



**Forschungszentrum
Informatik**




**Universität
Karlsruhe (TH)**

Architekturen und Systeme zur Integration autonomer Informationsquellen



Andreas Schmidt

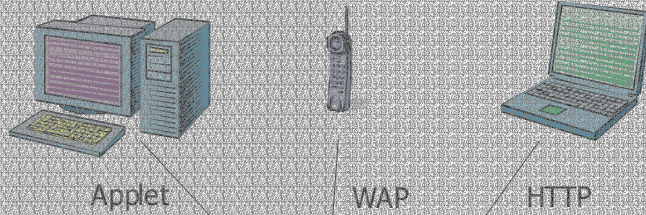
WS 2003/2004



- Integration von Diensten innerhalb eines Unternehmens mittels Middleware
 - ▶ Verteilte Objekte: CORBA
 - ▶ Applikationsserver: J2EE
- Anbindung der eigenen Systeme an das Web
 - ▶ Web-Technologien
 - ▶ XML als Technologie ohne konzeptionelle Brüche: Datenbanken und XML, Verarbeitung von XML
- Immer bestand eine relativ große Kontrolle über die zu integrierenden Systeme bzw. die Systemumgebung.
 - ▶ Wir konnten etwas verändern
 - ▶ Beteiligte Systeme relativ verlässlich (LAN)

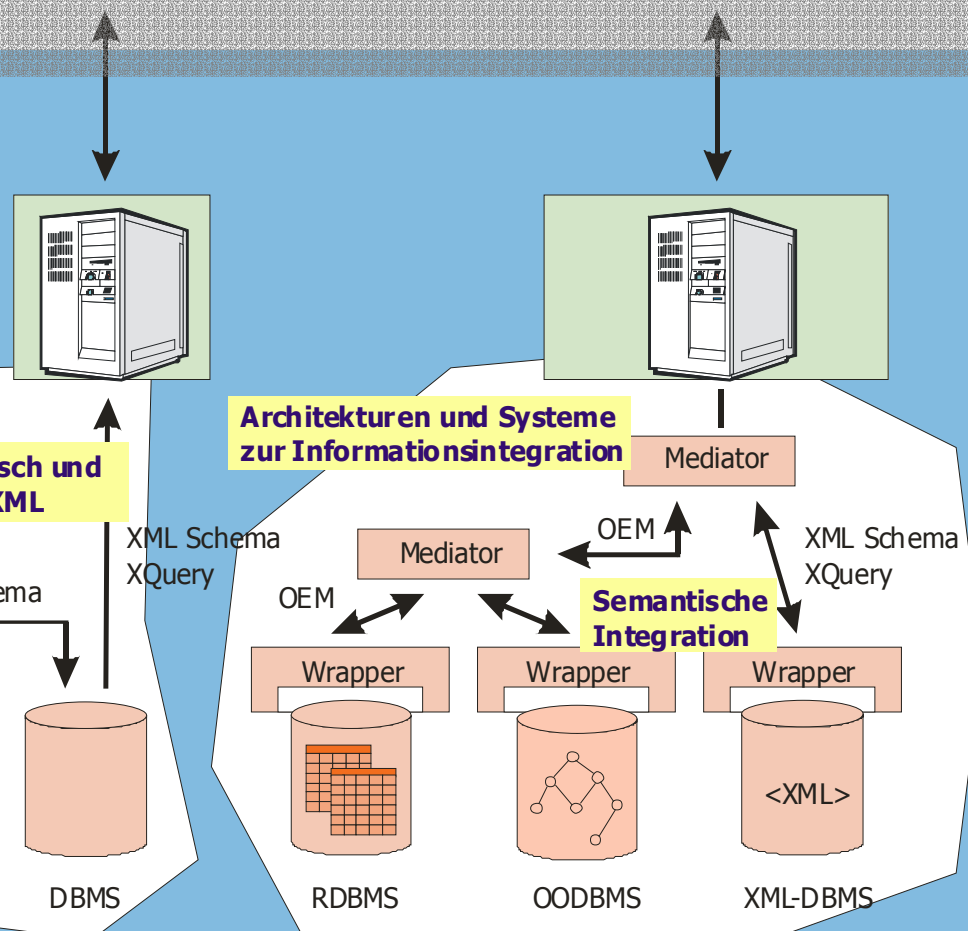
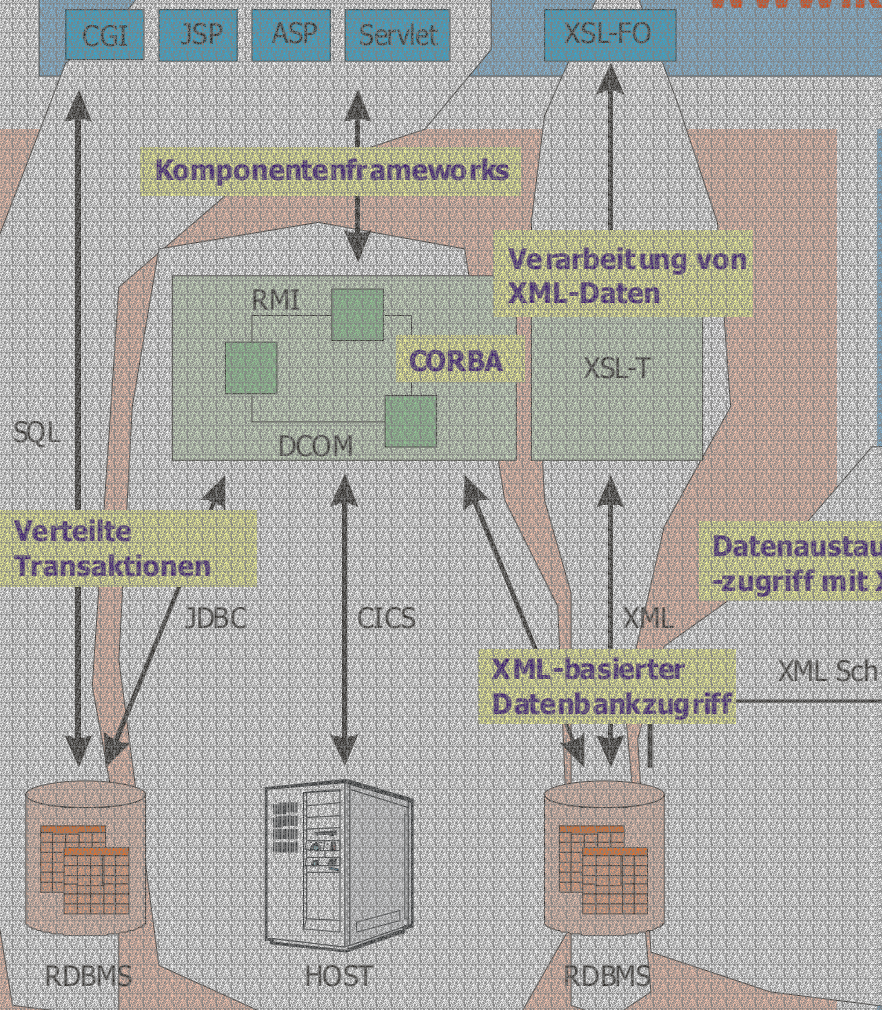
- Um das Angebot zu vergrößern und zu verbessern, sollen weitere Anbieter in das Portal mit aufgenommen werden.
 - ▶ Anbieter mit anderen Schwerpunkten
 - ▶ Verbraucherurteile über Produkte
 - ▶ Behördeninformationen zu Bauvorschriften
- Randbedingungen:
 - ▶ Die Einbindung der neuen Angebot soll integriert erfolgen, so daß sich für den Benutzer eine einheitliche Sicht ergibt.
 - ▶ Die Systeme der eingebundenen Partner bleiben autonom und können sich unabhängig vom Portal weiterentwickeln.
 - ▶ Für die Einbindung sollen keine großen Änderungen an den einzubindenden Dienste notwendig werden.

- Integrationsebenen
- Integration auf der Informationsebene
 - ▶ Materialisierte Integration
 - ▶ Virtuelle Integration (I³)
 - Referenzarchitektur
 - Semistrukturiertes Datenmodell
 - Problembereiche
 - ▶ Schemaintegration
- Fallbeispiele: Integrationsprojekte
- Weiteres Programm



Web-Technologien im Überblick

www.klick-and-bau.com



Aspekte der Autonomie

- Beschränkter Lösungsraum
 - ▶ Keine Eingriffsmöglichkeiten in Infrastruktur oder Informationsmodelle
 - ▶ Insbesondere: keine Optimierungen bzgl. Performanz
- Unzuverlässigkeit
 - ▶ Netzwerk
 - ▶ Dienste
- Dynamik
 - ▶ Unabhängige (unkoordinierte) Weiterentwicklung der einzelnen Partner
 - Existenz mit eingeschlossen!
- keine absolute Wahrheit

Integrationsebenen (1)

Präsentationsebene

Präsentationsfragmente
Portlets, „Screen scraping“

Prozeßebene

Anwendungslogikebene

Dienste
Dienstschnittstelle, -semantik
Dienstfindung, -orchestrierung

Informationsebene

Informationsquellen
Datenmodell, Schema,
semantische Heterogenität

Technische Ebene

Netzwerkprotokolle, RPC
Darstellungssyntax

Integrationsebenen (2)

- Wir wollen heute nur eine **Informationsperspektive** einnehmen
 - ▶ Einzubindende Anbieter sind **Informationsquellen**
 - ▶ Interaktion mit den externen Quellen läuft über das Anfrage-Ergebnis-Paradigma
 - ▶ Es existiert eine zentrale Stelle, an der alle Informationen zusammenlaufen
 - ▶ Hauptprobleme
 - Wie überwinde ich die technische Heterogenität?
 - Wie überwinde ich die semantische Heterogenität?

Integrationssebenen (2)

- Andere Möglichkeit: **Dienstintegration**
 - ▶ Einzubindende Anbieter sind **Dienste**
 - ▶ Autonomen Dienste sollen durch eine Infrastruktur die gegenseitige Nutzung ermöglichen
=> Dienstorientierte Architekturen
 - ▶ Hauptprobleme
 - Wie mache ich Dienste interoperabel?
 - Wie finde ich benötigte Dienste?
 - Wie beschreibe ich Dienste?
 - ▶ Wird in der Vorlesung über **Dienstorientierte Integration mit Web Services** behandelt
- Perspektiven schließen sich nicht gegenseitig aus, sondern ergänzen sich!

Szenario – Beispiel-Probleme (1)

- Die einzubindenden Systeme sind sehr heterogen:
 - ▶ **Anbieter 1** besitzt eine Oracle-Datenbank, auf die man über JDBC zugreifen könnte.
 - ▶ **Anbieter 2** bietet eine CORBA-Schnittstelle an, über die wir auf seinen Informationsbestand zugreifen könnten.
 - ▶ **Anbieter 3** setzt ein XML-Datenbanksystem ein. Wir könnten mittels XML-Standards (XPath, XQuery) darauf zugreifen.
 - ▶ **Anbieter 4** hat ein Angebot auf der Basis von statischen HTML-Seiten, auf die nur mittels HTTP zugegriffen werden kann.
- Alle verwenden unterschiedliche Schemata...

