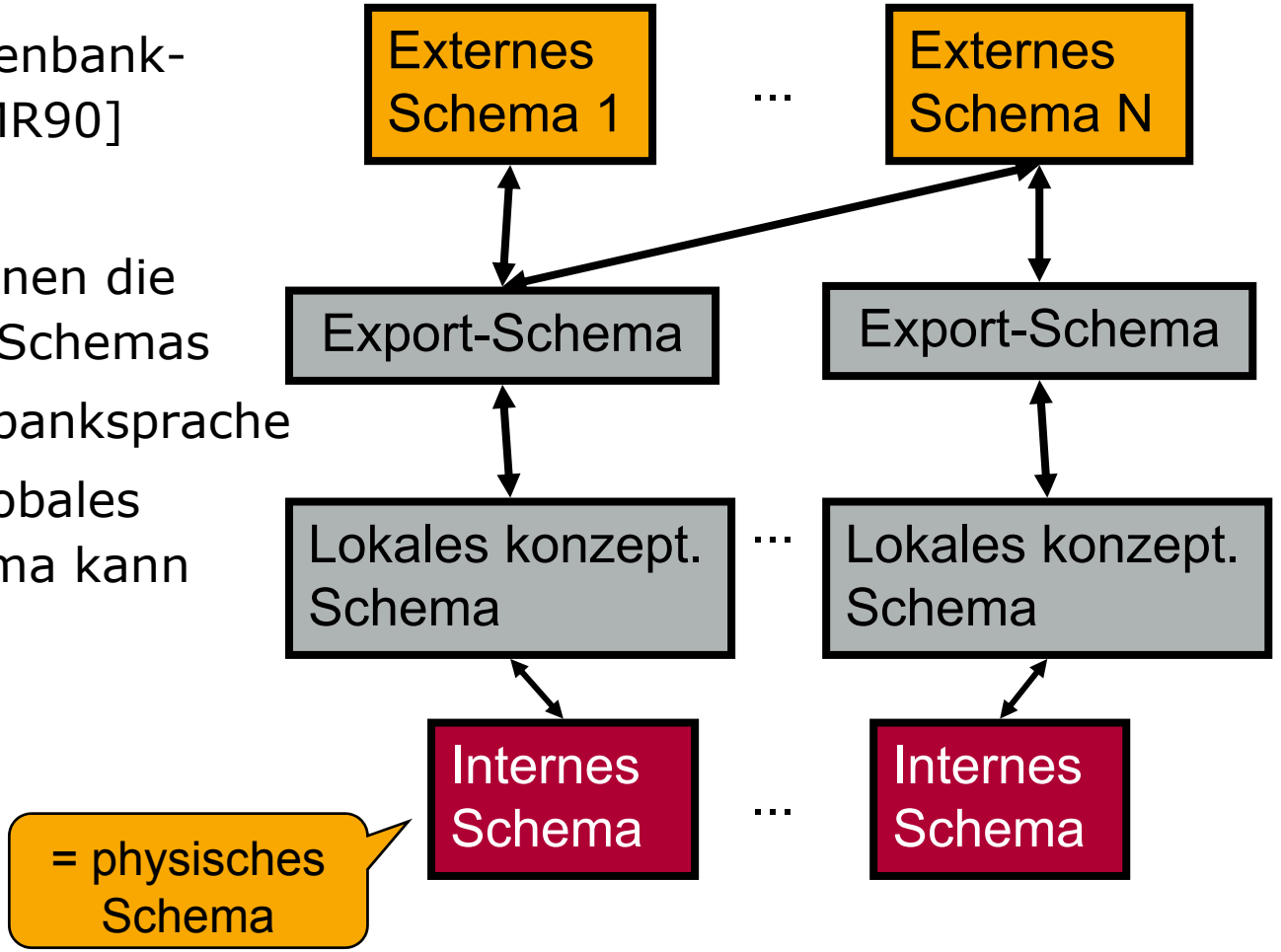
= lokales konzeptionelles Schema

Idee: Nur Teilmenge des lokalen konzeptionellen Schemas wird der Föderation zur Verfügung gestellt.

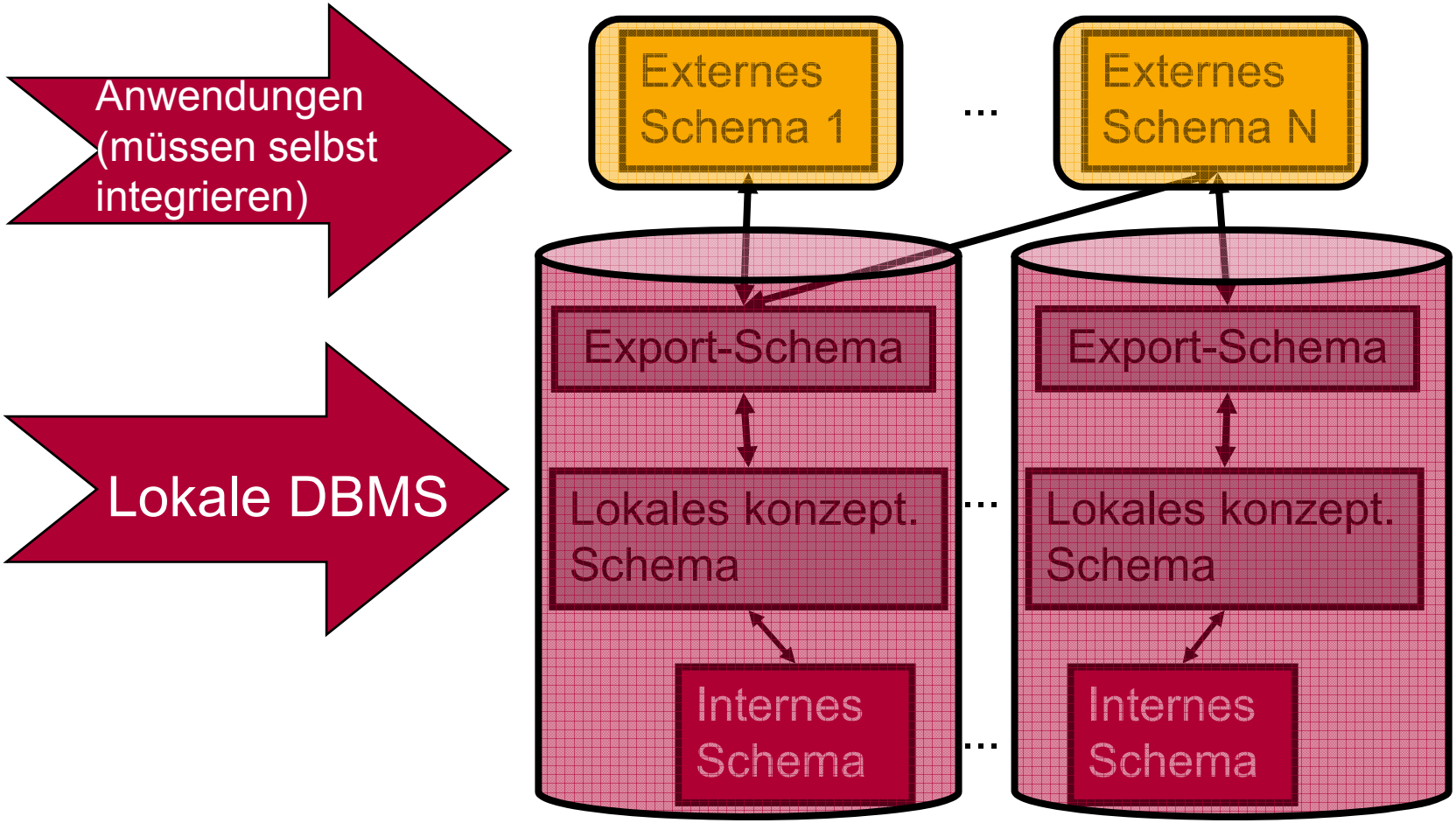
Idee: Nur Teilmengen der Exportschemas sollen verwendet werden.

4-Schichten Architektur

- Auch: Multidatenbank-architektur [LMR90]
- Voraussetzung
  - Nutzer kennen die jeweiligen Schemas
  - Multidatenbanksprache
- Lokales und globales konzept. Schema kann gleich sein.
- Lose Kopplung

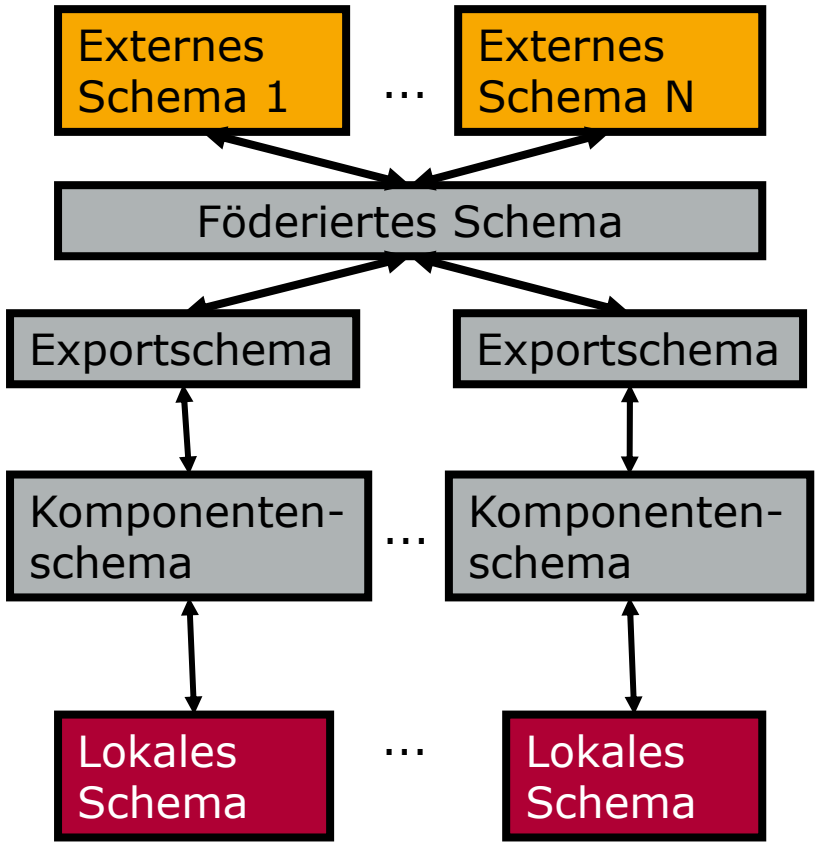


4-Schichten Architektur

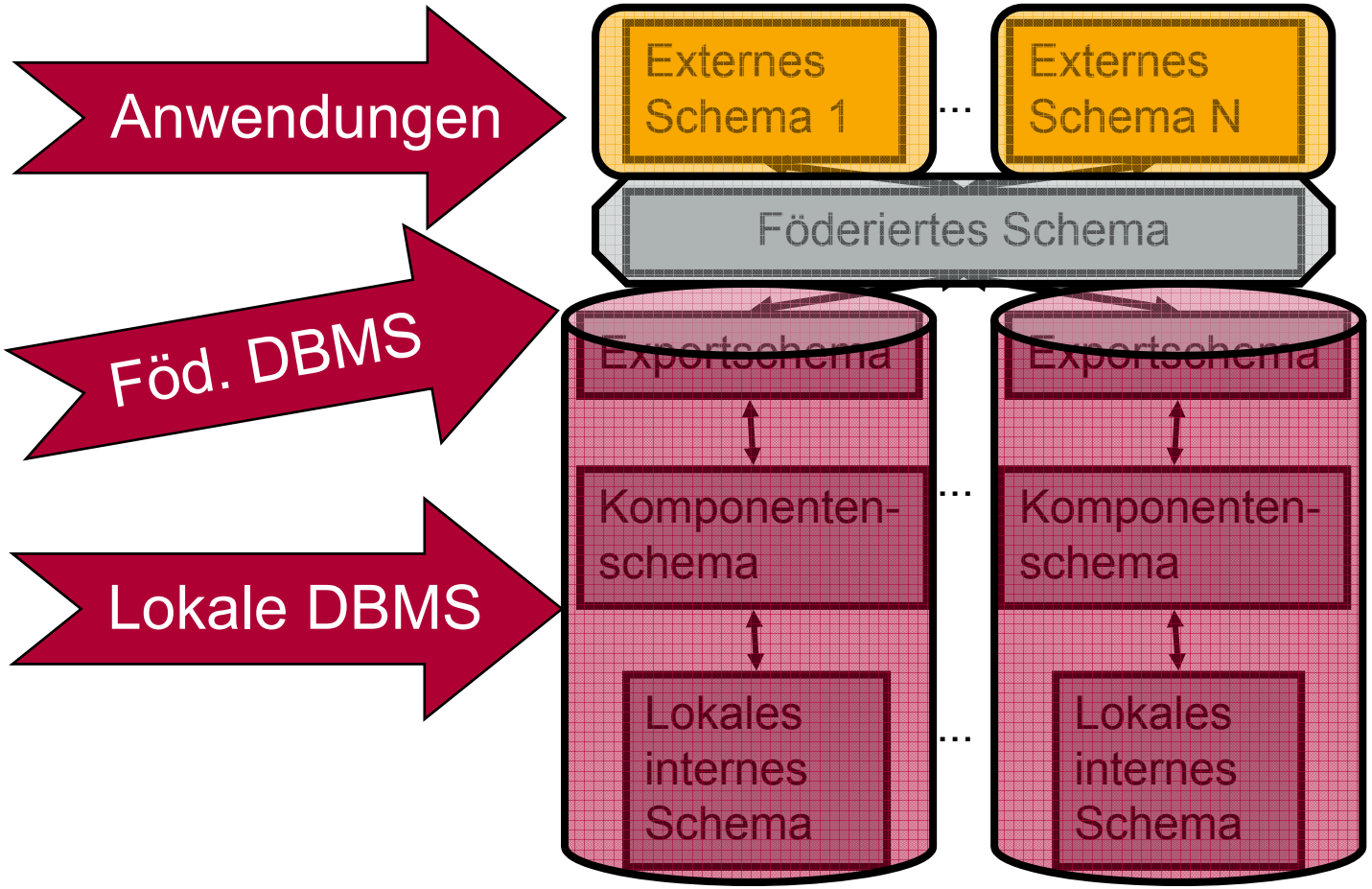


5-Schichten Architektur [SL90]

- Neu:
  - Interne Schemas werden nicht mehr betrachtet.
  - Exportschemas
  - Integriertes, föderiertes Schema
- Terminologie
  - Komponentenschema = lokales konzept. Schema
  - Föderiertes Schema = globales konzept. Schema

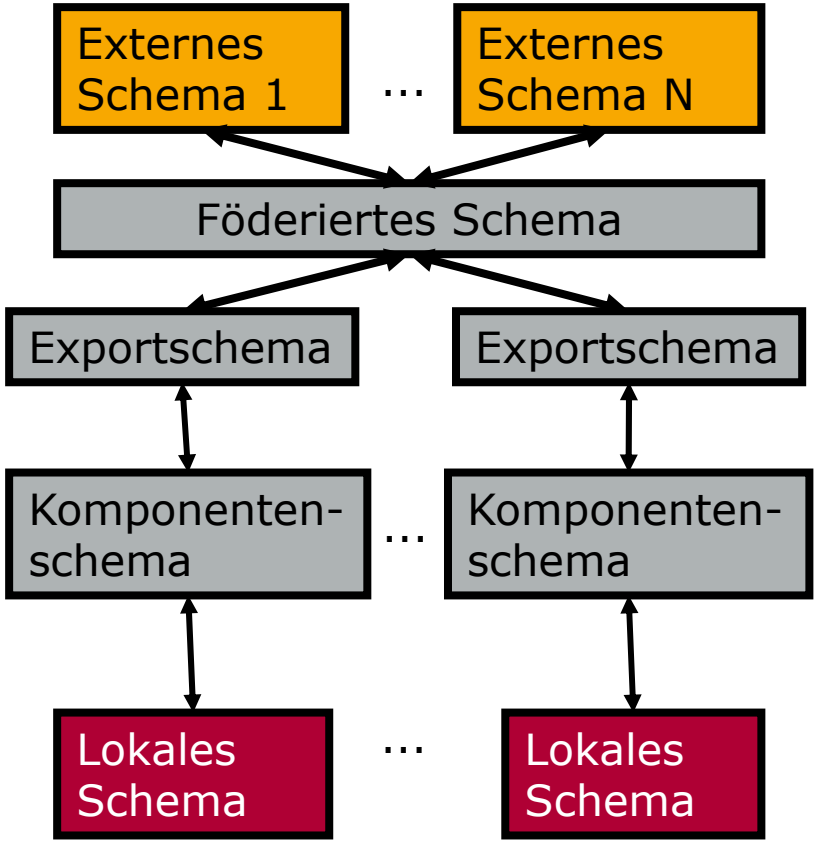


5-Schichten Architektur [SL90]



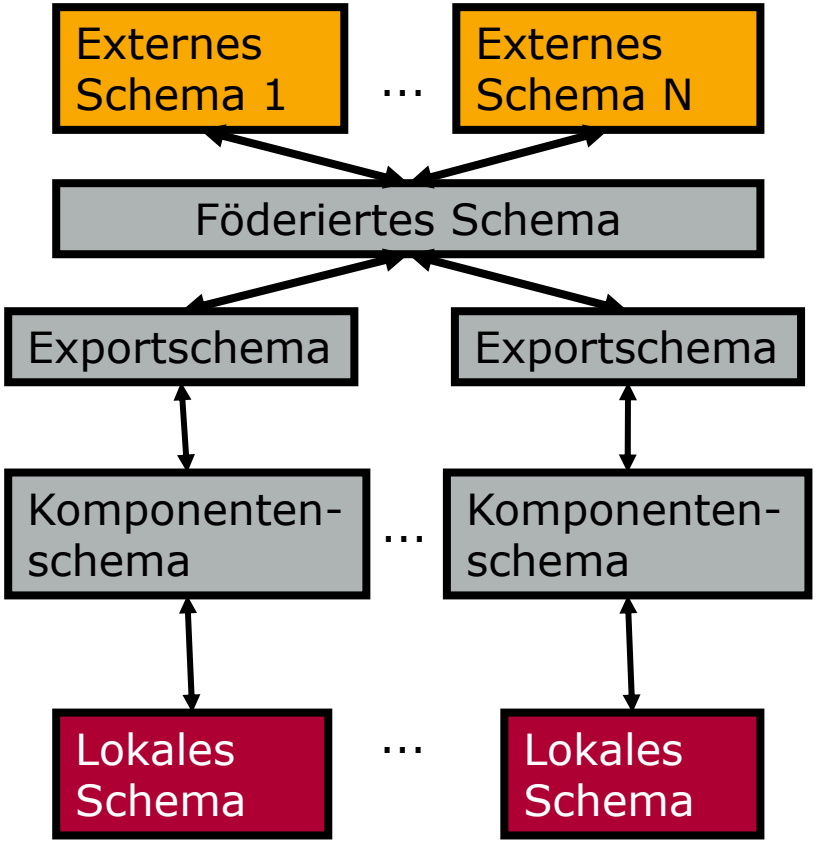
5-Schichten Architektur [SL90]

- Lokale Schemas
  - Konzeptionell
- Komponentenschemas
  - Kanonisches Datenmodell
  - Fügt fehlende Semantik hinzu.
  - Übergang durch Mappings.
- Exportschemas
  - Teilmenge des Komponentenschemas
  - Verwaltet Zugangsberechtigungen



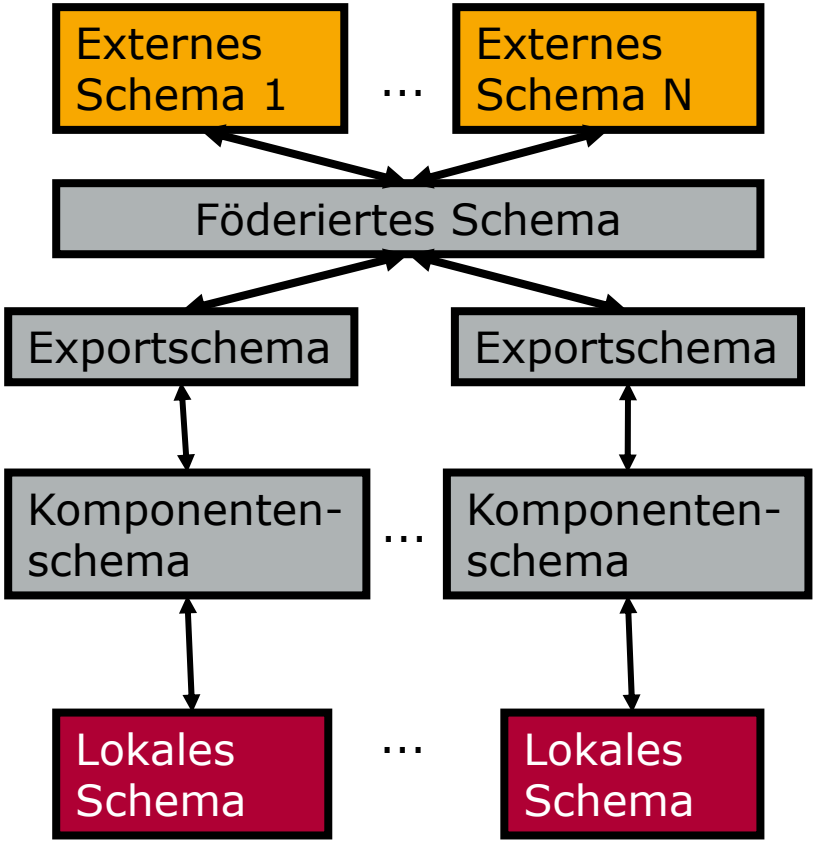
5-Schichten Architektur [SL90]

- Föderiertes Schema
  - Integriert aus den Exportschemas
  - Kennt Datenverteilung
  - Andere Namen:
    - Import Schema
    - Globales Schema
    - Enterprise Schema
    - Unified Schema
- Externes Schema
  - Föderiertes Schema kann sehr groß sein → Vereinfachung im Exportschema
  - „Schema Evolution" leichter
  - Zusätzliche Integritätsbedingungen
  - Zugangskontrollen



5-Schichten Architektur [SL90]

- Mischformen
  - Einige Schichten nicht immer nötig.
    - Z.B. wenn lokales und Komponentenschema gleich sind.
    - Z.B. wenn komplettes Komponentenschema exportiert werden soll.
  - Ein Komponentenschema kann mehrere Exportschemas haben.
  - Große FDBS können mehrere föderierte Schemas haben.
- Föderation!
  - Nur semi-autonom
  - Lokale DBMS müssen bereits kanonisches Datenmodell unterstützen.



Vergleich der Architekturen [Con97]

Import/Export

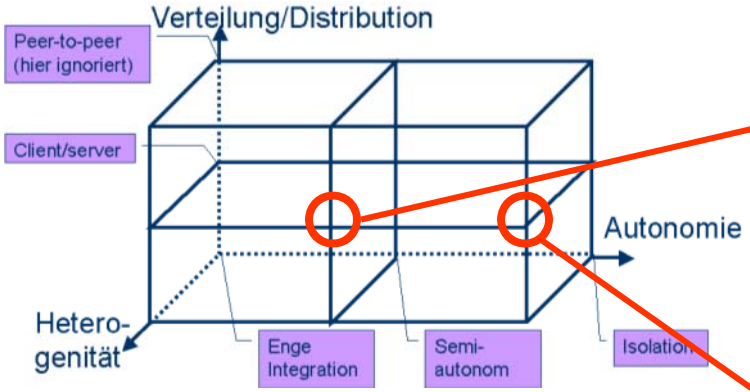
- Lose Kopplung
- Integration durch Nutzer und globalen Admin
- Zugriff über lokales System
- Keine globale DBMS-Funktionalität

- 4-Schichten
  - Lose Kopplung
  - Integration durch Nutzer
  - Zugriff über globale Schnittstellen
  - DBMS-Funktionalität nur durch Multidatenbank-sprachen

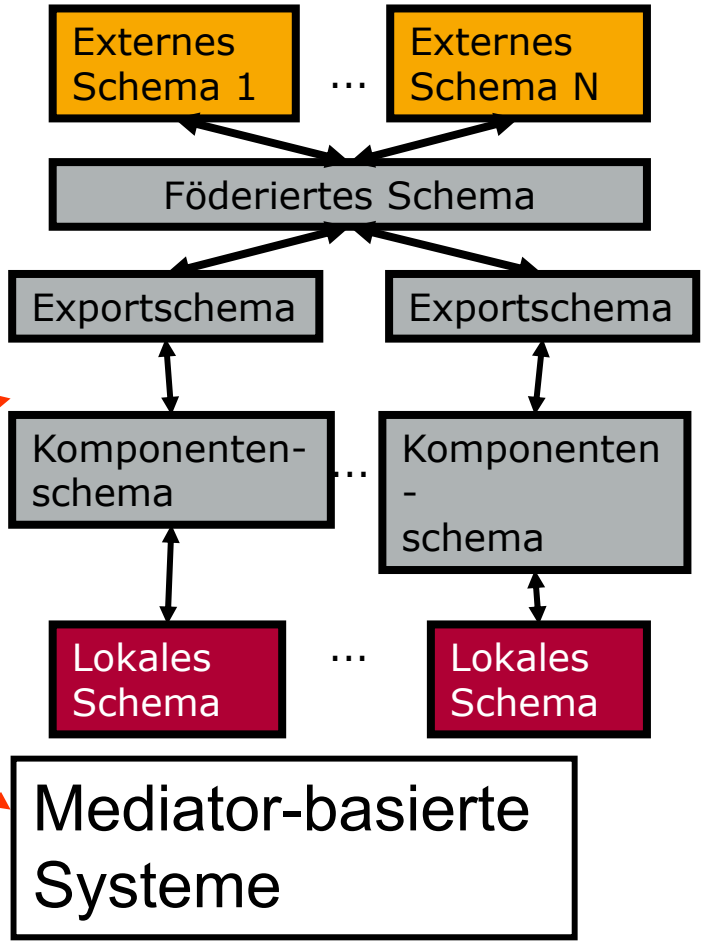
- 5-Schichten
  - Lose und enge Kopplung
  - Integration durch globalen Administrator
  - Zugriff durch globales System
  - DBMS-Funktionalität im globalen System

Zusammenfassung

- 1.



- 2.

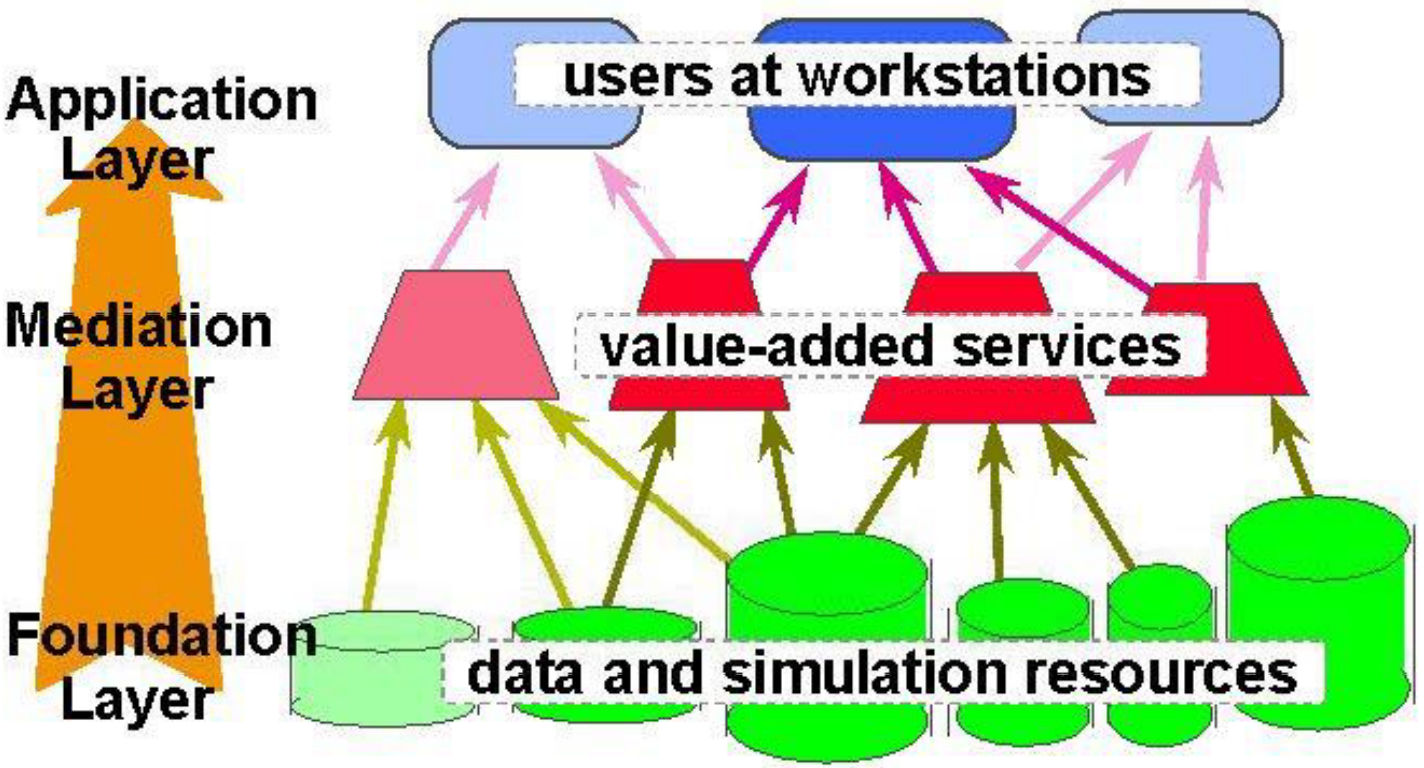


Überblick

- Überblick über Informationssysteme
  - Klassifikation
  - Weitere Kriterien
- Architekturen
  - 3 Schichten Architektur
  - 4 Schichten Architektur
  - 5 Schichten Architektur

- Mediator-Wrapper Architektur
  - Gio Wiederholds Definitionen
  - Konfigurationen
  - Mediatoren
  - Wrapper
- Peer-Data-Management
  - Architektur
  - Anwendungen

Daten werden zu Informationen



Mediatoren

- „A mediator is a software module that exploits encoded knowledge about certain sets or subsets of data to create information for a higher layer of applications“ Wiederhold `92 [Wie92]

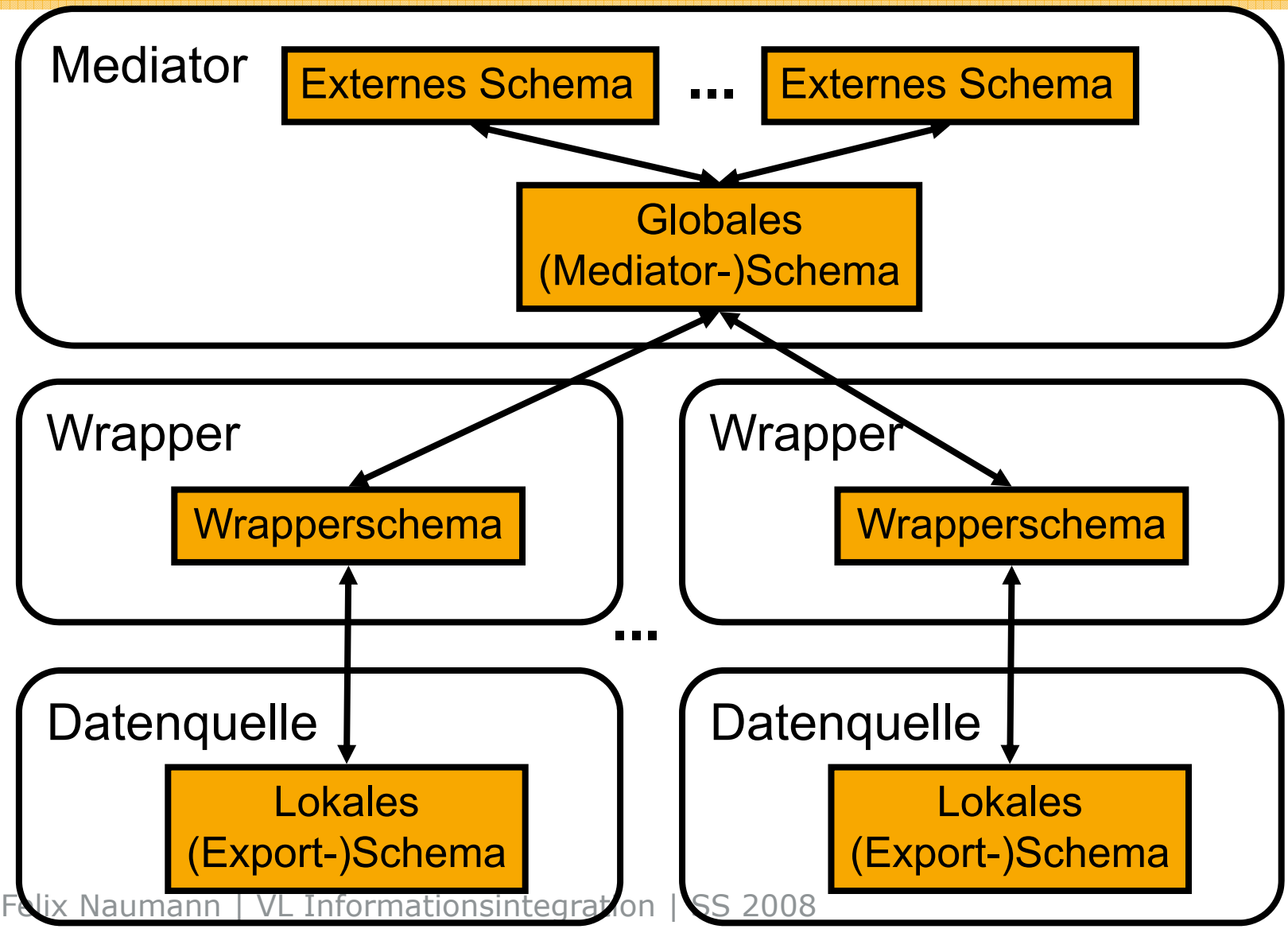
- Ein Mediator ist eine Softwarekomponente, die Wissen über bestimmte Daten benutzt, um Informationen für höherwertige Anwendungen zu erzeugen.