Szenario – Probleme (2)

■ Heterogenität hinsichtlich

- ▶ Zugriffsprotokoll
 - HTTP
 - JDBC
 - CORBA/IIOP
- ▶ Informationsmodell/Schema
 - unterschiedliche Modellierung der Daten
- ▶ Ergebnissyntax und Wertformat
 - Datumsrepräsentation, Einheiten/Währungen, ...
- ▶ Anfragemöglichkeiten
 - volle Mächtigkeit von SQL
 - navigierende Anfragen (HTML-Seiten)
 - unterschiedliche Informationsportionierung

Anbindung: Naiver Ansatz

- Die Portalanwendung greift direkt auf die einzubindenden Datenquellen zu
 - ▶ Anpassung von Protokoll, Format, Schema und Anfragesprache in der Portalanwendung selbst
- Nachteil
 - ▶ Jede neue Quelle und jede Änderung an bestehenden Quellen zieht eine Änderung an der Portalanwendung nach sich
 - ▶ Kaum wartbare und beherrschbare Lösung
 - keine Skalierbarkeit
- Deshalb: Entkopplung durch eine Zwischenschicht, die eine integrierte Sicht zur Verfügung stellt

Anbindung: Virtuell vs. materialisiert

- Aufbau einer zentralen Datenbasis im Portal, in die die Inhalte der angeschlossenen Anbieter importiert werden. Diese Datenbasis stellt die integrierte Sicht dar
 - ⇒ **materialisierter Ansatz**
a priori Integration
- Nutzung der Systeme der Fremdanbieter für die Anfrageauswertung (Durchreichen von Anfragen). Die integrierte Sicht ist physisch nicht vorhanden; sie wird von Mediatoren dynamisch bereitgestellt:
 - ⇒ **virtueller Ansatz**
Integration bei Bedarf

Materialisierter Ansatz



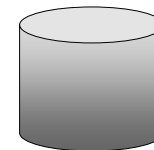
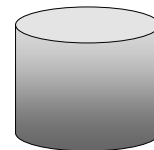
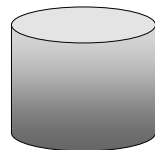
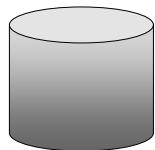
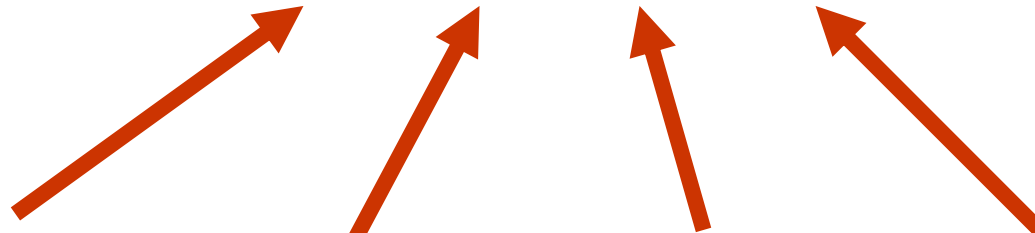
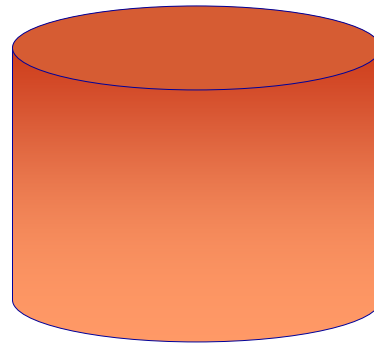
Materialisierter Ansatz

Anwendung

für Anfragen wird nur die zentrale Datenbasis genutzt

Zentrale Datenbasis

(periodischer) Import in die zentrale Datenbasis



Quellen

Informationsintegration und Web-Portale

WS 2003/04

Materialisierter Ansatz – Aufgaben

- Wie kommen die Daten in die Datenbasis?
- **Push:** Die Datenquellen liefern die Daten in einem bestimmten **Austauschformat**
 - ▶ UN/EDIFACT bzw. X.12, XML, ...
 - ▶ Für B2B wichtigste Alternative (vgl. nächste Vorlesung)
 - ▶ Einigung der kooperierenden Partner erforderlich
 - Starker Eingriff in die Autonomie
- **Pull:** Die Daten werden auf den Diensten der Datenquellen gesammelt (Crawling, vgl. Suchmaschinen)
 - ▶ Ähnliche Probleme wie bei virtueller Integration
- Effiziente Importmechanismen für große Datenmengen

Materialisierter Ansatz – Aufgaben



- Wie werden die Daten aktuell gehalten?
 - ▶ Erkennung von Änderungen
 - meist organisatorische und technische Eingriffe in die Systeme erforderlich
 - Oder: Suchmaschinenstrategie
 - ▶ Effiziente Durchführung der Aktualisierung der Datenbasis

Materialisierter Ansatz – Vorteile

- Einfach zu realisieren
 - ▶ Anwendungsentwicklung unterscheidet sich durch zentrale Datenbasis kaum vom Ein-Quellen-Fall
 - ▶ Mehr Informationen über vorhandene Daten
- Performant
 - ▶ Direkte Datenbankzugriffe für Anfrageauswertung
 - ▶ Entkopplung von (evtl. langsamen, nur teilweise verfügbaren) externen Systemen
 - ▶ gezielte Optimierungen möglich
- Nachbearbeitungsoperationen möglich
 - ▶ Von den Fremdanbietern gelieferte Daten können (auch aufwendig) geprüft und bereinigt werden
 - ▶ Aggregation von Daten ebenfalls leicht möglich

Materialisierter Ansatz: Nachteile

- (redundante) Speicherung evtl. großer Datenmengen
 - ▶ leistungsfähige Infrastruktur auf Portal-Seite erforderlich
- Aktualität der Daten ist nicht gewährleistet
 - ▶ klassisches Caching-Problem
- Aktualisierung
 - ▶ auf Initiative der Datenquellen
 - organisatorische Maßnahmen erforderlich
 - Insbesondere: wir brauchen ein Austauschformat!
 - ▶ Aktualisierung auf Initiative des Portals
 - bei großen Datenmengen häufig unpraktikabel
 - keine Information, was geändert wurde
- Keine Kontrolle des Urhebers mehr über die Daten!

Data Warehousing



- Auch unternehmensintern kann eine lose Kopplung ansonsten autonomer Teilsysteme sinnvoll sein.
- Entkopplung der operationalen Systeme von Systemen zur Auswertung und zur Entscheidungsunterstützung
- zentrale Datenbasis heißt dort **Data Warehouse**
- Weitere Aggregationsebenen für spezifische Auswertungen: **Data Marts**

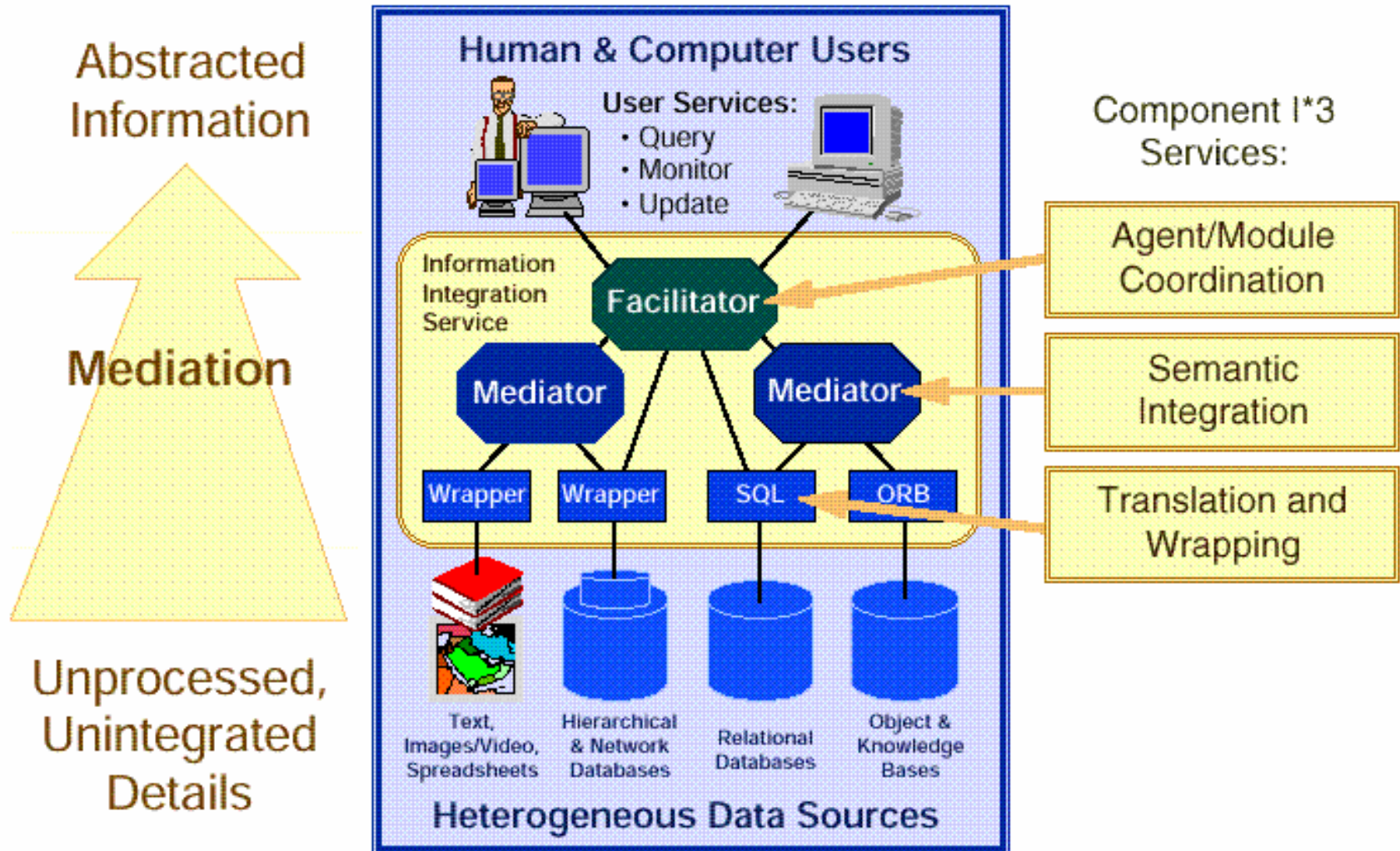
Virtuelle Integration



- Idee: Lasse die Daten in den Quellsystemen und benutze für jede Benutzeranfrage deren Anfragemöglichkeiten
- Entkopplung von Datenquellen und Anwendungen durch eine virtuelle Sicht:

Mediatorarchitektur

I³-Referenzarchitektur



Intelligent Integration of Information

Koordinations- & Managementdienste

- Dienstauswahl und -kombination
- Entdecken von Ressourcen

- Inferenz
- Aktive Mechanismen
- Zustandsverwaltung
- Persistenz

Semantische Integrations- & Transformationsdienste

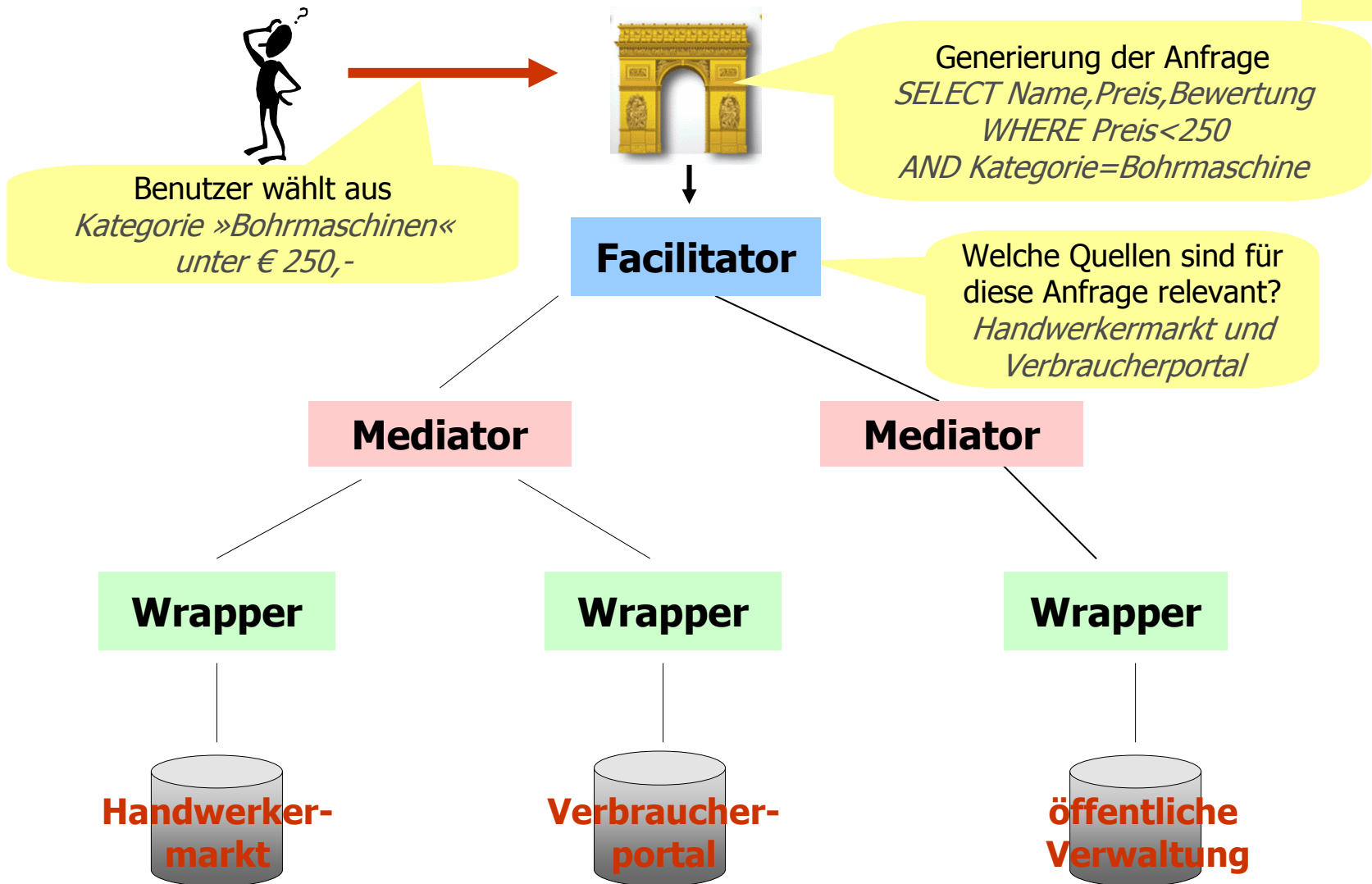
- Schemaintegration
- Datenintegration
- Prozeßintegration