Überblick

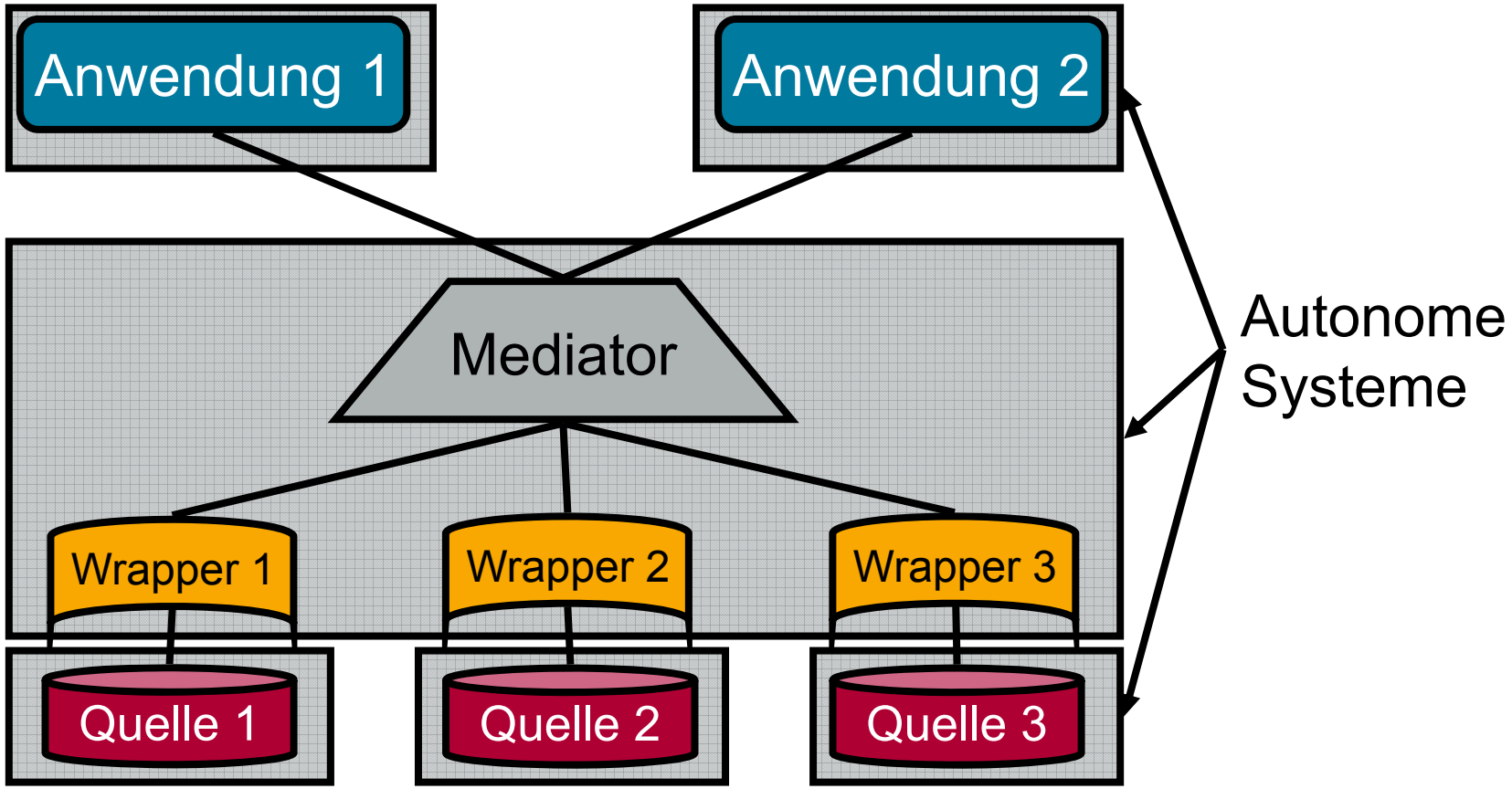
- Überblick über Informationssysteme
  - Klassifikation
  - Weitere Kriterien
- Architekturen
  - 3 Schichten Architektur
  - 4 Schichten Architektur
  - 5 Schichten Architektur

- Mediator-Wrapper Architektur
  - Gio Wiederholds Definitionen
  - Konfigurationen
  - Mediatoren
  - Wrapper
- Peer-Data-Management
  - Architektur
  - Anwendungen

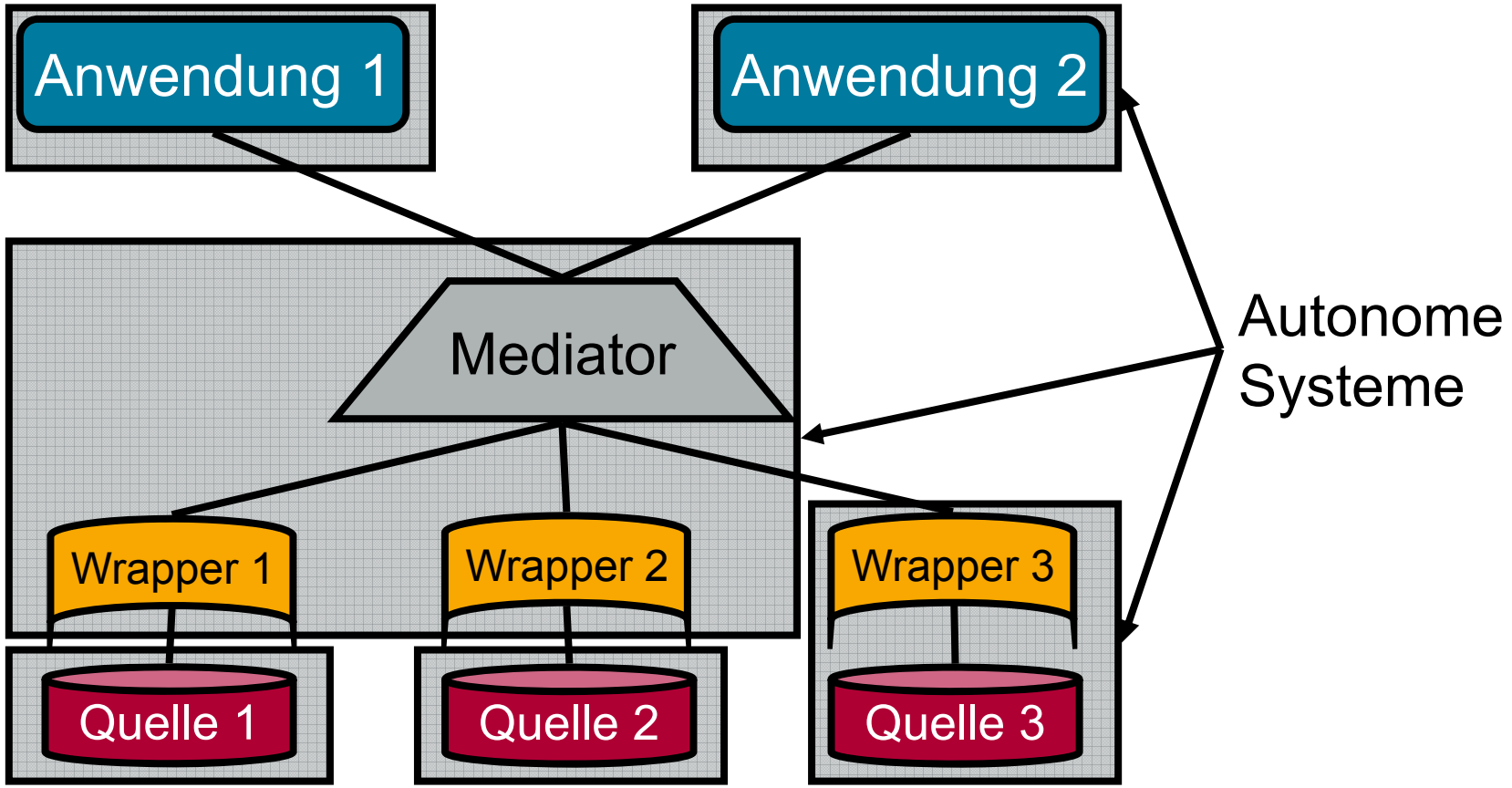
Die Architektur



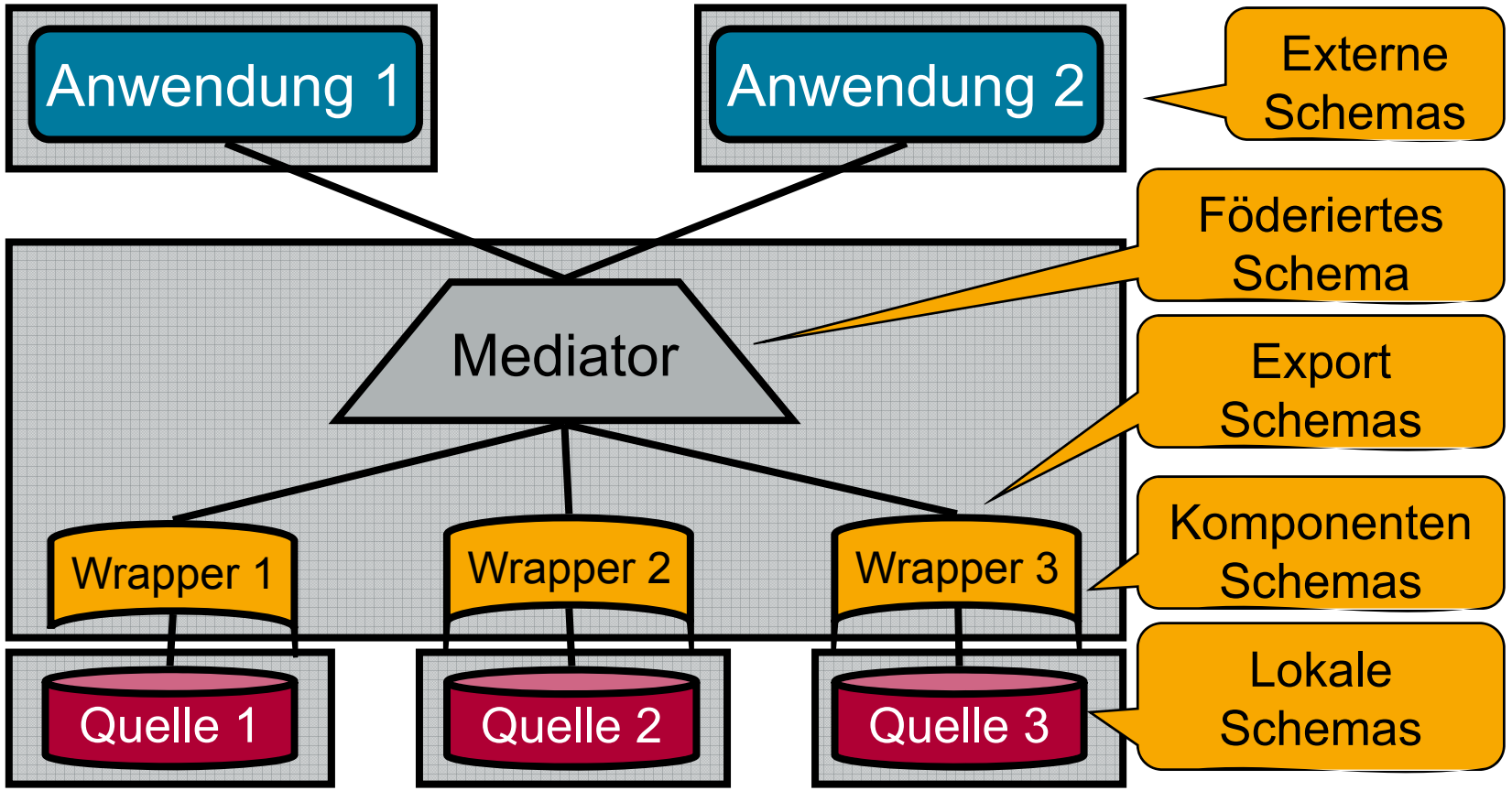
Mediator-Wrapper Architektur



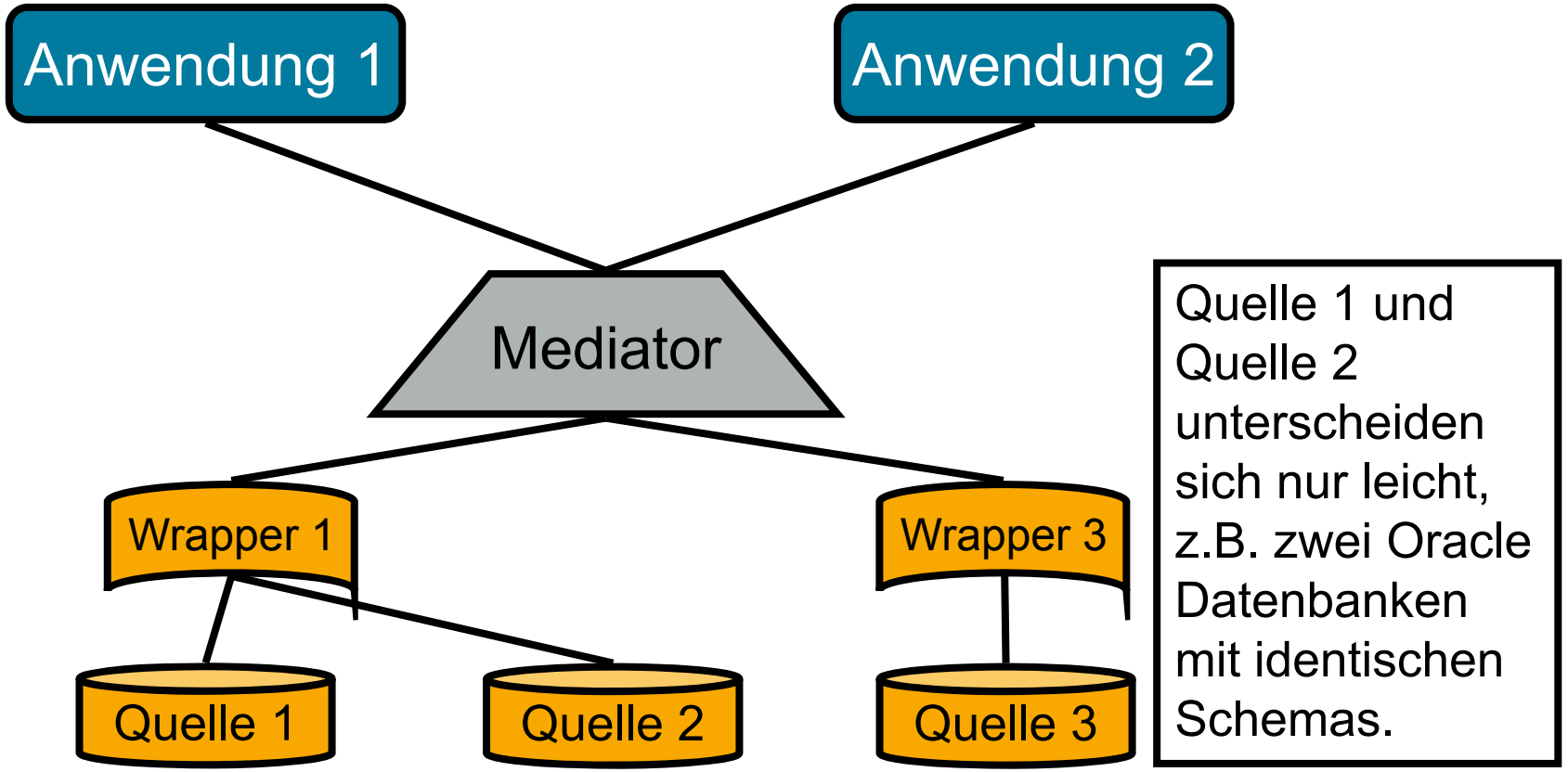
Mediator-Wrapper Architektur



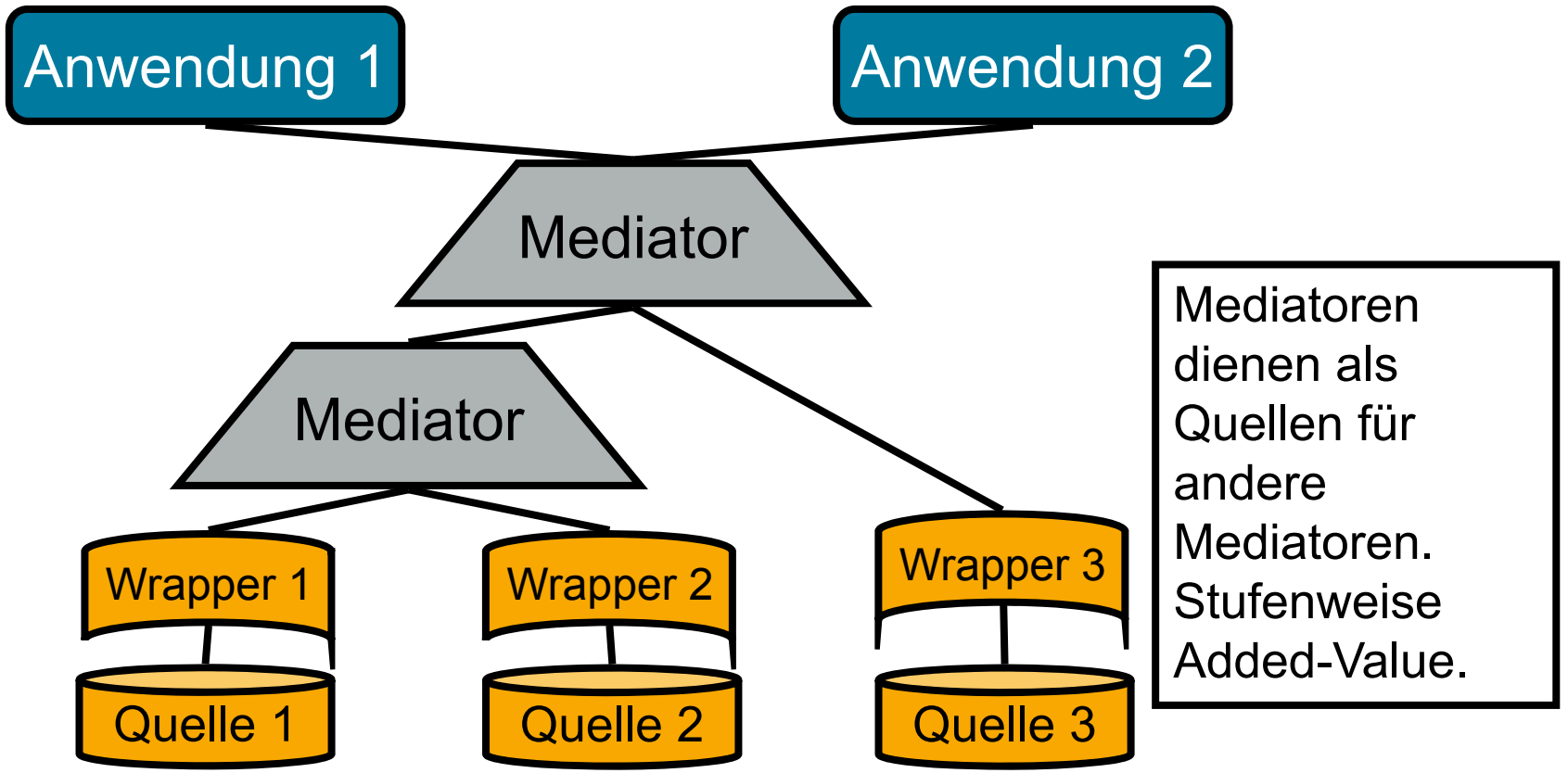
Mediator-Wrapper Architektur



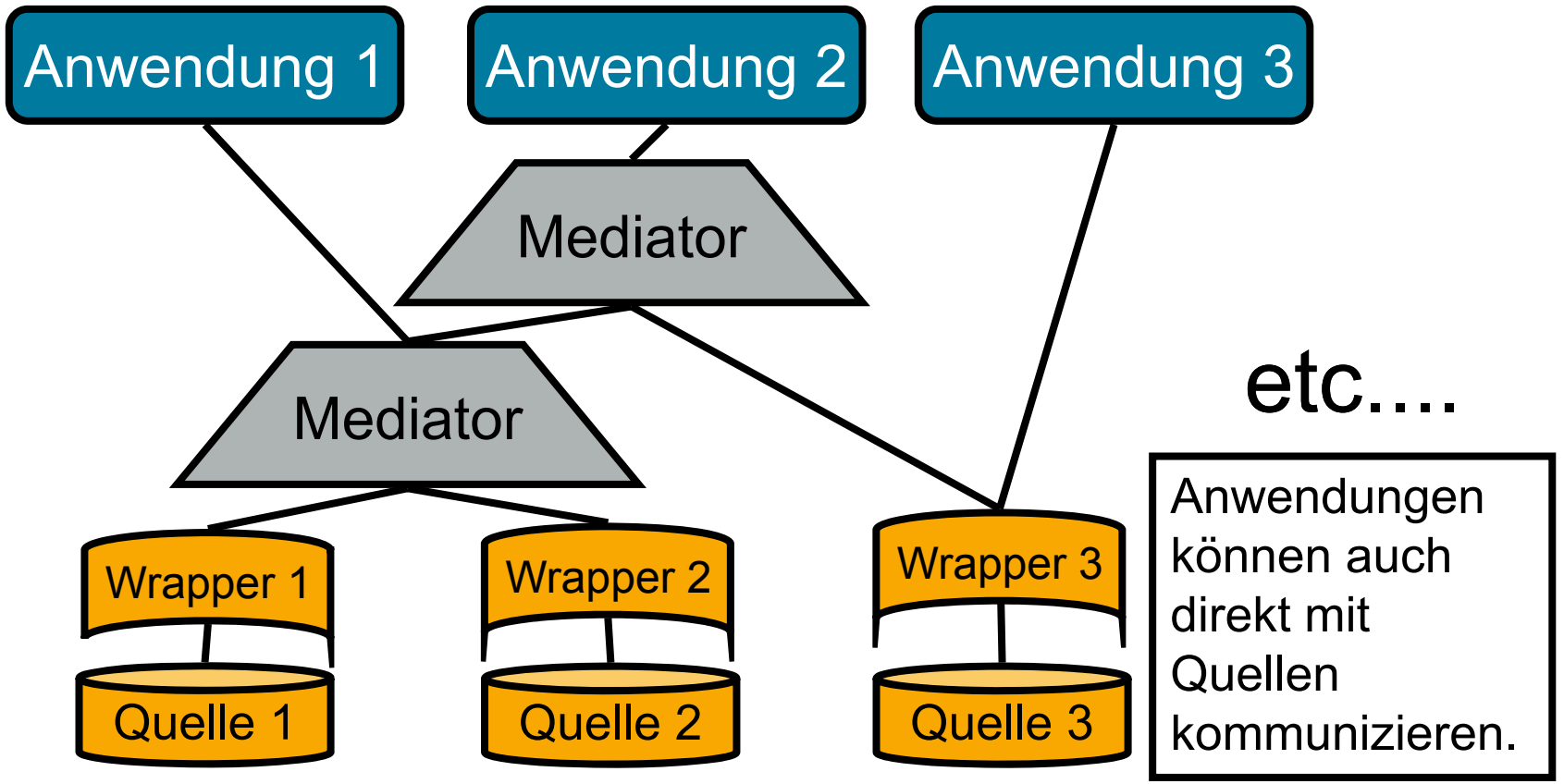
Mediator-Wrapper Architektur



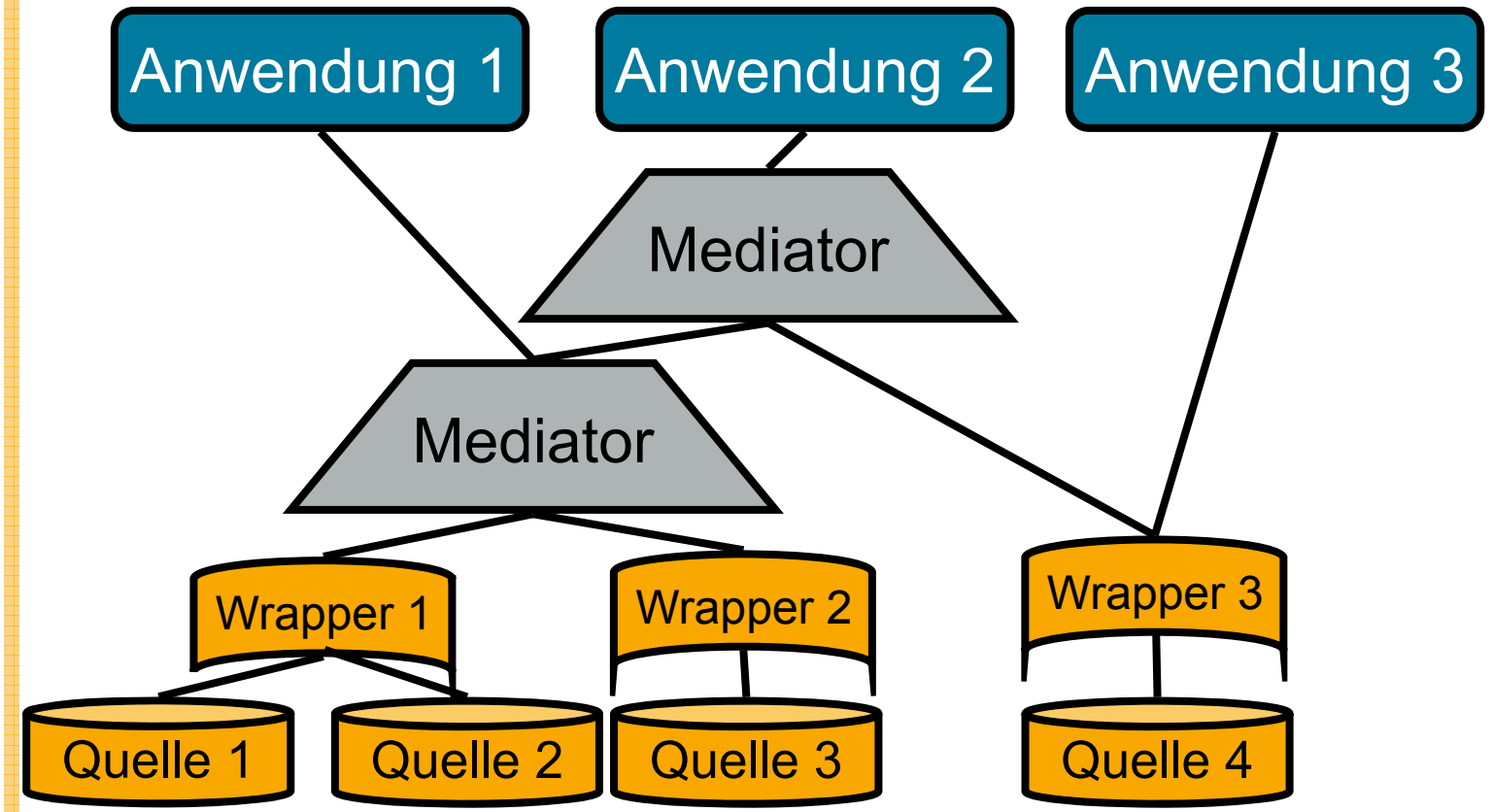
Mediator-Wrapper Architektur



Mediator-Wrapper Architektur



Mediator-Wrapper Architektur



Überblick

- Überblick über Informationssysteme
  - Klassifikation
  - Weitere Kriterien
- Architekturen
  - 3 Schichten Architektur
  - 4 Schichten Architektur
  - 5 Schichten Architektur

- Mediator-Wrapper Architektur
  - Gio Wiederholds Definitionen
  - Konfigurationen
  - Mediatoren
  - Wrapper
- Peer-Data-Management
  - Architektur
  - Anwendungen

Einfache Mediatoren

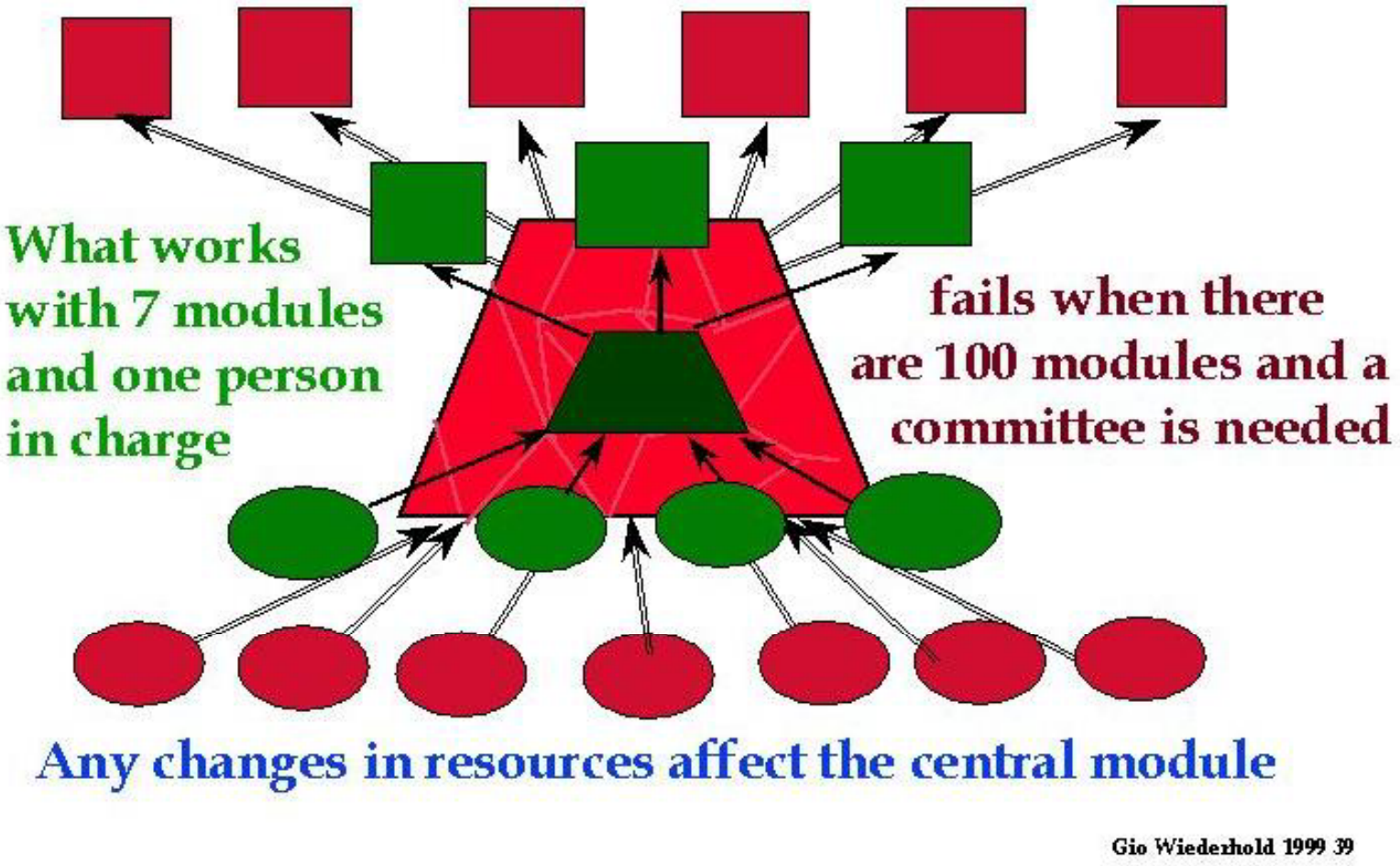
„[A mediator] should be small and simple, so that it can be maintained by one expert or, at most, a small and coherent group of experts.“ Wiederhold `92

Ein Mediator sollte klein und einfach genug sein, um durch einen einzigen oder höchstens eine kleine Gruppe von Experten gewartet werden zu können.