Funktionale Erweiterungen

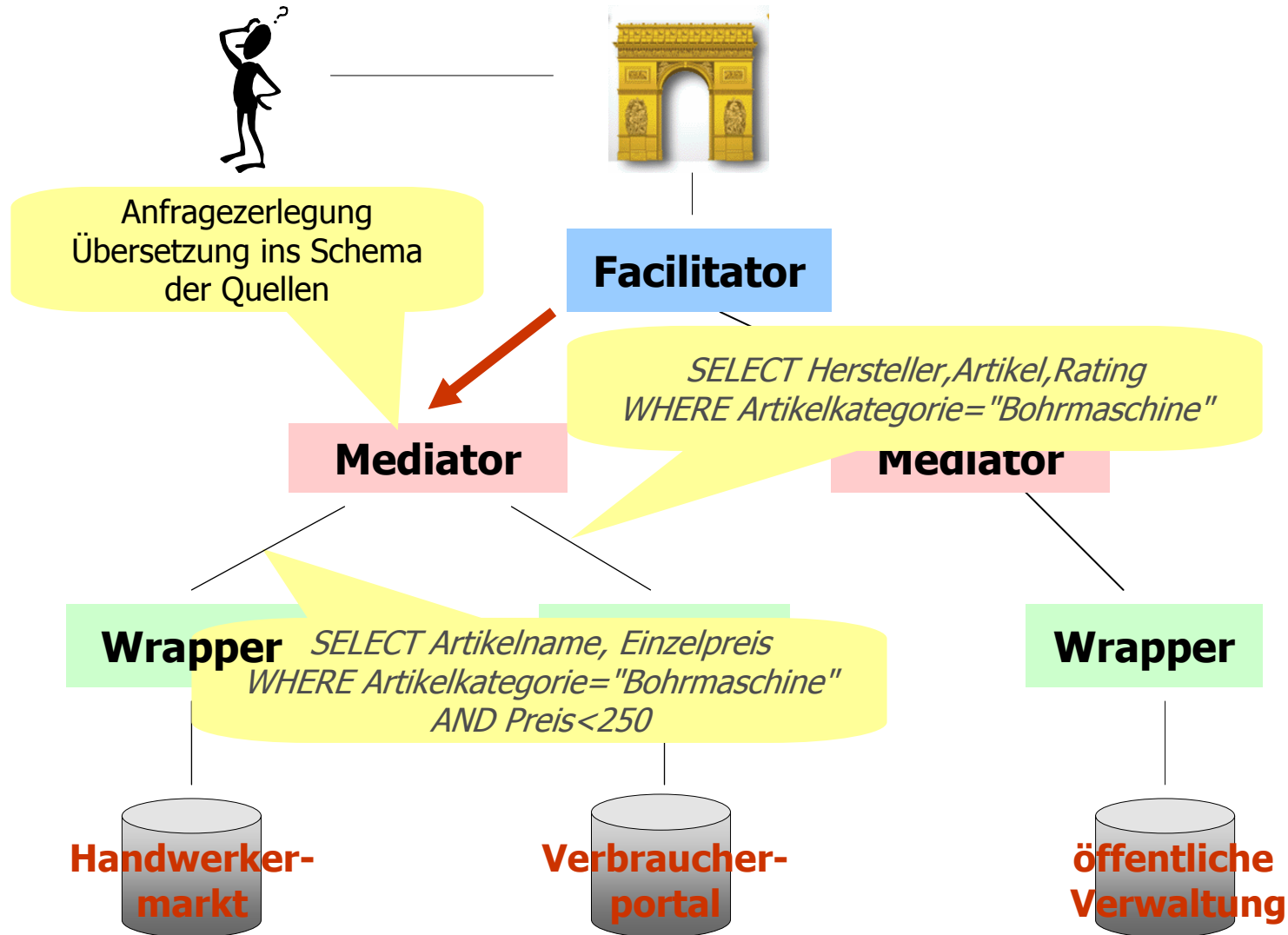
- Kommunikation
- Datenrestrukturierung
- Verhaltensanpassung

Kapselung (*wrapping*)

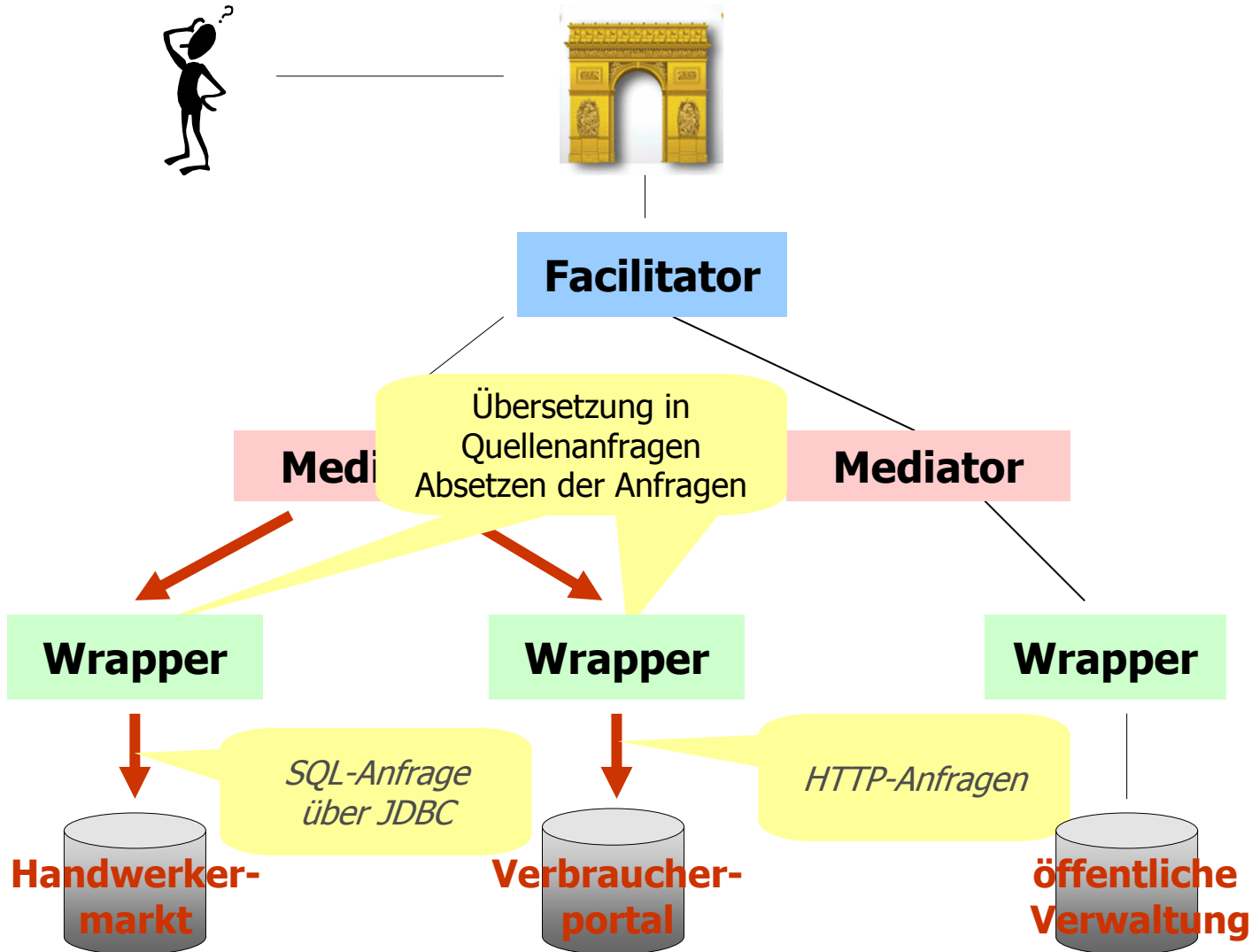
Virtuelle Integration – Beispiel



Virtuelle Integration – Beispiel



Virtuelle Integration – Beispiel



Virtuelle Integration – Beispiel



Facilitator

Zusammenführung der Ergebnisse einer Quelle
Transformation ins gemeinsame Datenmodell
Ausführung von Filteroperationen

Wrapper

Wrapper

Wrapper

JDBC-ResultSet

Quellen liefern
Ergebnis zurück

HTML-Seite

**Handwerker-
markt**

**Verbraucher-
portal**

**öffentliche
Verwaltung**

Virtuelle Integration – Beispiel



Aufbereitung der Ergebnisse für den Benutzer

Übersetzung ins Informationsmodell des Portales
z.B. *Artikelname* -> *Name*
Verschmelzen der Ergebnismengen

Facilitator

Sammeln der Ergebnisse

Mediator

Mediator

Wrapper

Wrapper

Wrapper

**Handwerker-
markt**

**Verbraucher-
portal**

**öffentliche
Verwaltung**

Auf folgende Problembereiche soll im folgenden näher eingegangen werden:

- Facilitator

- ▶ Quellenauswahl

- Mediator

- ▶ einheitliches Informationsmodell
- ▶ Anfragezerlegung, Anfrageübersetzung
- ▶ semantische Integration (s. nächste Vorlesung)
- ▶ Objektverschmelzung

- Wrapper

- ▶ Informationsextraktion

Einheitliches Informationsmodell

Globales Schema

- Den Nutzern bzw. der Portalanwendung soll eine einheitliche Sicht der Informationsobjekte präsentiert werden.
 - ▶ Semantische Integration (s. zweiter Teil)
- Vorgehensweisen:
 1. *Möglichkeit:*
Integration der Schemata vorhandener Quellen in ein **globales Schema**
 - ▶ globales Schema ist als Sicht auf die Quellen definiert (*global as view*)
 - ▶ Ausgehend von Quellenschemata wird globales Schema gebildet
 - ▶ Vgl. Techniken zur Schemaintegration
 - One-shot, iterativ etc.
 - semi-automatische Unterstützung möglich
 - ▶ ändert sich mit der Integration neuer Quellen

Einheitliches Informationsmodell

Domänenmodell

■ 2. Möglichkeit:

Modellierung der Anwendungsdomäne in einem

Domänenmodell

- ▶ Quellen sind als Sicht des Domänenmodells definiert
(*local as view*)
- ▶ Modellierungsaufgabe: zunächst Modellierung des Problembereichs, dann Einbinden der Quellen in dieses Modell
- ▶ (im wesentlichen) quellenunabhängig
- ▶ Dieselbe Aufgabe ist auch beim materialisierten Ansatz zu lösen
 - vergleichbar zu Datenaustauschformat