D.h.: Einfaches föderiertes Schema, begrenzte Domäne, einfache Schnittstellen

Erfahrung: Suchmaschinen ändern wöchentlich ihre Schnittstelle

Einfache Mediatoren



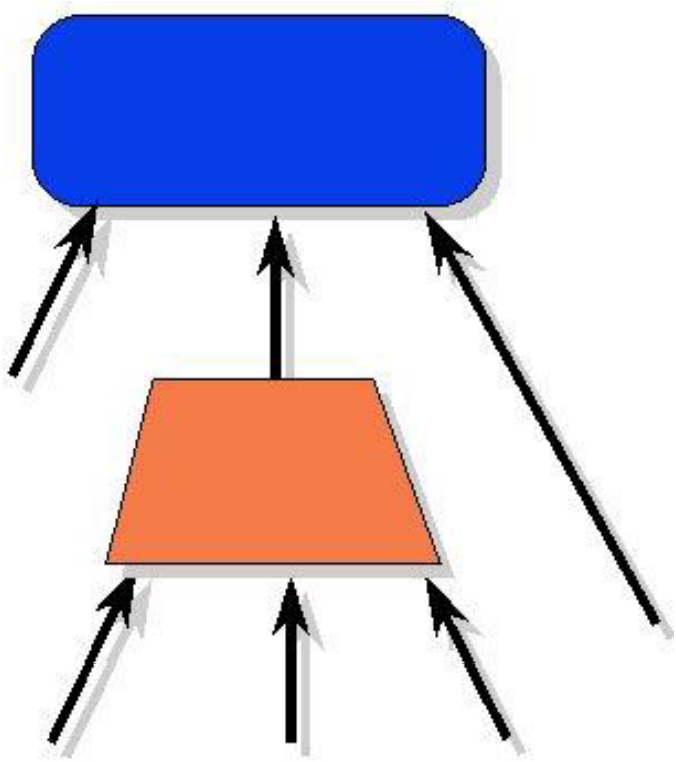
Integration mit Mediatoren

Application

- Informal, pragmatic
- Client-control
- Use up to 7±2 mediators

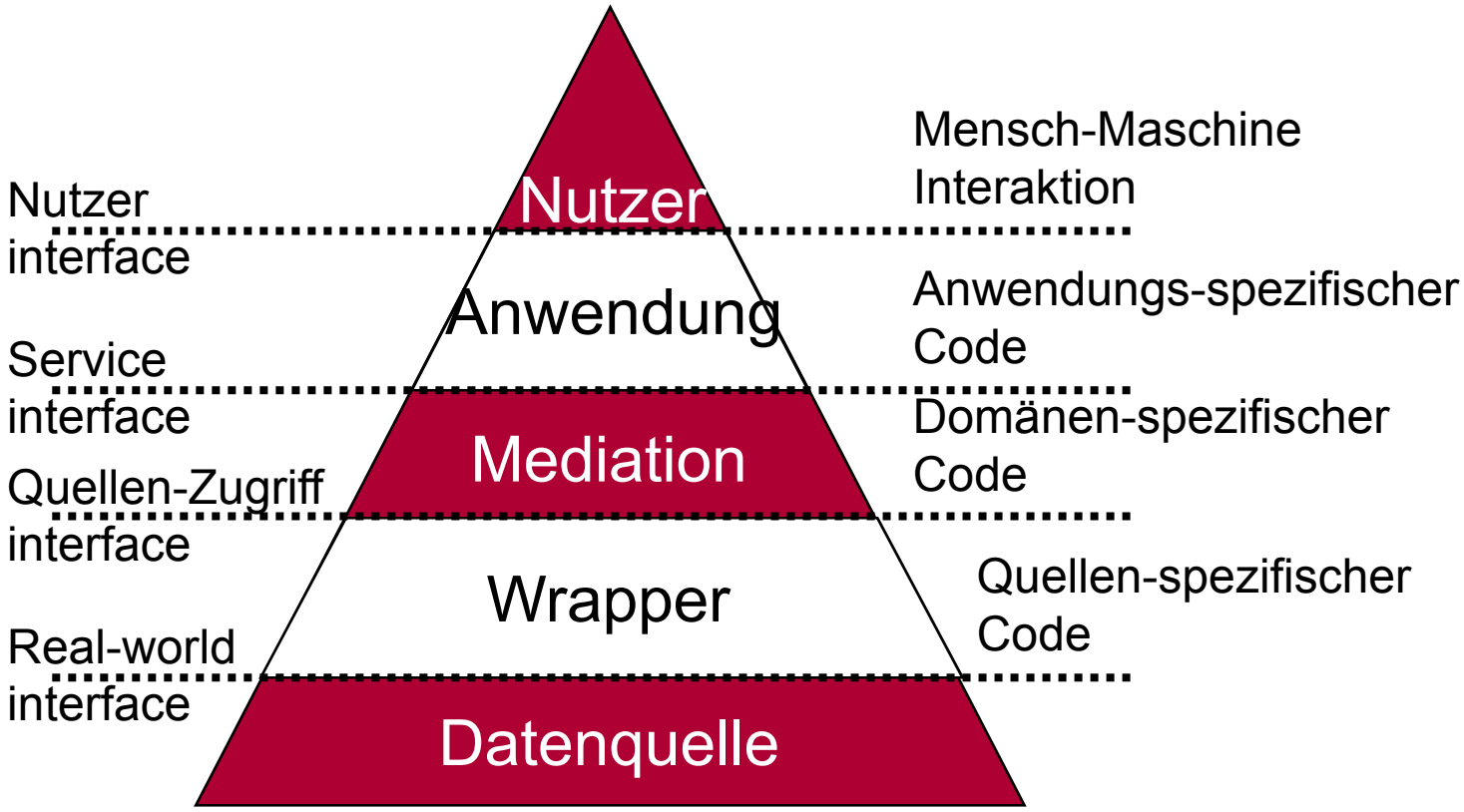
Mediation

- Formal, reliable service
- Domain-Expert control
- Use up to 7±2 sources

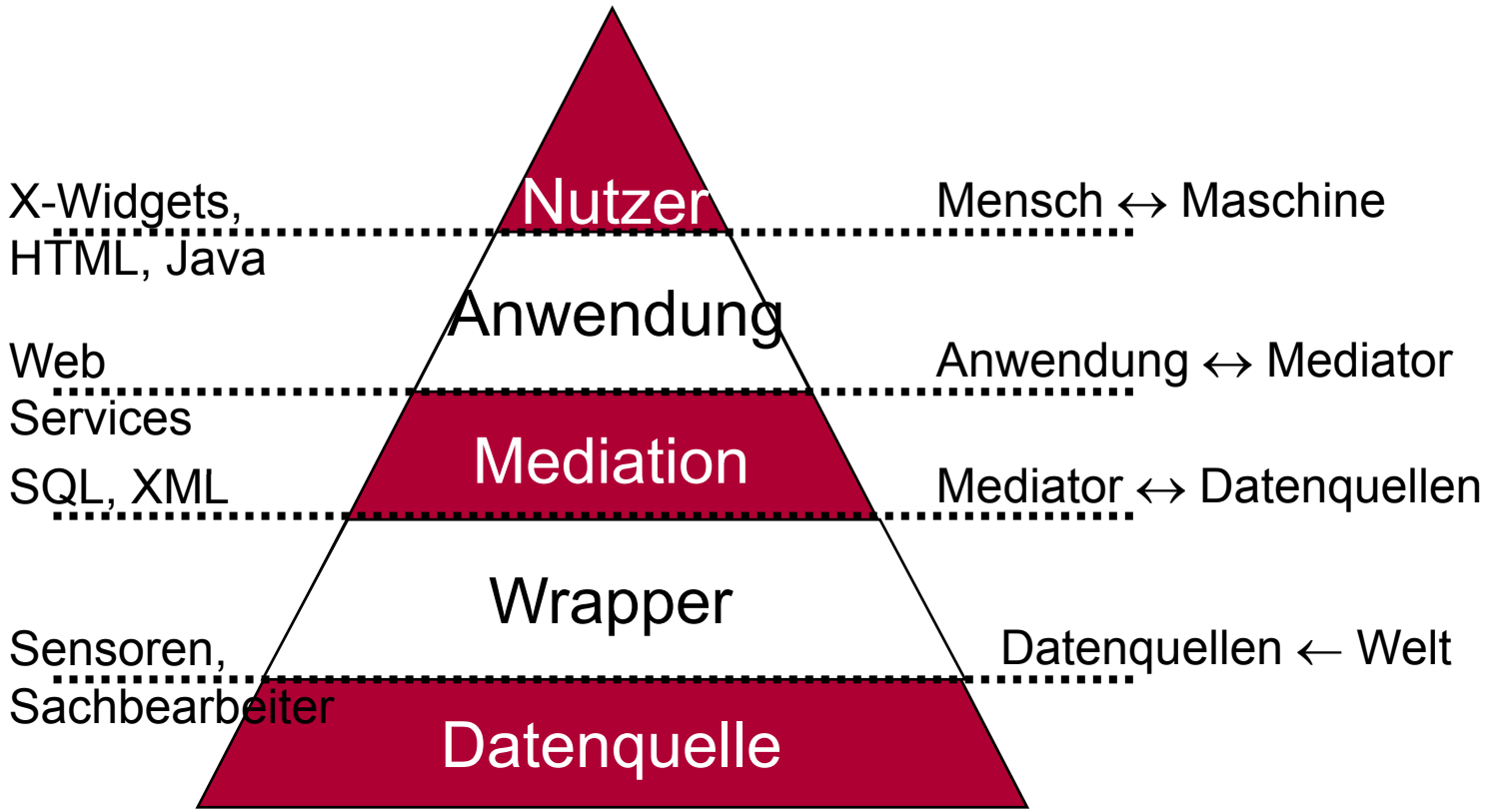


Gio Wiederhold 1999 41

Funktionale Schichten



Schnittstellen



Funktionen der Mediation

- Erbracht durch Domänen-Experten
  - Suche und Auswahl relevanter Informationsquellen
    - „Source selection“
  - Transformationen zur Konsistenzerhaltung
  - Metadaten zur Verarbeitung
  - Abstraktion zum Verständnis
  - Integration verschiedener Quellen
  - Zusammenfassung zur Präsentation
- All dies transformiert Daten zu Informationen.

Mehrwert durch Mediatoren

- **On-line presentation services,** appropriate for audience
- **Integration of documents and figures** for WWW access
- **Abstraction services:** summaries of papers, reports (with references to base mat.)
- **Review services** over suppliers, technologies, services
- **Alternative ranking** of suppliers, parts, materials, . . .

Gio Wiederhold 1999 60

Dicke und Dünne Mediatoren

Überblick

- Überblick über Informationssysteme
  - Klassifikation
  - Weitere Kriterien
- Architekturen
  - 3 Schichten Architektur
  - 4 Schichten Architektur
  - 5 Schichten Architektur

- Mediator-Wrapper Architektur
  - Gio Wiederholds Definitionen
  - Konfigurationen
  - Mediatoren
  - Wrapper
- Peer-Data-Management
  - Architektur
  - Anwendungen

Wrapper

- Wrapper sind Softwarekomponenten, die die Kommunikation und den Datenfluss zwischen Mediatoren und Datenquellen herstellen.
- Wrapper sind jeweils spezialisiert auf eine Ausprägung autonomer, heterogener Quellen.
- Wrapper vermitteln zwischen Mediator und Quelle.

Wrapper – Aufgaben

- Lösen Schnittstellenheterogenität
  - technisch
  - SQL, HTML Formulare, http, CORBA, ...
  - Mächtigkeit der Anfragesprache
- Lösen Datenmodellheterogenität
- Lösen schematische Heterogenität
  - Liefern globales Schema
- Reduzieren Anzahl der Datenmodelle (mit denen das IIS umgehen muss)
- Reduzieren Anzahl der Schemata
- Unterstützen globale Optimierung
  - Kostenmodell
  - Anfragefähigkeiten

Wrapper – Anforderungen

- Sollten schnell implementiert werden können
  - (< 1 Woche)
- Sollten wiederverwendbar sein
- Lokale Wartung (bei föderierten Systemen)

- An den Wrappern scheitern viele Projekte!
  - Deshalb Forschung zur schnellen oder sogar automatischen Wrappergenerierung.
  - Wrapperbibliotheken

Garlic Wrapper Generierung nach [TS97]

Praktische Anforderungen aus [TS97]

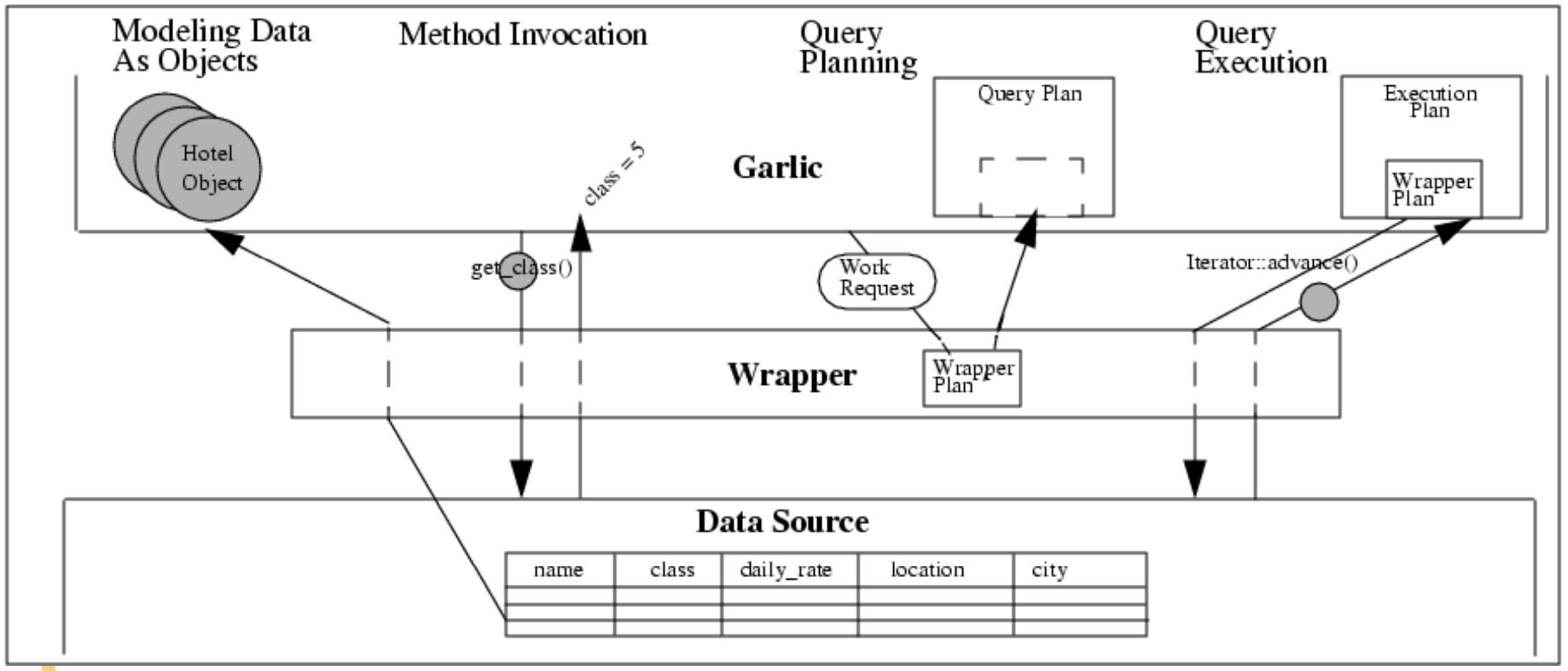
- Start-up Kosten gering (Stunden)
- Erweiterbarkeit
  - Einfacher Start
  - Später Fähigkeiten der Quellen hinzufügen
- Flexibilität
  - Möglichst breites Spektrum an Quellen abdecken
  - Neue Quellen stören Architektur nicht.
- Optimierung
  - Nicht durch Autoren sondern durch Garlic

Garlic Wrapper Generierung

Vier Grund-Services

1. Modellierung und Zugriff auf die Daten in der Quelle
2. Aufruf von Methoden in der Quelle
3. Mithilfe bei der Anfrageplanung
4. Anfrageausführung

Garlic Wrapper Generierung



Quelle: [TS97]

Garlic Wrapper Generierung

Modellierung und Zugriff auf die Daten

- Garlic nutzt OO Modell
- Wrapper stellt Daten als Objekte mit Interface (globales Schema) und Implementierung (lokales Schema) dar.
- Stellt Identität von Objekten her.

Garlic Wrapper Generierung

Aufruf von Methoden in der Quelle

- Implizit immer: Get_attr() für jedes Attribut
- Implizit immer: Set_attr() für jedes nicht-read-only Attribut
- Um auch Quellen abzudecken, die nur über Methoden zu erreichen sind.
- Um besondere Fähigkeiten von Quellen auszuschöpfen.
- Beispiel: display_Image(ImageID)

Garlic Wrapper Generierung

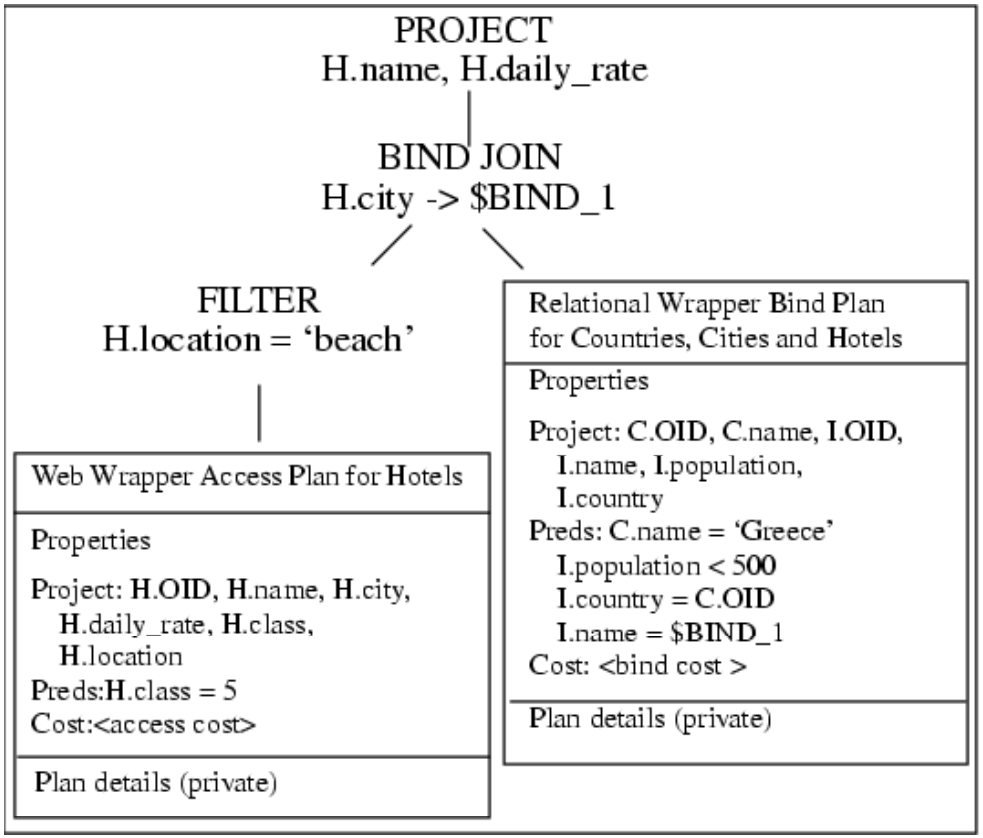
Mithilfe bei der Anfrageplanung (query planning)

- Garlics Anfrageplanung betrachtet alternative Pläne und sucht den besten heraus.
  - Kostenbasiert
- Mediator verschickt „Teilaufgaben" an Wrapper.
  - Wrapper kann Teile davon ablehnen (je nach Fähigkeiten der Quelle).
  - Mediator gleicht aus.
    - Preprocessing
    - Postprocessing
- Wrapper liefert null oder mehr Teilpläne zurück.
- Teilpläne werden in Gesamtplan eingebaut.