- Das pauschale Durchreichen aller Anfragen an alle Quellen ist nicht besonders effizient.
 - ▶ Ziel: a priori Abschätzung der Relevanz einer Quelle für eine Anfrage
- Wie beschreibt man die Inhalte der Quelle?
 - ▶ gelieferte Entitäten und Attribute
 - ▶ bei Stichwortsuche: Index der Stichwörter
 - ▶ Beschreibung durch eine Anfragebedingung
 - Auswertung: Kann eine Benutzeranfrage überhaupt von einer Quelle beantwortet werden?
- Wie werden Inhaltscharakterisierungen gewonnen?
 - ▶ ohne weitergehenden Zugriff auf die Quellen schwierig
 - ▶ statistische Methoden

Anfragezerlegung

- Allgemeine Anfragen können aus Verknüpfungen (Joins) mehrerer Quelleninhalte bestehen
- Effiziente Zerlegung erforderlich
- Spezielle Klasse von Anfragen: Verschmelzungsanfragen (*fusion queries*)
 - ▶ nur ein Typ von Informationsobjekten, keine Verknüpfungen
 - ▶ Ziel: Zusammenführung der Informationen über ein Informationsobjekt
 - ▶ dann geht die identische Anfrage an alle (relevanten) Quellen

Anfrageübersetzung – Probleme

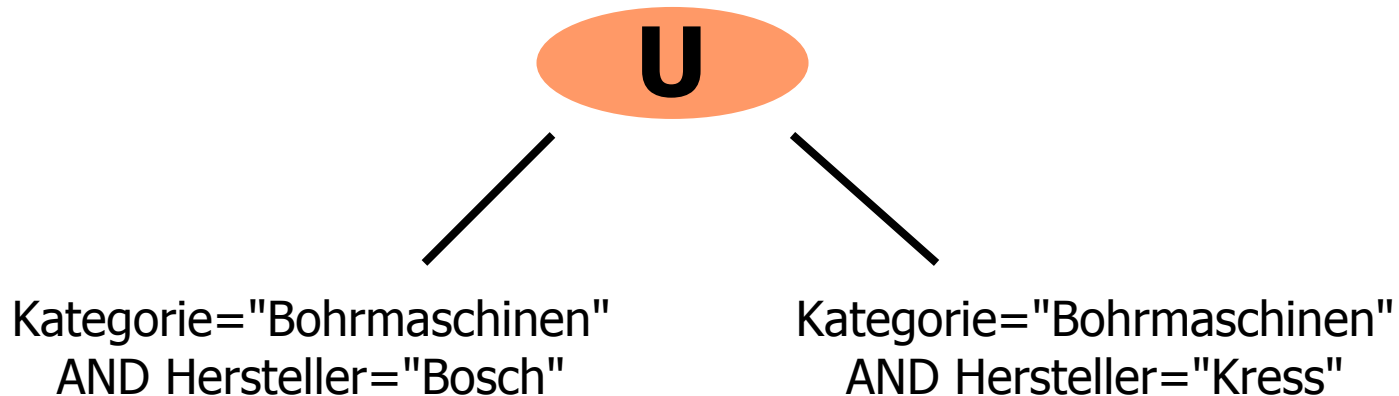
- Ziel: möglichst viel an die Quelle zur Ausführung übergeben (*query shipping*)
- Problem: Wie bildet man fehlende Mächtigkeit der Anfragesprache der Quellen nach?
(**Anfragefähigkeitsanpassung**)
 - ▶ boolesche Operatoren (AND, OR, NOT)
 - ▶ Beschränkungen der Anfragestruktur
 - ▶ Beschränkungen der anfragbaren Attribute
 - ▶ Fehlende Anfrageprädikate (Phrasen, ...)

Anfrageübersetzung – Subsumierende Anfragen und Filter

- Lösung: Konzept der »subsumierenden Anfragen«
 - ▶ Finde eine Anfrage, die von der Quelle unterstützt wird und die eine minimale Obermenge zum gewünschten Ergebnis liefert
 - ▶ Anschließend werden Filteroperationen angewendet.
 - ▶ Nicht immer möglich

Anfrageübersetzung – Bsp.: Fehlender boolescher Operator

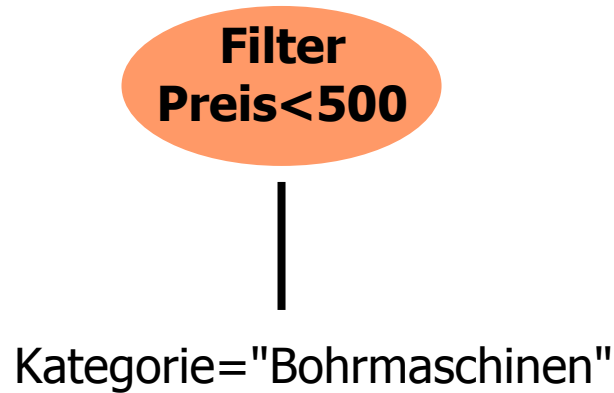
- *Kategorie="Bohrmaschinen" AND
(Hersteller="Bosch" **OR** Hersteller="Kress")*



- ersetze durch äquivalenten Mengenoperator
- vorher evtl. Transformation in DNF/KNF o.ä. erforderlich

Anfrageübersetzung – Beispiel: Nicht suchbares Attribut

- *Kategorie="Bohrmaschinen" AND Preis<500*



- konjunktiv verknüpfte Anfrageprädikate können als Filter nachgeschaltet werden.

Anfrageübersetzung – Unvollständige Information

- Was ist, wenn eine Quelle ein Attribut in der Anfragebedingung nicht besitzt?
- Zwei Lösungsstrategien
 - **Vollständigkeit:** ignoriere die entsprechenden Anfrageteile
 - **Qualität:** werte die entsprechenden Anfrageteile zu falsch aus
- Beispiel: Gesucht sind Bohrmaschinen mit einer Garantiezeit von mindestens 3 Jahren
 - Vollständigkeit: Wenn Garantiezeit nicht verfügbar, dann trotzdem zurückgeben.
 - Qualität: Auf jeden Fall keine Bohrmaschine mit einer Garantiezeit unter 3 Jahren zurückliefern.

Anfrageübersetzung – Lokalisierung

- Wo findet die Anfrageübersetzung statt?
 - ▶ hängt von der Schnittstellendefinition des Wrappers ab
- in den Wrappern
 - ▶ alle Wrapper haben eine einheitliche Schnittstelle im Hinblick auf die Anfragesprache
 - ▶ quellspezifische Anfrageübersetzungstechniken leicht möglich
- in den Mediatoren
 - ▶ Wrapper beschreiben ihre Anfragefähigkeiten
 - ▶ Generische Anfragebearbeitungsfunktionalität
 - ▶ Wrapperübergreifende Optimierung (Alternativquellen o.ä.)
 - ▶ reduziert die Wrapper-Komplexität
- Fazit: meistens Aufteilung der Aufgabe zwischen Wrappern und Mediatoren

Informationsextraktion

■ Einfacher Fall:

- ▶ Quelle liefert Information bereits in sehr strukturierter Form (z.B. relationale DBMS)
- ▶ nur Transformation ins gemeinsame Datenmodell erforderlich

■ Schwieriger:

- ▶ Informationselemente müssen aus unstrukturierter Information extrahiert werden (z.B. HTML-Seiten) und sind über unterschiedliche Ergebnisseiten hinweg fragmentiert
- ▶ z.B. Ausnutzen der HTML-Struktur
 - `html.body.table[2].td`
- ▶ reguläre Ausdrücke
 - Preis: `(\d)+ €`
- ▶ Werkzeuge zur Generierung: z.B. <http://www.equero.de>

■ Transformation ins Domänenmodell

- ▶ Strukturtransformationen
- ▶ Werttransformationen
- ▶ berechnete Attribute
- ▶ Aggregation

**Integration auf
Schemaebene**

■ Objektverschmelzung

- ▶ über semantische Schlüssel
 - evtl. Normalisierung erforderlich
- ▶ paarweiser Ähnlichkeitsvergleich
- ▶ Noch viele offene Probleme (insbesondere Objektidentität)

**Integration auf
Instanzebene**

Virtuelle Integration: Nachbemerkung

- Die hier vorgestellte Aufteilung der Funktionalität auf unterschiedliche Schichten/Komponenten ist idealisiert.
- In realen System ist die Aufgabenteilung zwischen Wrappern und Mediatoren uneinheitlich
 - ▶ **Fette Wrapper.** Hier werden z.B. große Teile der Anfrageübersetzung vom Wrapper übernommen.
 - ▶ **Dünne Wrapper.** Hier ist der Wrapper nur für die Informationsextraktion und die Transformation der Daten in das gemeinsame Datenmodell zuständig.
- Auch die Aufgaben der Facilitator-Schicht werden oft entweder in die Mediatoren oder die Anwendung direkt verlagert.

Virtuelle Integration – Vorteile

■ Aktualität

- ▶ Es wird immer auf den aktuellen Datenbestand zugegriffen.
- ▶ Eine Aktualisierung ist nicht erforderlich.

■ Ressourcenverbrauch

- ▶ Durch die Nutzung der Fremdsysteme ist der Ressourcenverbrauch auf Portalseite niedriger.

■ Autonomie

- ▶ Nutzt man ausschließlich vorhandene Zugänge (z.B. WWW), so ist keinerlei Eingriff in die Datenquellen erforderlich.

Virtuelle Integration – Nachteile

- Performanz
 - ▶ abhängig von den Fremdsystemen und der Netzwerkverbindung
 - ▶ keine gezielten Optimierungen möglich
- Mangelnde Kenntnisse über den Datenbestand
 - ▶ Komfortfunktionen sind erschwert, da keine Analyse des vollständigen Datenbestandes möglich ist
- Nachbereitungsoperationen
 - ▶ alle Transformationen der Daten müssen ausgeführt werden, wenn die Anfrage ausgewertet wird
- Änderungen an den Datenquellen führen zu Problemen
 - ▶ da keine wohldefinierte Schnittstelle

Virtuell vs. materialisiert – Fazit

- Virtueller Ansatz ist zu bevorzugen bei Datenquellen
 - ▶ mit hoher Änderungsrate
 - ▶ mit großem Datenvolumen
 - ▶ mit geringer Anfragehäufigkeit/geringem Anfragevolumen
- Um den Implementierungsaufwand für ein virtuelles Integrationssystem in Grenzen zu halten:
 - ▶ Wie heterogen sind die anzubindenden Systeme?
 - ▶ Welche Anfragefunktionalität wird benötigt?
(Joins?, boolesche Verknüpfungen, ...)
- Oft bietet sich auch ein hybrider Ansatz an
 - ▶ kleinere, weniger mächtige Quellen werden repliziert
 - ▶ die restlichen werden virtuell angebunden

Semantische Integration



Schemaintegration: Motivation

- Neben der Anfrageübersetzung ist die Herstellung eines einheitlichen Informationsmodells/Schemas das wichtigste Problem
- Zwei Schritte
 - ▶ Definition des integrierten Schemas
 - ▶ Definition der Abbildungsregeln (»mappings«)
 - von den Quellen in das integrierte Schema
 - im virtuellen Fall bzw. bei bidirektionaler Kommunikation: vom integrierten Schema in die Quellschemata
- in dieser Vorlesung nur kurzer Überblick
 - ▶ Vertiefung in „Datenbankeinsatz“

Arten von Integrationskonflikten

Datenmodell

Unterschiedliche Ausdrucksmächtigkeit von Datenmodellen

Relational, Objektorientiert, XML, Semistrukturiert

Logische Ebene: Schema

Freiheitsgrade in der Modellierung der realen Welt
klassische »Schemaintegration«

Daten Instanzen

Zusammenführen von Instanzen
(Objektidentität/Duplikaterkennung)

Widersprüchlichkeit, Inkonsistenzen, Subjektivität

Ebenen der semantischen Integration

■ **Schemaintegration**

(bzw. Integration auf Schemaebene)

- ▶ Zusammenführen und semantische Vereinheitlichung allein auf der Basis von Schemainformationen
 - Typen, Klassen, Attribute, Vererbungsrelation etc.
 - Abbildung wird auf alle Instanzen angewendet

■ **Instanzintegration** (bzw. Integration auf Instanzebene)

- ▶ Zusammenführen von Informationen über einzelne Instanzen (oder Objekte)
 - auf der Basis von Werten
 - typisch: Ähnlichkeitsmaße, paarweise Vergleich, Objektidentität, wertabhängige Abbildungsregeln
 - Schemainformation reicht als Information nicht aus!