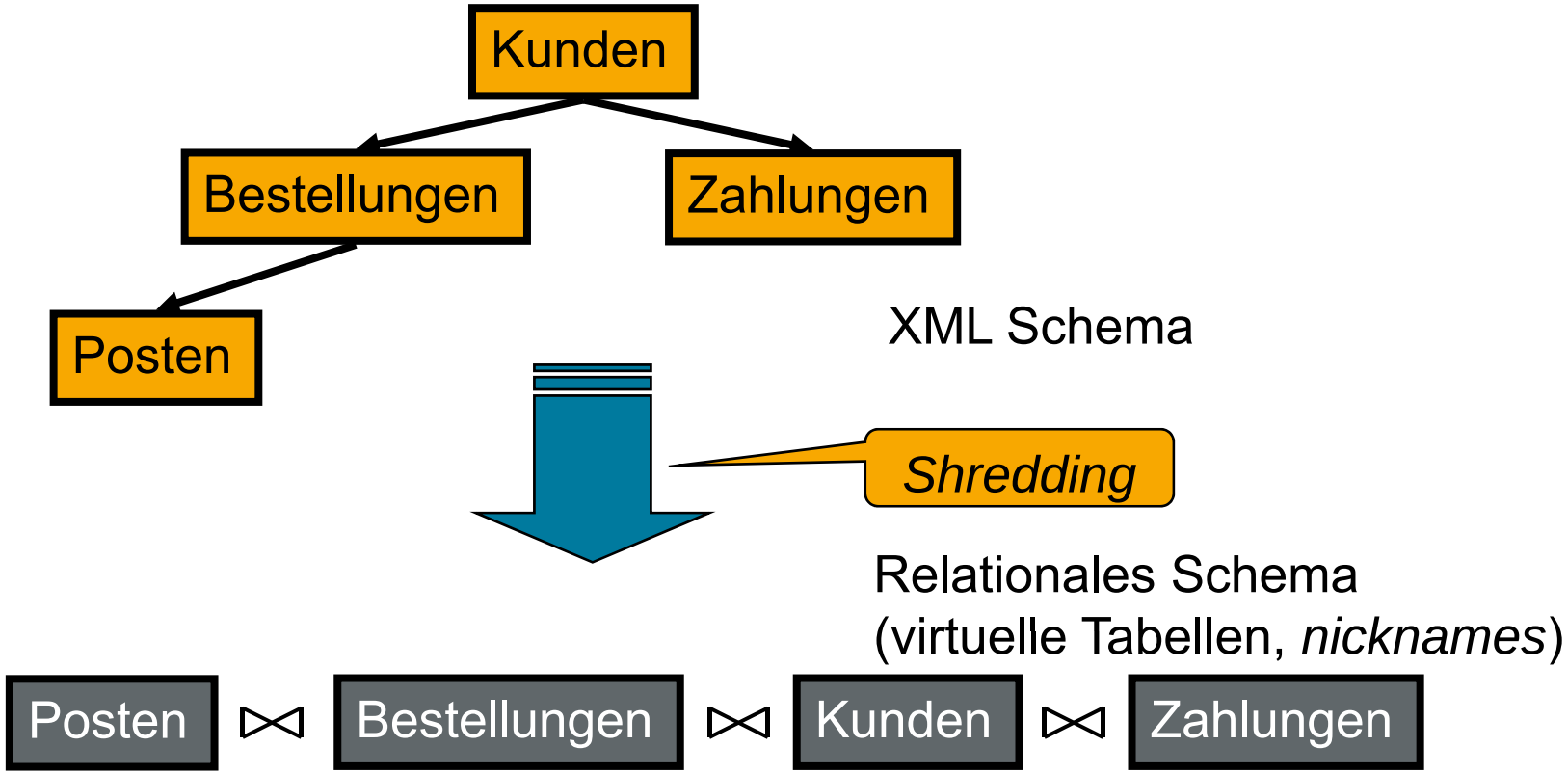
Garlic Wrapper Generierung

Anfrageausführung (query execution)

- Mediator produziert Operatorbaum.
- Wrapper-Teilpläne sind Blätter in dem Baum.
- Pläne werden in Iteratoren umgewandelt
- Pipelining



Beispiel: XML Wrapper für DB2



Beispiel: XML Wrapper für DB2


```
CREATE NICKNAME kunden_NN(
name            VARCHAR(48) OPTIONS(XPATH './name/text()'),
addresse        VARCHAR(48) OPTIONS(XPATH './address/text()'),
kunden NN ID VARCHAR(48) OPTIONS(PRIMARY KEY 'YES'))
FOR SERVER xml_server
  OPTIONS(XPATH '//customer', FILE_PATH ‘customers.xml');
```

```
CREATE NICKNAME order_NN(
amount          DOUBLE      OPTIONS(XPATH './amount/text()'),
date            VARCHAR(48) OPTIONS(XPATH './date/text()'),
order_NN_ID     VARCHAR(48) OPTIONS(PRIMARY_KEY 'YES'),
customer_NN_FID VARCHAR(48) OPTIONS(FOREIGN_KEY 'CUSTOMER_NN'))
FOR SERVER xml_server OPTIONS(XPATH './/order');
```

```
CREATE NICKNAME item_NN(
name          VARCHAR(48) OPTIONS(XPATH './name/text()'),
quant         INTEGER     OPTIONS(XPATH './quant/text()'),
order_NN_FID  VARCHAR(48) OPTIONS(FOREIGN_KEY 'ORDER_NN'))
FOR SERVER xml_server OPTIONS(XPATH './/item');
```

```
CREATE NICKNAME payment_NN(
amount          INTEGER     OPTIONS(XPATH './amount/text()'),
date            VARCHAR(48) OPTIONS(XPATH './date/text()'),
customer_NN_FID VARCHAR(48) OPTIONS(FOREIGN_KEY 'CUSTOMER_NN'))
FOR SERVER xml_server OPTIONS(XPATH './/payment');
```

Firmen, die Mediatoren und Wrapper einsetzen

- BEA systems
- CA (Computer Associates)
  - Product: OPAL; Specialty: Screenscraper, extract and integratate output without an API.
- Enosys: XML-based data integration
- Genelogic: genomics information in object form
- DiscoveryLink / Information Integrator
- MetaMatrix: Enterprise Content Integration,
  - eCommerce infrastructure software to manage the metadata of disparate data repositories and to provide uniform access to these information silos.
- Nimble Technnology: XML-based data integration
- From: http://www-db.stanford.edu/LIC/companies.html

Forschungsprojekte

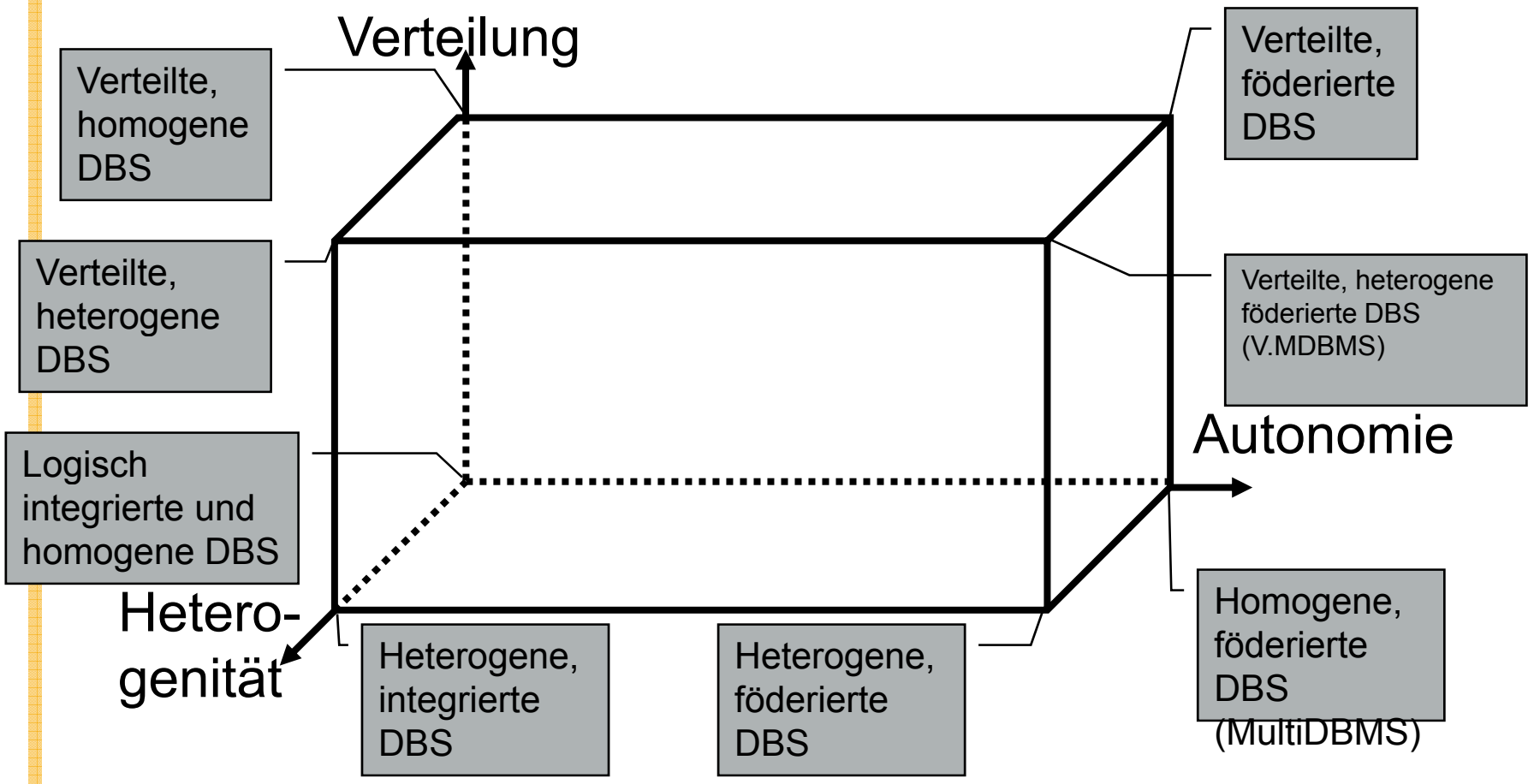
- Carnot
- CoBase
- COIN
- Garlic
- Harvest Information Discovery and Access System
- HERMES
- Info*
- Infomaster
- Information Manifold
- INFOSLEUTH
- OBSERVER
- SIMS
- SKC
- Tsimmis

Überblick

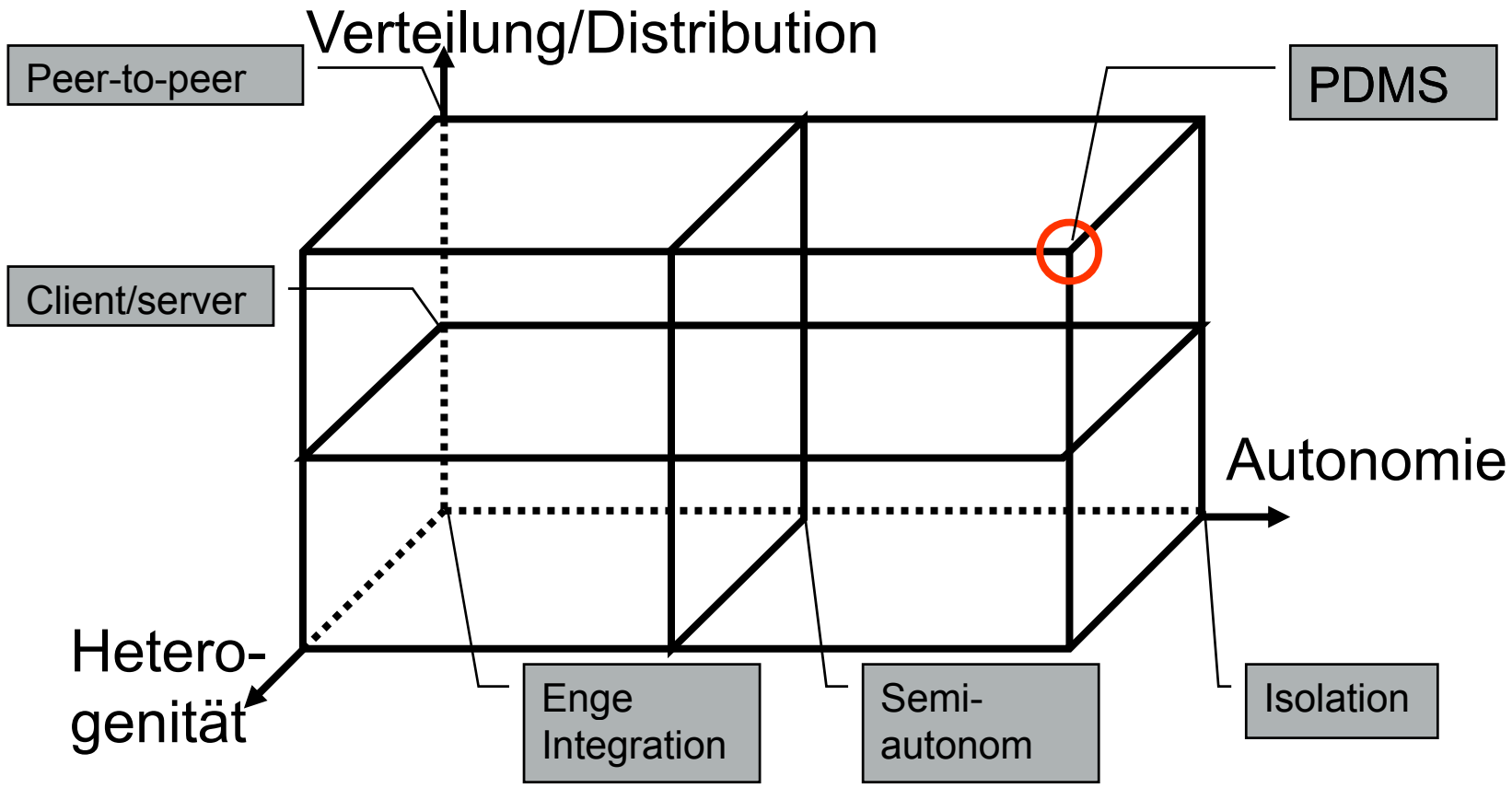
- Überblick über Informationssysteme
  - Klassifikation
  - Weitere Kriterien
- Architekturen
  - 3 Schichten Architektur
  - 4 Schichten Architektur
  - 5 Schichten Architektur

- Mediator-Wrapper Architektur
  - Gio Wiederholds Definitionen
  - Konfigurationen
  - Mediatoren
  - Wrapper
- Peer-Data-Management
  - Architektur
  - Anwendungen

Wdh: Klassifikation von Informationssystemen nach [ÖV91]



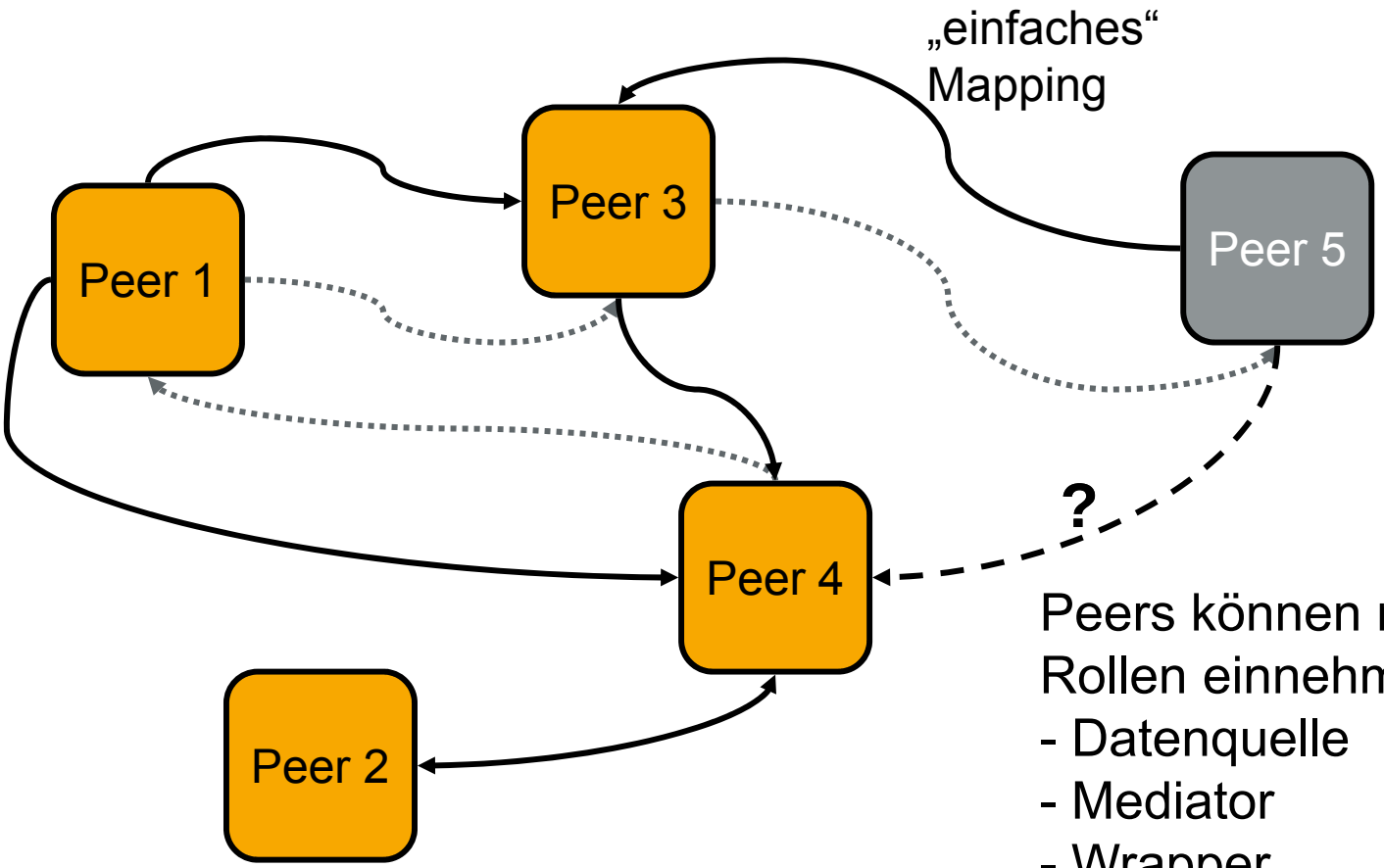
Wdh: Erweiterung der Klassifikation nach [ÖV99]



PDMS – Idee

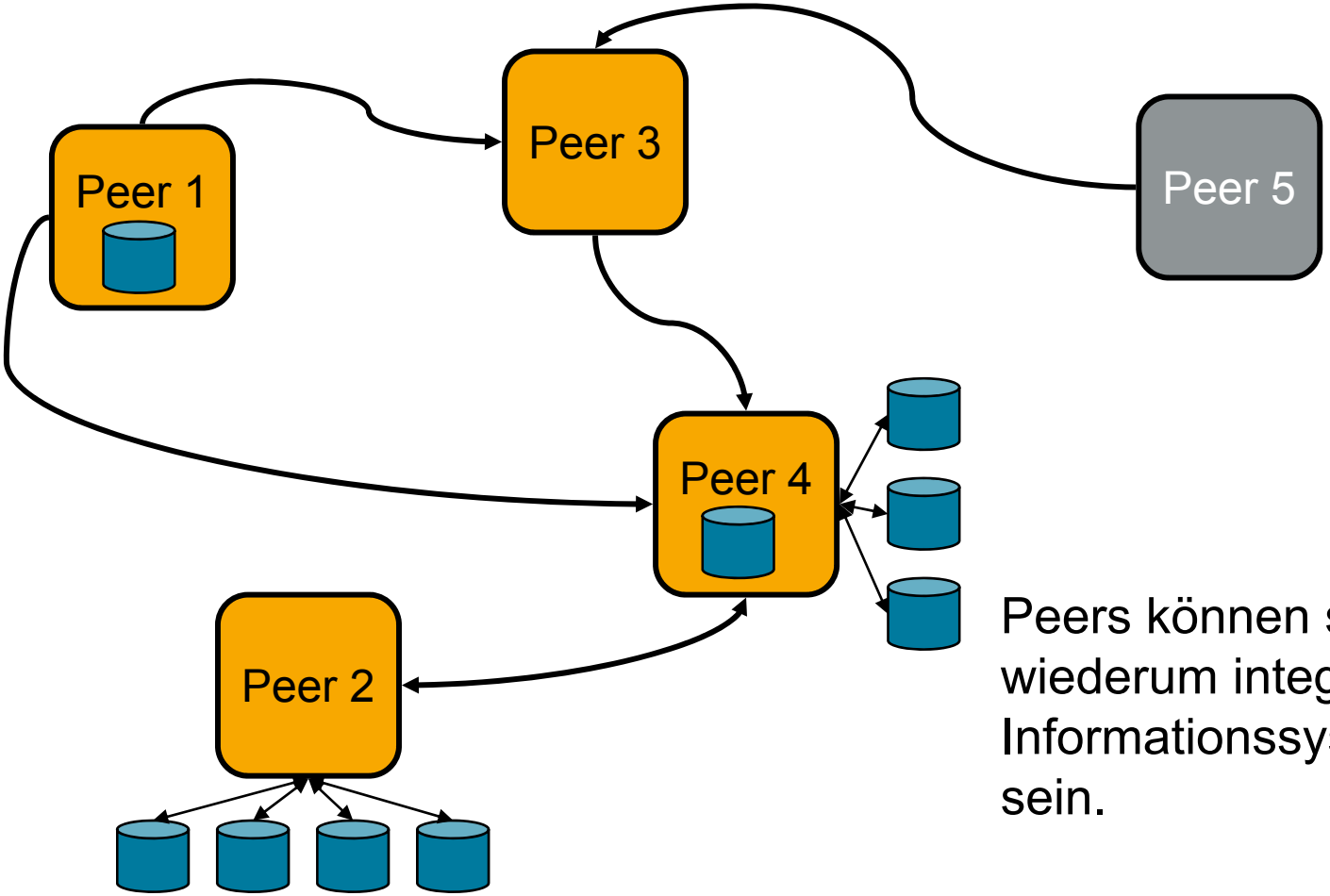
- Idee: Peer Netzwerk (P2P)
  - [HIST03], [HIMT03], [BGK+02]
- Jeder Peer kann
  - Daten exportieren (= Datenquelle)
  - Sichten auf Daten zur Verfügung stellen (= Wrapper)
  - Anfragen anderer Peers entgegennehmen und weiterleiten (= Mediator)
  - Anfrage stellen
- Verknüpfungen nicht zwischen lokalen und globalem Schema, sondern zwischen Paaren von Peers.

Peer-Data-Management Systeme (PDMS)



Peers können mehrere Rollen einnehmen:
- Datenquelle
- Mediator
- Wrapper
- Anfrager

Peer-Data-Management Systeme (PDMS)



Peers können selbst wiederum integrierte Informationssysteme sein.

PDMS Architektur (Piazza)

- Overlay Netzwerk aus „Peers“, verbunden über Internet
- Relationales oder XML Datenmodell
- Jeder Peer kann bereitstellen:
  - Daten (materialisiert)
  - Ein (oder mehr) Schemas
  - Mappings
- Jeder Peer kann anbieten
  - Anfragebearbeitung (für eigenes oder fremdes Schema)
  - Materialisierung
  - Metadaten zur Koordination

PDMS vs. P2P Dateiaustausch

- P2P
  1. Nur ganze Dateien (niedrige Granularität)
  2. Einfachste Anfragen
     - Dateinamen, Keywords
  3. Unvollständige Anfrageergebnisse
  4. Einfaches Schema
  5. Hoch dynamisch
  6. Millionen Peers
  7. Datenübertragung direkt

- PDMS
  1. Objekte (hohe Granularität)
  2. Komplexe Anfragen
     - Anfragesprache (SQL, etc.)
  3. Vollständige Anfrageergebnisse (zumindest erwartet)
  4. Schema
  5. Kontrollierte Dynamik
  6. Zig peers
  7. Datenübertragung entlang des Mapping-Pfads

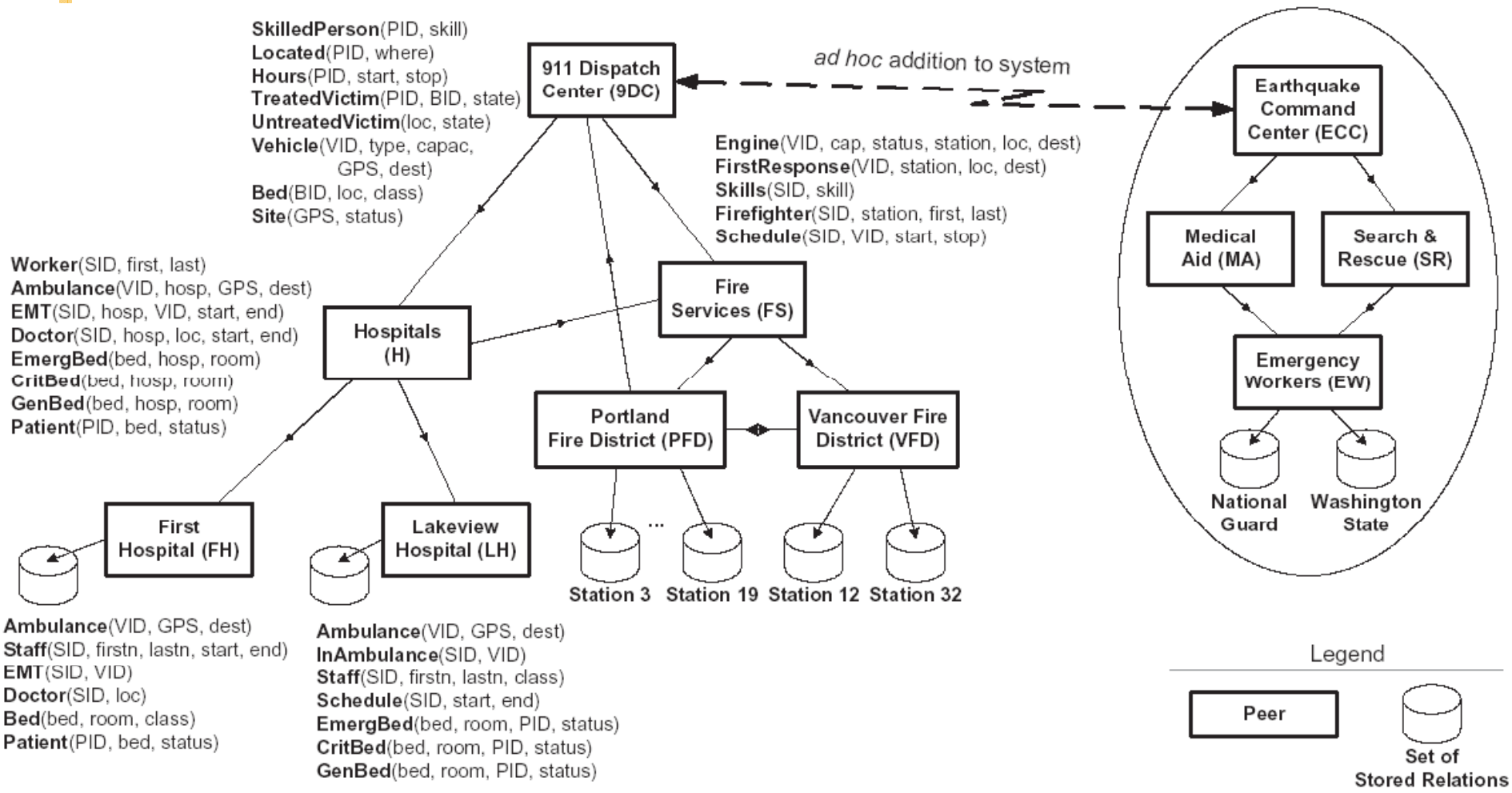
PDMS vs. FDBMS

- Vorteile
  - Nutzer/Anwendungen müssen nur das eigene Schema kennen.
  - Alle Daten sind erreichbar (über die transitive Hülle der Mappings).
  - Neue Schemata und Peers können leicht hinzugefügt werden (inkrementell).
  - Mappings nur zu ähnlichsten Schemata
- Nachteile / Probleme
  - Mapping-Erstellung ist schwierig.
  - Mapping Komposition ist schwierig.
  - Viele Mappingschritte durch das Netzwerk:
    - Effizienz
    - Skalierbarkeit
    - Datenqualität
  - Effiziente Platzierung von Daten?
  - Updates?

PDMS Anwendungen

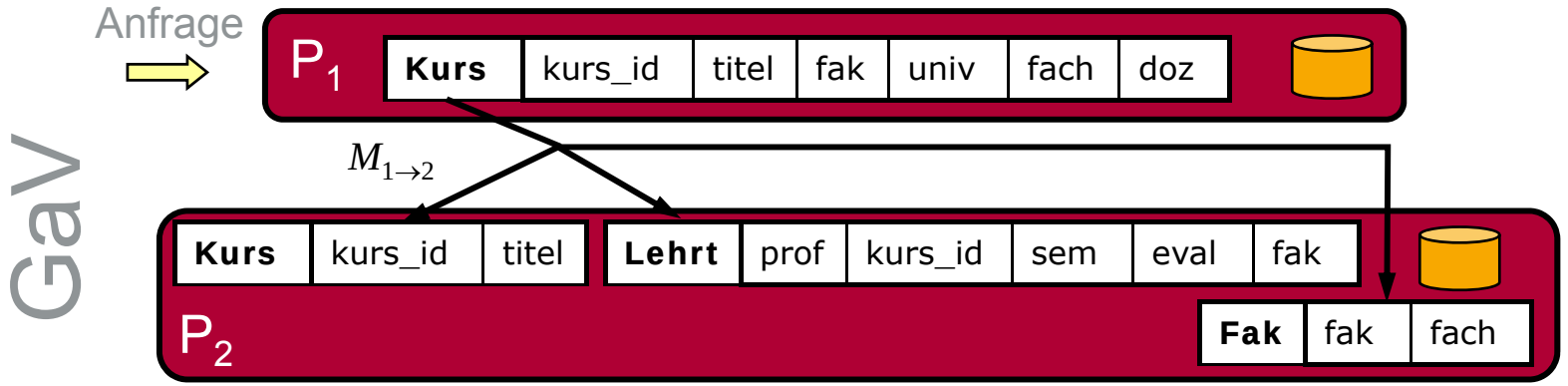
- Gesundheitsinformationssystem
  - Krankenhausdaten auf vielen Systemen verteilt
  - Ärzte wollen manche Daten verbreiten, andere nicht.
  - Content-management-artige Suche ist wichtig.
  - Verschiedenste und komplexe Schemata
  - Mehrwert (für Patienten) durch Teilen der Daten
- Genomdaten
  - Forscher haben den Willen (und die Pflicht), Daten weltweit zu veröffentlichen.
  - Komplexe Schemata und komplexe Anfragen
  - Bekannte Zusammenhänge zwischen den Daten
  - Bildung eines globalen Schemas nicht immer einfach
- Automobil-Industrie
- Katastrophen-Management

Piazza – Beispiel



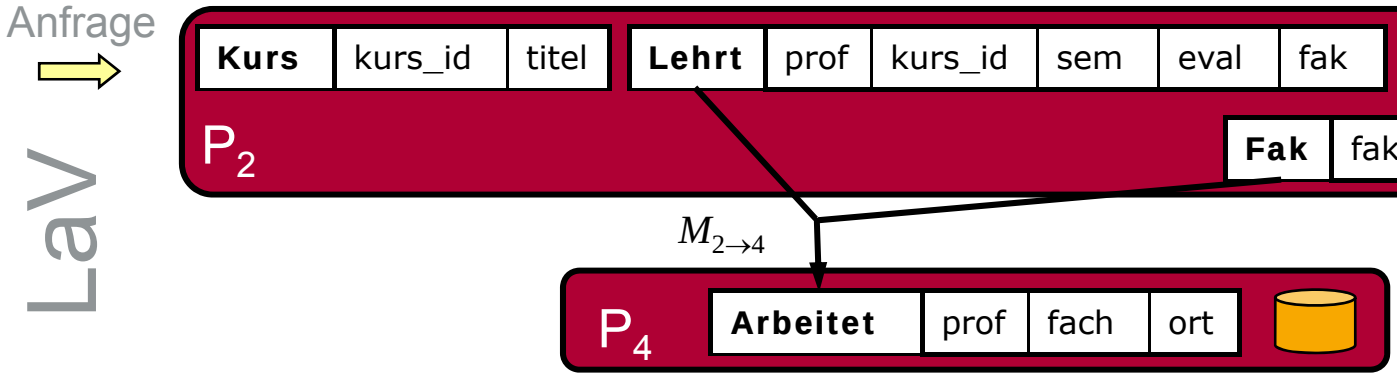
GaV und LaV Mappings

GaV



$M_{1\to 2}$ **P2.Kurs**(kurs_id, titel), **P2.Lehrt**(prof, kurs_id, sem, eval, fak), P2.Fak(fak, fach) $\subseteq$ **P1.Kurs**(kurs_id, titel, fak, univ, fach, doz)

LaV

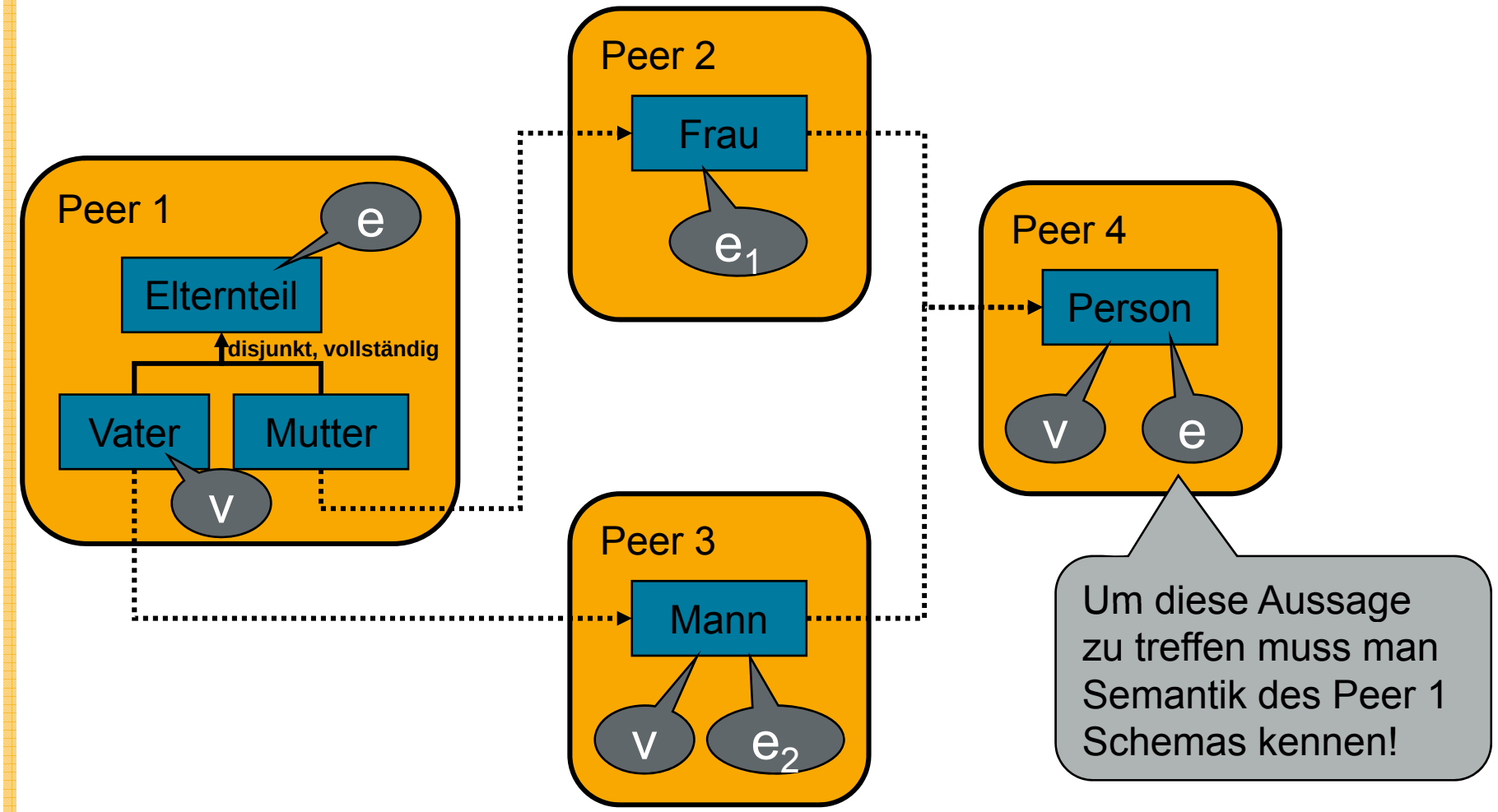


$M_{2\to 4}$ **P4.Arbeitet**(prof, fach, ort) $\subseteq$ **P2.Lehrt**(prof, kurs_id, sem, eval, fak), **P2.Fak**(fak, fach)