Datenmodellkonflikte (1)

- Datenmodellkonflikte kommen durch unterschiedliche Ausdrucksmächtigkeit von Datenmodellen zustande
 - ▶ unterschiedliche Strukturbeschreibungen
 - ▶ unterschiedliche Integritätsbedingungen (Kardinalitäten, referentielle Integrität etc.)

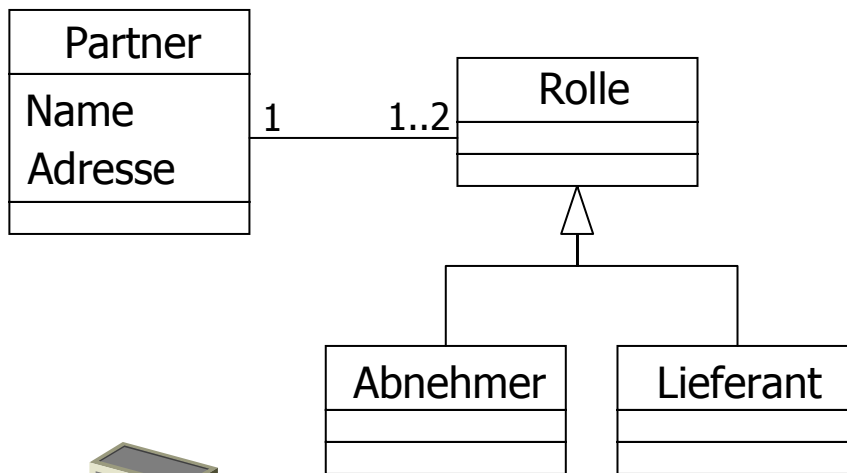
- In der Praxis wichtige Datenmodelle sind
 - ▶ relationales Modell
 - ▶ objektorientiertes Modell
 - ▶ objektrelationales Modell
 - ▶ XML-Datenmodell(e)

Datenmodellkonflikte (2)

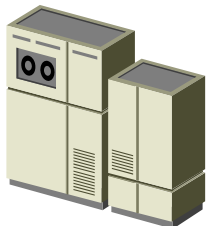
```

<!ELEMENT hersteller (name, adresse, ware*) >
<!ELEMENT name (#PCDATA) >
<!ELEMENT adresse (strasse, stadt, plz) >
<!ELEMENT ware (nummer, preis) >
    
```

XML



Objektorientiert

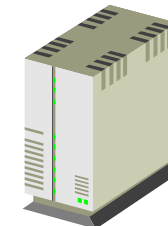


System 1

Hersteller			
Name	Straße	Stadt	PLZ

Waren		
Nummer	Preis	hergestelltVon

Relational



System 2

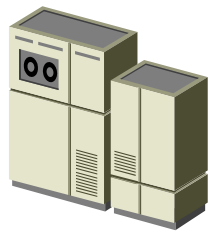
Datenmodellkonflikte (3)

- Datenmodellkonflikte treten sehr häufig auf
 - ▶ Mehrschichtenarchitekturen:
RDBMS ↔ EntityBeans ↔ Präsentation HTML
 - ▶ in virtuellen Integrationsansätzen:
meistens die Aufgabe von Wrappern
- Für »klassische Anwendungen« sehr viele Lösungsansätze
 - ▶ OR-Mapping-Tools
 - JDO (Java Data Objects)
 - ▶ XML-Export aus Datenbanken
 - siehe Vorlesung „XML und Datenbanken“
 - ▶ JAXB (Java XML-Binding)
 - ▶ ...

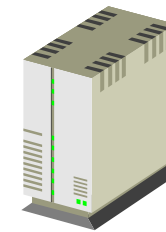
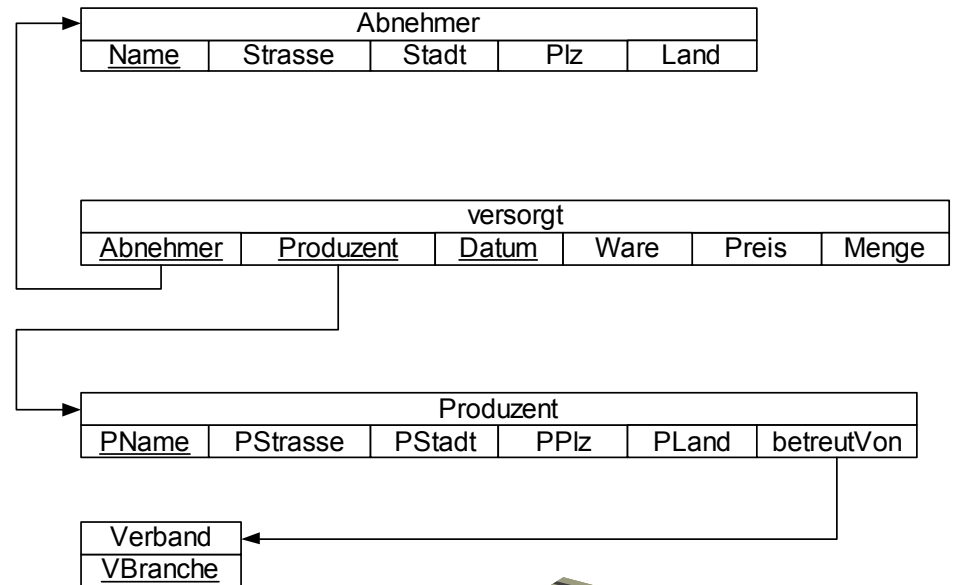
Logische Konflikte (1)

- Logische Konflikte kommen durch Freiheitsgrade in der Modellierung der Realwelt durch Schemakonstrukte zustande
 - ▶ Beschreibung
 - unterschiedliche Benennung
 - unterschiedliche Wertdarstellung
 - ▶ Struktur
 - unterschiedliche Schemakonstrukte
 - ▶ Semantik
 - implizites Wissen
 - Inkonsistenzen
 - ...

Logische Konflikte: Beispielschemata



System 1

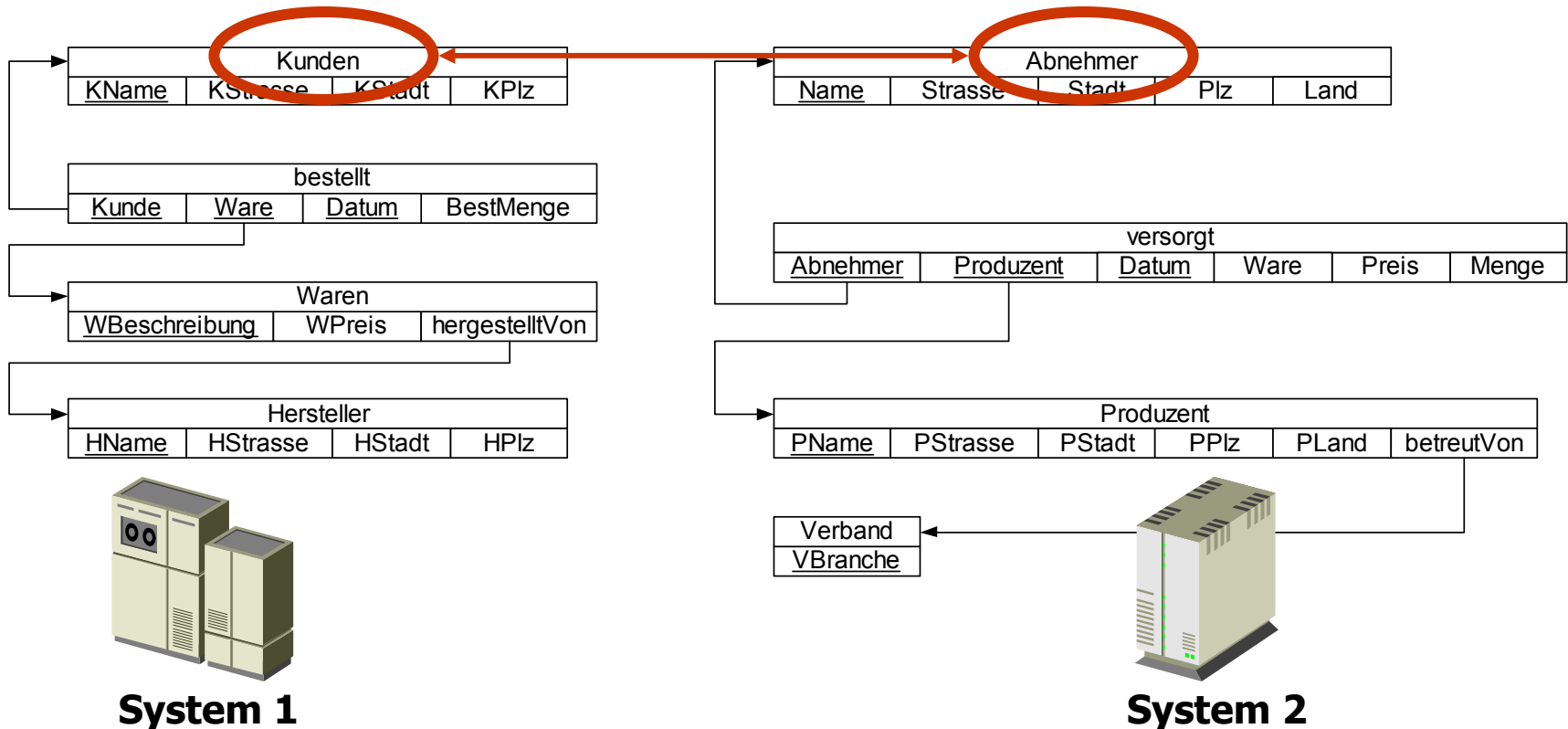


System 2

Logische Konflikte: Beschreibungskonflikte (1)

■ Tabellennamenkonflikte

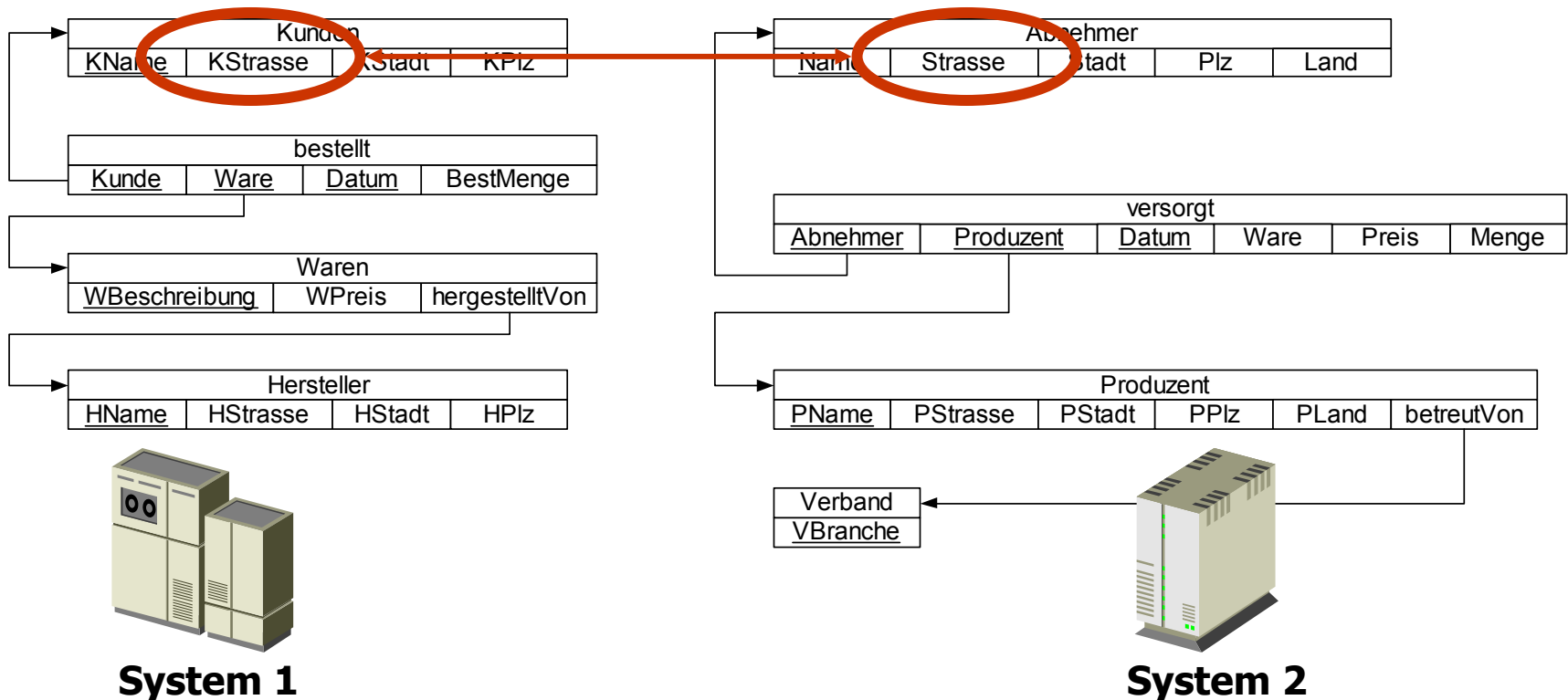
- Verschiedene Namen für gleiche Tabellen
- Gleiche Namen für verschiedene Tabellen



Logische Konflikte: Beschreibungskonflikte (2)

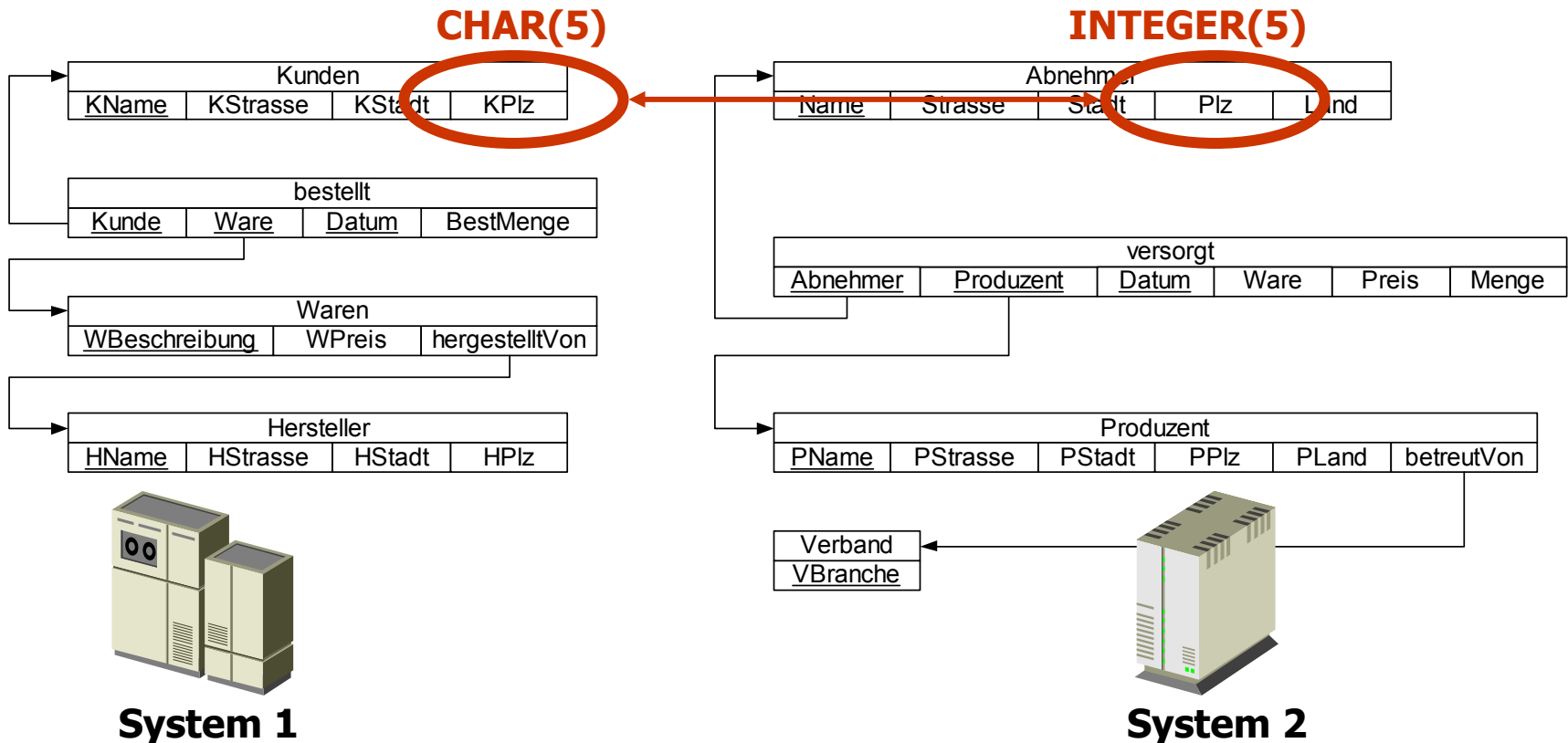
■ Attributnamenskonflikte

- Verschiedene Namen für gleiche Attribute
- Gleiche Namen für verschiedene Attribute



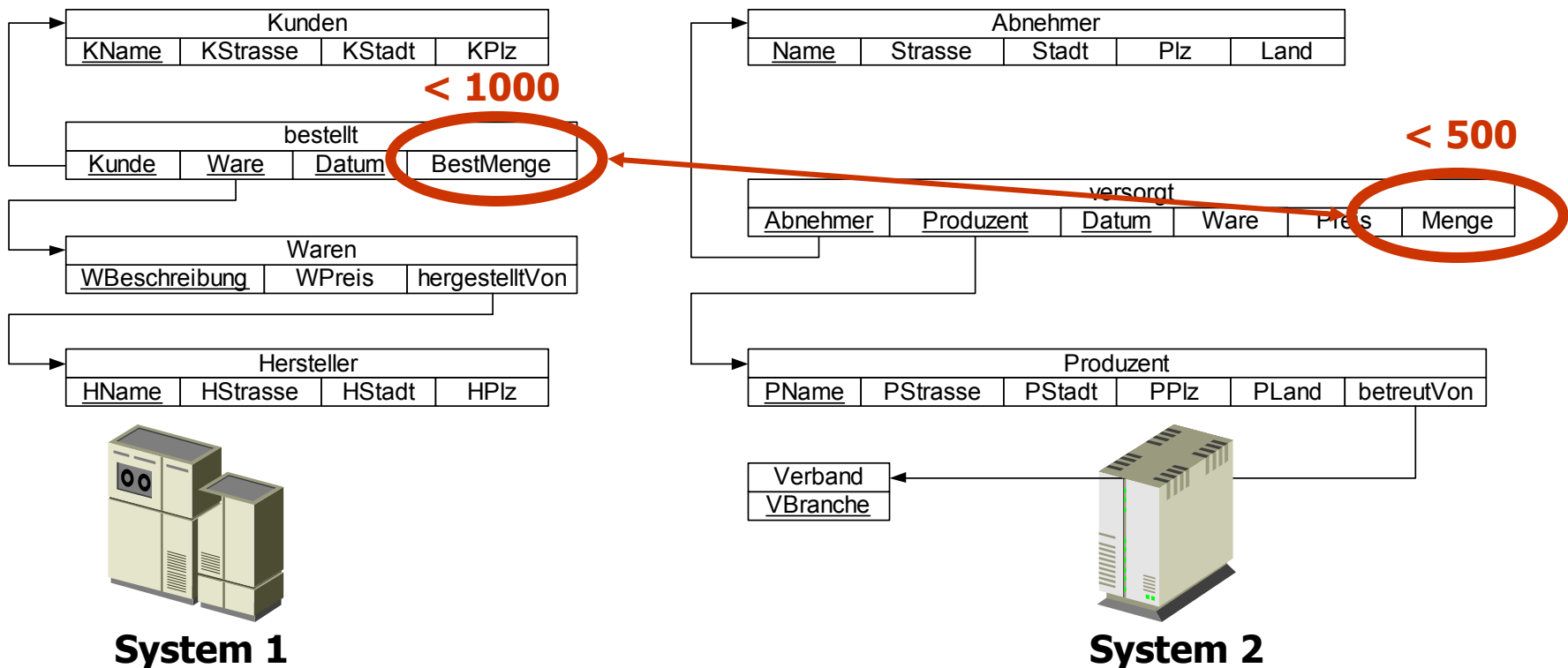
Logische Konflikte: Beschreibungskonflikte (3)

- Integritätsbedingungskonflikte
 - ▶ Datentypkonflikte



Logische Konflikte: Beschreibungskonflikte (4)

- Integritätsbedingungskonflikte
 - ▶ Bedingungskonflikte



Logische Konflikte: Beschreibungskonflikte (5)

- Unterschiedliche Repräsentationen von Werten



- ▶ Verschiedene Ausdrücke für gleichen Wert

Kunden			
KName	KStrasse	KStadt	KPlz
Blau AG	Domstr. 3	Köln	50033
...

Abnehmer				
Name	Strasse	Stadt	Plz	Land
BLAU	Domstrasse 3			
...

bestellt			
Kunde	Ware	Datum	BestMenge
Blau AG	Zement	20.05.01	100
...

versorgt					
Abnehmer	Produzent	Datum	Ware	Preis	Menge

Waren		
WBeschreibung	WPreis	hergestelltVon

Produzent					
PName	PStrasse	PStadt	PPlz	PLand	betreutVon

Hersteller			
HNar	HStrasse	HStadt	HPlz
oo			

Verband
VBranch

System 1

Informationsintegration und Web-Portale

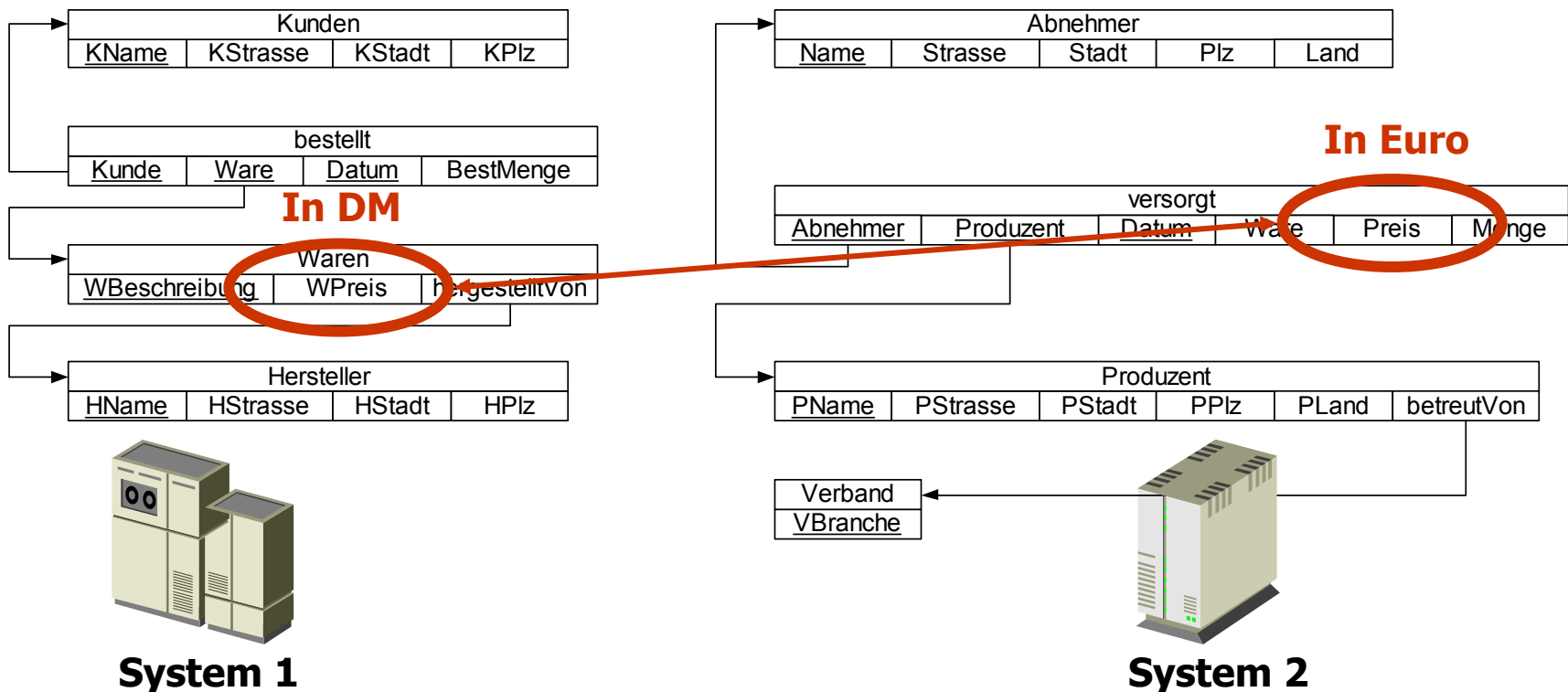
System 2

WS 2003/04

60

Logische Konflikte: Beschreibungskonflikte (5)