PDMS – Diskussion

- Vorteile
  - Nutzer müssen nur eigenes Schema kennen.
  - Dennoch sind alle Daten (über transitive Hülle der Mappings) verfügbar
  - Neue Schemas können leicht und inkrementell hinzugefügt werden.
  - Mapping nur zum ähnlichsten Schema nötig.
- Nachteile/Probleme
  - Mappings zwischen Schemas nötig
    - Aber: Mappings automatisch erstellen (Schema Matching)
  - Mapping Komposition
  - Effizienz (bei vielen Zwischenstationen)
  - Effiziente Datenverteilung
  - Read-only oder Updates?
  - Außerdem:
    - Verlust der Semantik
    - Verlust an Informationsqualität, z.B. Vollständigkeit

Semantik in PDMS [Len04]

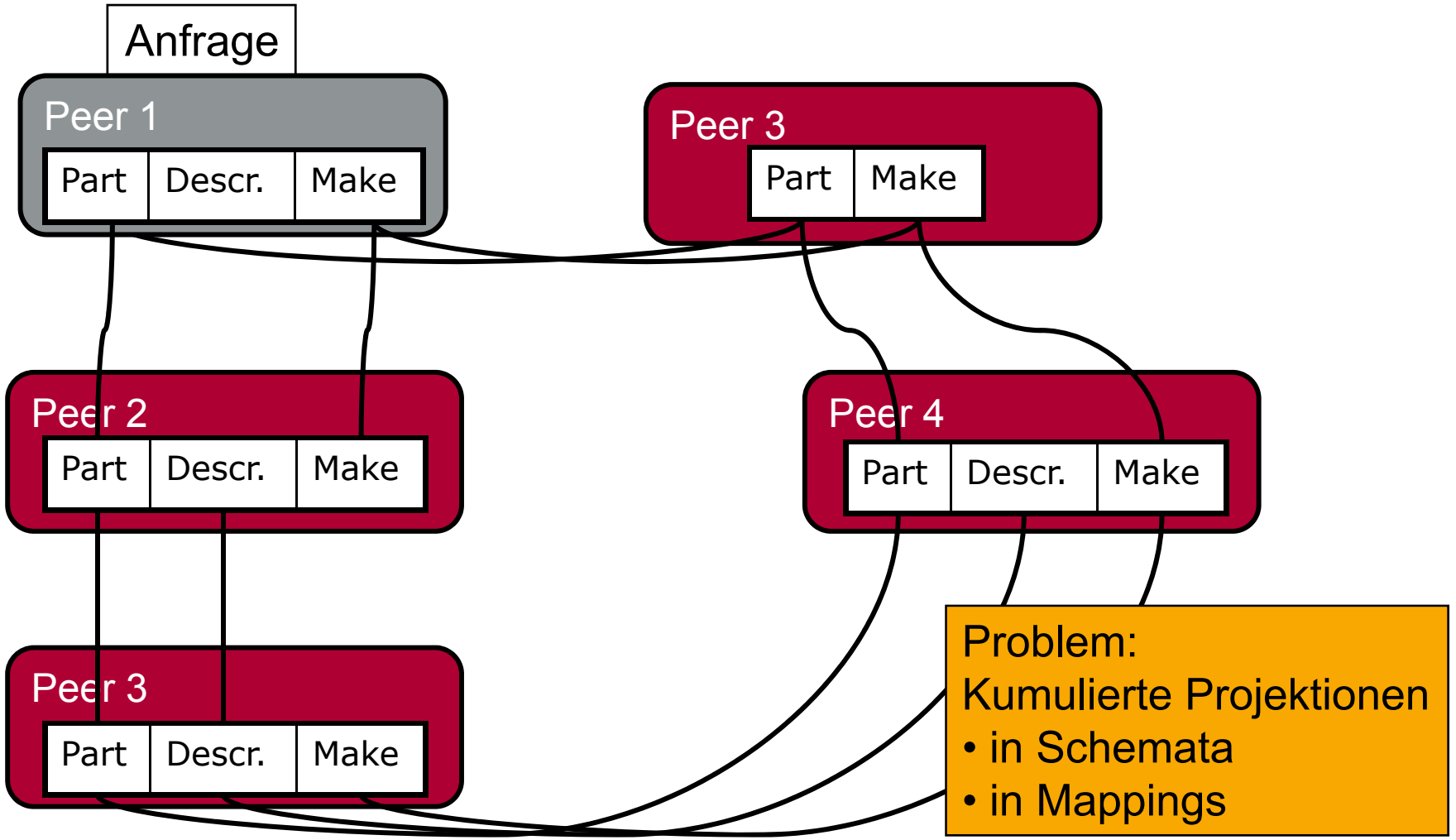


Vollständigkeit in PDMS

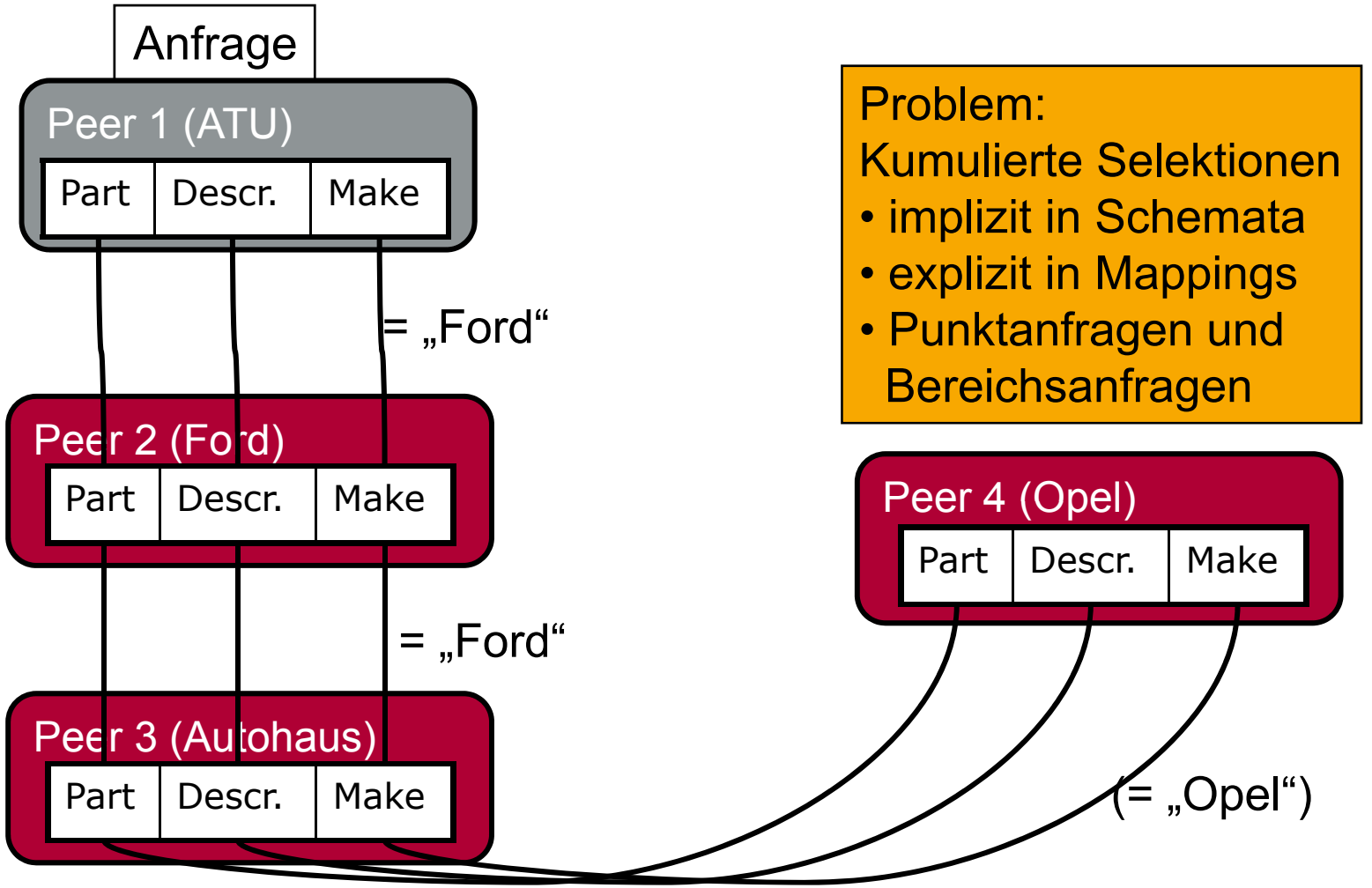
- Einsicht: Vollständiges Ergebnis ist zu teuer.
  - Also neues Optimierungsziel: Vollständigkeit
  - Extensionale Vollständigkeit: Viele Tupel
    - Metadaten: *Kardinalitäten*
  - Intensionale Vollständigkeit: Viele Nicht-null-Werte
    - Metadaten: *Werteverteilungen*
  - Projektionen und Selektionen in Mappings führen zu Informationsverlust.
    - Metadaten: *Schema Mappings*

- Idee: Gezieltes Beschneiden des Anfrageplans (*Pruning*)
    - Schnitt bei vielen Projektionen
    - Schnitt bei starken Selektionen

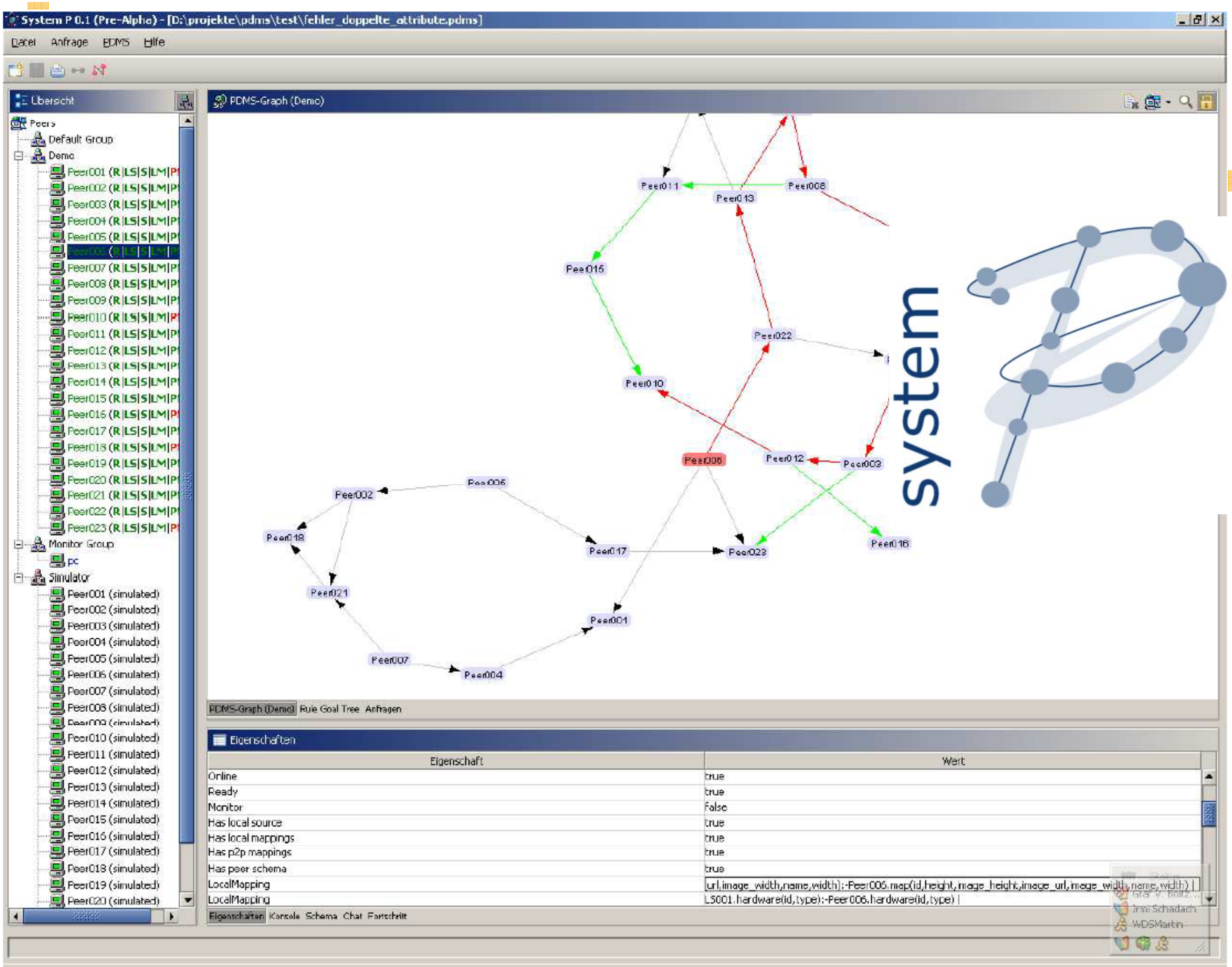
Unvollständige Mappings



Selektive Mappings



Problem:
Kumulierte Selektionen
- implizit in Schemata
- explizit in Mappings
- Punktanfragen und Bereichsanfragen


System P 0.1 (Pre-Alpha) - [D:\projekte\pdms\test\fehler_doppelte_attribute.pdms]
Datei Anfrage PDMS Hilfe
Übersicht
Peers
Default Group
Demo
Monitor Group
pc
Simulator
PDMS-Graph (Demo)
Peer011
Peer013
Peer008
Peer015
Peer022
Peer010
Peer006
Peer012
Peer009
Peer002
Peer005
Peer018
Peer017
Peer023
Peer016
Peer021
Peer001
Peer007
Peer004
system
PDMS-Graph (Demo) Rule Goal Tree Anfragen
Eigenschaften
Eigenschaft
Wert
Online
true
Ready
true
Monitor
false
Has local source
true
Has local mappings
true
Has p2p mappings
true
Has peer schema
true
LocalMapping
LocalMapping
Eigenschaften Konsole Schema Chat Fortschritt

Literatur

Wichtige Literatur

- [ÖV99] Principles of Distributed Database Systems. M. Tamer Özsu, Patrick Valduriez, Prentice Hall, 1999.
- [SL90] Amit P. Sheth and James A. Larson, Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases, ACM Computing Surveys, Vol. 22(3), pp183-236, 1990.
- [Con97] Stefan Conrad, Föderierte Datenbanksysteme. Springer, Heidelberg 1997.
- [HIST03] Alon Y. Halevy, Zachary G. Ives, Dan Suciu, Igor Tatarinov. Schema Mediation in Peer Data Management Systems, ICDE conference 2003
- [Wie92] Mediators in the Architecture of Future Information Systems, Gio Wiederhold, IEEE Computer Journal, 25(3), 38-49, 1992

Weitere Literatur zu Architekturen

- [LMR90] W. Litwin, L. Mark, N. Roussoupoulos, Interoperability of Multiple Autonomous Databases, ACM Computing Surveys, Vol. 22(3), pp267-293, 1990.
- [HM85] Dennis Heimbigner, Dennis McLeod: A Federated Architecture for Information Management. ACM Trans. Inf. Syst. 3(3): 253-278 (1985)

Literatur

Weitere Literatur zu Mediator Wrapper Architektur

- [Gar95] Michael J. Carey, Laura M. Haas, Peter M. Schwarz, Manish Arya, William F. Cody, Ronald Fagin, Myron Flickner, Allen Luniewski, Wayne Niblack, Dragutin Petkovic, Joachim Thomas II, John H. Williams, Edward L. Wimmers: Towards Heterogeneous Multimedia Information Systems: The Garlic Approach. RIDE-DOM 1995: 124-131
- [JS03] Querying XML data sources in DB2: the XML Wrapper, Vanja Josifovski and Peter Schwarz, ICDE conference, Bangalore, India, 2003
- [TS97] Don't Scrap It, Wrap It! A Wrapper Architecture for Legacy Data Sources, Mary Tork Roth and Peter Schwarz, VLDB Conference 1997 Athens, Greece, 1997.

Weitere Literatur zu PDMS

- [BGK+02] Philip A. Bernstein, Fausto Giunchiglia, Anastasios Kementsietsidis, John Mylopoulos, Luciano Serafini, Ilya Zaihrayeu: Data Management for Peer-to-Peer Computing : A Vision. WebDB 2002: 89-94
- [ÖV91] & [ÖV99] Principles of Distributed Database Systems. M. Tamer Özsu, Patrick Valduriez, Prentice Hall, 1991/1999.
- [HIMT03] Alon Y. Halevy, Zachary G. Ives, Peter Mork, Igor Tatarinov. Peer Data Management Systems: Infrastructure for the Semantic Web. WWW Conference, 2003.
- [NOTZ03] W.S.Ng, B.C. Ooi, K.L. Tan und A. Zhou: PeerDB: A P2P-based System for Distributed Data Sharing. ICDE 2003
- [Len04] Maurizio Lenzerini: Quality-aware data integration in peer-to-peer systems. Invited talk at IQIS workshop, Paris, 2004.