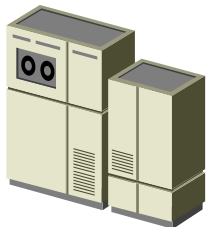
- Unterschiedliche Repräsentationen von Werten
 - ▶ Verschiedene Einheiten



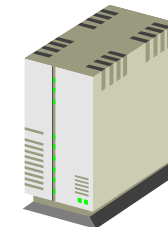
Logische Konflikte: Beschreibungskonflikte (6)



- Unterschiedliche Granularität
 - ▶ Beispiel: „Hersteller.Groesse“:
 - Eins aus { groß, mittel, klein }
 - Jahresumsatz in Mio. €
 - ▶ Beispiel: „Kleidung.Groesse“:
 - { 24, 25, 26, ..., 48, 50, 52, ... }
 - { XS, S, M, L, XL, XXL }



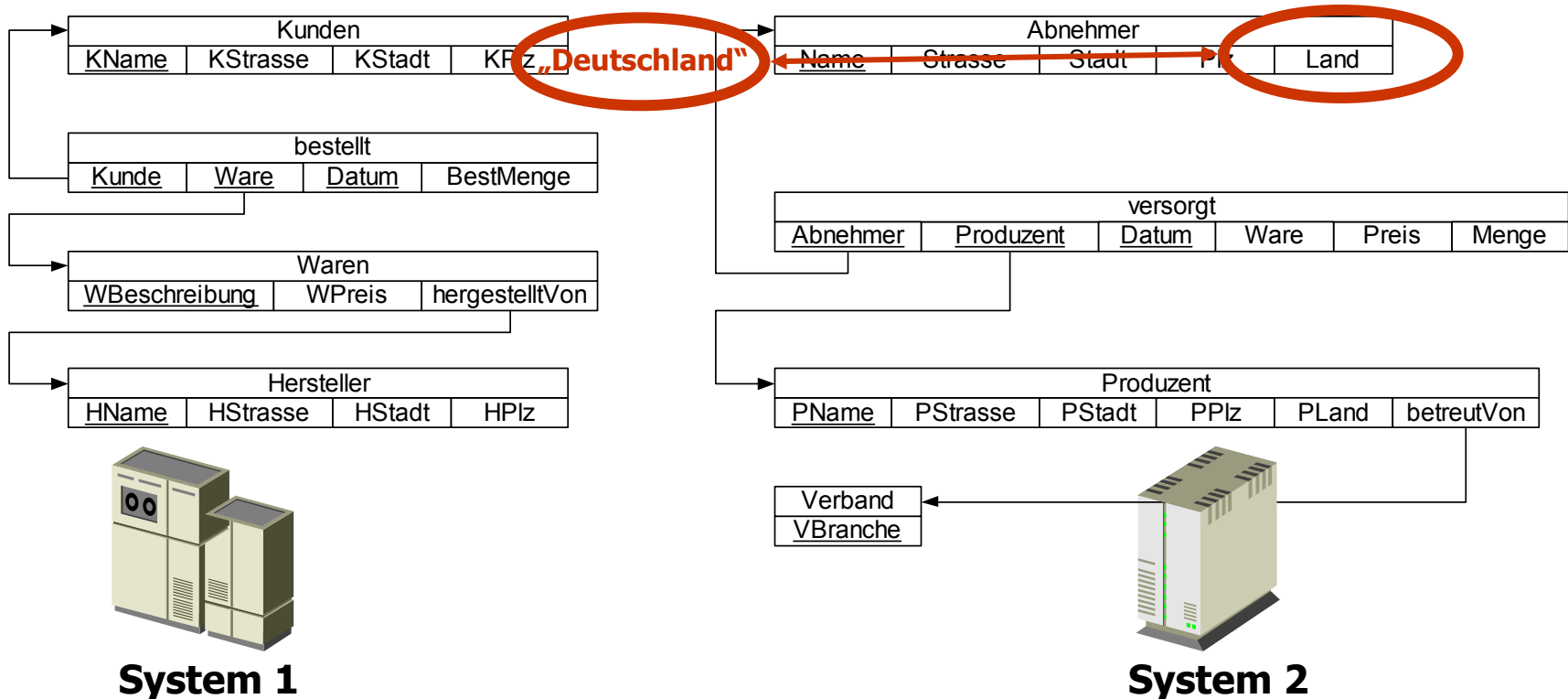
System 1



System 2

Logische Konflikte: Strukturkonflikte (1)

- Tabellenstrukturkonflikte
 - ▶ Fehlende, aber implizite Attribute



Logische Konflikte: Strukturkonflikte (2)

■ Meta-Konflikte



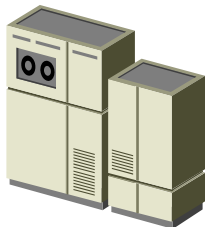
„Status“ eines Kunden ist ein Wert in der Datenbank

Status eines Kunden ist im Schema kodiert

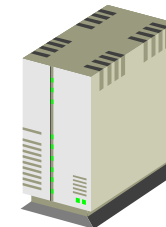
Kunden		
<u>Name</u>	Adresse	Status
Carl	Mannheim	privat
Katja	Karlsruhe	geschaeftlich

Privatkunden	
<u>Name</u>	Adresse
Paul	76131 Karlsruhe, ...

Geschäftskunden	
<u>Name</u>	Adresse
Inge	69115 Heidelberg, ...



System 1



System 2

Logische Konflikte: Semantische Konflikte (1)

■ Semantische Konflikte

- ▶ Ursache: Mehrere Systeme modellieren denselben Weltausschnitt (gleiche Real-Welt-Objekte)
 - Sog. „Semantische Überlappung“
- ▶ Horizontale vs. vertikale Überlappung
 - Horizontal: Weltausschnitte und Daten überschneiden sich
 - Vertikal: Weltausschnitte überschneiden sich, Daten ergänzen sich
- ▶ Problem der Identität von Real-Welt-Objekten



Hersteller			
Name	Straße	Stadt	PLZ

Hersteller			
Name	Straße	Stadt	PLZ

System 1

Informationsintegration und Web-Portale

Horizontal

Hersteller			
Name	Straße	Stadt	PLZ

Vertikal

Hersteller		
Name	PartnerSeit	Branche

System 2

WS 2003/04

65

Logische Konflikte: Semantische Konflikte (2)

■ Feststellung der Objektidentität

- ▶ „Object Fusion“
- ▶ Duplikaterkennung

Kunden			
<u>KName</u>	<u>KStrasse</u>	<u>KStadt</u>	<u>KPlz</u>
Blau AG	D...str. 3	Köln	50033
...

Identisch?

Abnehmer				
<u>Name</u>	<u>Strasse</u>	<u>Stadt</u>	<u>Plz</u>	<u>Land</u>
BLAU				
...

bestellt			
<u>Kunde</u>	<u>Ware</u>	<u>Datum</u>	<u>BestMenge</u>
Blau AG	Zement	20.05.01	100
...

versorgt					
<u>Abnehmer</u>	<u>Produzent</u>	<u>Datum</u>	<u>Ware</u>	<u>Preis</u>	<u>Menge</u>

Waren		
<u>WBeschreibung</u>	<u>WPreis</u>	<u>hergestelltVon</u>

Produzent					
<u>PName</u>	<u>PStrasse</u>	<u>PStadt</u>	<u>PPlz</u>	<u>PLand</u>	<u>betreutVon</u>

Hersteller			
<u>HName</u>	<u>HStrasse</u>	<u>HStadt</u>	<u>HPlz</u>

Verband
<u>VBranche</u>

Logische Konflikte: Semantische Konflikte (3)

■ Falsche Daten

- Nicht korrekte Einträge
- Nicht identisch? Identisch aber eins fehlerhaft?

Kunden			
KName	KStrasse	KStadt	KPlz
Blau AG	Domstr. 3	Köln	50033
...

Abnehmer				
Name	Strasse	Stadt	Plz	Land
BLAU			50044	
...



bestellt			
Kunde	Ware	Datum	BestMenge
Blau AG	Zement	20.05.01	100
...

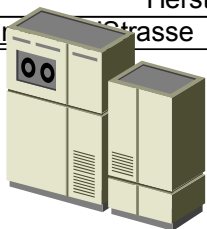
versorgt					
Abnehmer	Produzent	Datum	Ware	Preis	Menge

Waren		
WBeschreibung	WPreis	hergestelltVon

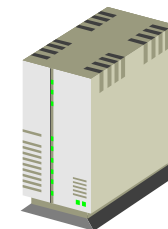
Produzent					
PName	PStrasse	PStadt	PPlz	PLand	betreutVon

Hersteller			
HNar	HStrasse	HStadt	HPlz

Verband
VBranch



System 1



System 2

Logische Konflikte: Semantische Konflikte (4)

- Falsche Daten
 - ▶ Veraltete Daten

Kunden			
KName	KStrasse	KStadt	KPlz
Blau AG	Domstr. 3	Köln	50033
...

Abnehmer				
Name	Strasse	Stadt	Plz	Land
BLAU	Neustr. 8, Heidelberg	69115	Deutschland	
...

bestellt			
Kunde	Ware	Datum	BestMenge
Blau AG	Zement	20.05.01	100
...

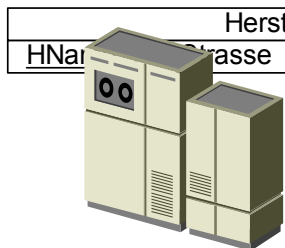
versorgt					
Abnehmer	Produzent	Datum	Ware	Preis	Menge

Waren		
WBeschreibung	WPreis	hergestelltVon

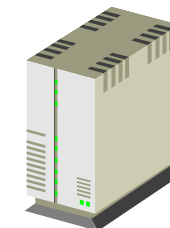
Produzent					
PName	PStrasse	PStadt	PPlz	PLand	betreutVon

Hersteller			
HName	HStrasse	HStadt	HPlz

Verband
VBranche



System 1



System 2

Konflikte – Logik (Semantik)

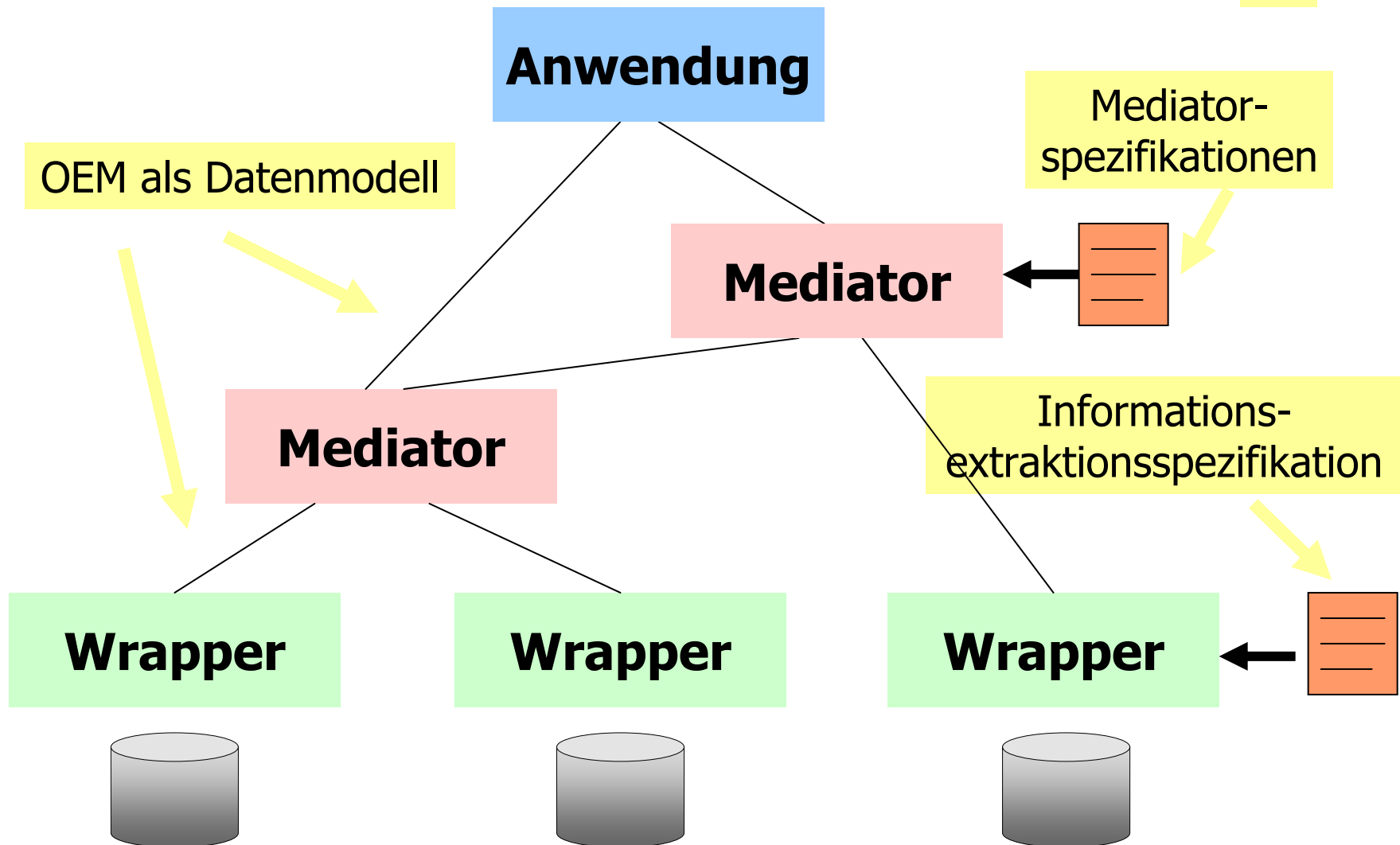
- „Implizites semantisches Wissen“
- Zwei Schemata mit: Hersteller.Sitz
 - ▶ Schema A verwendet „Stadt“
 - ▶ Schema B verwendet „Land“
- Gesucht: „Alle französischen Hersteller“
 - ▶ Man muss wissen, welche Städte in Frankreich liegen !

Fallbeispiel: TSIMMIS



- Forschungsprojekt der Stanford University 1994-1997
- Grundproblem:
 - ▶ Erstellen von Mediatoren und Wrappern ist aufwendig
- Ideen
 - ▶ Automatische Generierung von Mediatoren und Wrappern auf der Basis von deklarativen Spezifikationen
 - ▶ hierzu: Verwendung eines flexiblen Datenmodells

TSIMMIS: Überblick



TSIMMIS: Datenmodell OEM – Grundidee

- Beobachtung (Autonomie der Quellen!):
 - ▶ Wissen über das Schema ist unvollständig
 - ▶ Struktur ist irregulär
 - ▶ Schema ändert sich, ohne daß die Quellen darüber benachrichtigen
- Deshalb
 - ▶ traditionelle Datenmodelle (relational, objektorientiert) sind mit ihren Beschreibungsmitteln zu starr
 - ▶ insbesondere Schemabegriff zu inflexibel

=> **Semistrukturiertes Datenmodell**

Semistrukturiertes Datenmodell

Semistrukturierte Daten haben eine Struktur, aber ...

- Struktur weist Variationen auf
 - ▶ fehlende oder zusätzliche Inhalte
 - ▶ wechselnde Repräsentationen
- Struktur ist implizit und in die Daten eingebettet
- Strukturbeschreibungen sind nur indikativ, nicht einschränkend
 - ▶ a posteriori Schema, gewonnen durch Analyse
- Struktur ist partiell

TSIMMIS: Datenmodell OEM

<oid, label, type, value>

Objektidentifikator
(optional)

beschreibender
Objektbezeichner

Typ:
*integer, real,
string, set*

Wert

```
<produkte, set,  
  { <produkt, set, {  
    <name, string, "Bosch PSB 500 RE">,  
    <preis, real, 51.50>,  
    <währung, string, "€"> }  
  }  
>
```

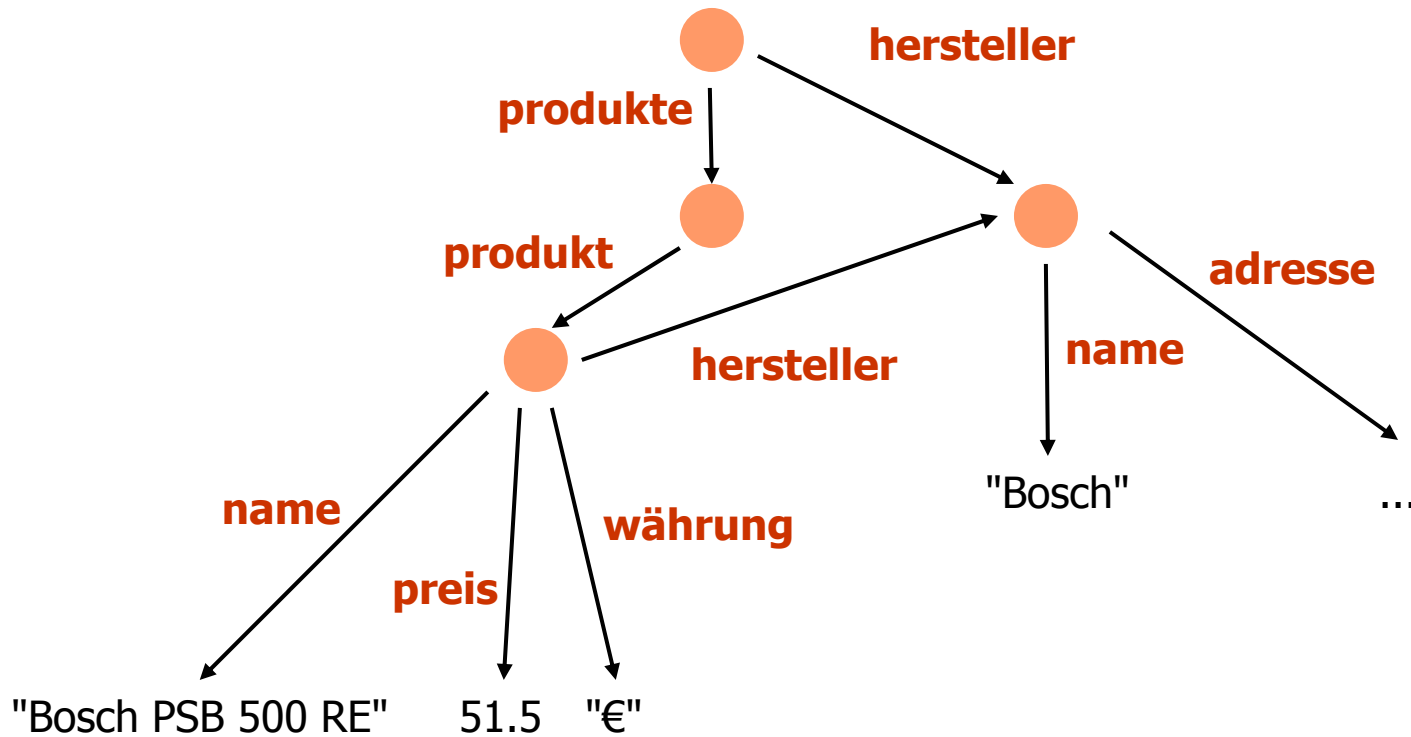
Erinnert das
nicht an XML?

TSIMMIS: OEM – Objektidentität

- OEM bietet auch eine Objektidentität
- Durch die Verwendung von Objektreferenzen *&oid* lassen sich auch beliebige Graphen aufbauen
 - ▶ insbesondere auch zyklische Strukturen sind möglich

```
<&bosch, hersteller, set, {  
    <name, string, "Bosch">,  
    ... }  
>  
<produkt, set, {  
    <name, string, "PSB 500 RE">,  
    &bosch  
}>
```

TSIMMIS: OEM – Graphendarstellung



TSIMMIS: Mediatorspezifikationen

- Aufgabe des Mediators
 - ▶ generiert einen logischen Ausführungsplan, der Wrapperanfragen enthält
 - ▶ integriert die unterschiedlichen Ergebnisse
 - ▶ berücksichtigt die unterschiedlichen Anfragefähigkeiten
- Herausforderungen
 - ▶ Anfragezerlegung und -übersetzung
 - ▶ Modellierung von Anfragefähigkeiten
 - ▶ Integrationsregeln

TSIMMIS: Mediatorspezifikation – MSL

- Mediatorsicht wird definiert durch Regeln, die die Quelleninformationen transformieren
- MSL – Mediator Specification Language
 - ▶ es werden grundsätzlich nur konjunktive (SPJ)-Anfragen betrachtet

- Regeln Kopf : – Rumpf
 - ▶ Kopf Konstruktionsteil, beschreibt die vom Mediator bereitgestellte Sicht
 - ▶ Rumpf Objektmuster für die zugrundeliegenden Quellen
 - Matching der Objektmuster bewirkt eine Variablenbindung
 - Auf der Basis dieser Variablenbindungen können auch Bedingungen formuliert werden
 - externe Prädikate möglich

TSIMMIS: MSL-Beispiele – Joins (1)

<produkt, { <name N>, <preis P>, <bewertung B> }>:-

AND <produkt { <bezeichnung N>, <preis P>}>@baumarkt
 <produkt { <name N>, <bewertung B> }>@verbraucherportal

■ Anfragezerlegung

▶ *Anfrage:*

<produkt { <name "Bosch PSB 500 RE">, <preis P>,
 <bewertung B> }>

▶ *Quellenanfragen:*

<produkt { <bezeichnung "Bosch PSB 500 RE">, <preis P>>
<produkt { <name "Bosch PSB 500 RE">, <bewertung B>>

TSIMMIS: MSL-Beispiel – Joins (2)

<produkt, { <name N>, <preis P>, <bewertung B> }>:-

AND <produkt { <bezeichnung N>, <preis P> }>@baumarkt
 <produkt { <name N>, <bewertung B> }>@verbraucherportal

■ Integration

▶ Quellen:

<produkt { <bezeichnung "Bosch PSB 500 RE">, <preis "120"> }>

<produkt { <name "Bosch PSB 500 RE">, <bewertung "4"> }>

▶ Ergebnis:

<produkt { <name "Bosch PSB 500 RE">, <preis "120">, <bewertung "4"> }>

TSIMMIS: MSL-Beispiel – Platzhalter

- MSL soll eine gewisse Robustheit gegenüber Änderungen an der Struktur der Datenquellen bringen

- ▶ Platzhalterzeichen

```
<produkt, { <name N>, <herstellergarantiezeit H> }>:-  
  <produkt { <bezeichnung N>, <*garantiezeit H >}>
```

- ▶ integriert auch korrekt:

```
<produkt { <bezeichnung "Bosch PSB 500 RE">,  
  <hersteller {  
    <name "Bosch">, <garantiezeit "3 Jahre">} >  
}>
```

MSL-Beispiel – Objektverschmelzung

■ Verwendung von Skolemfunktionen zur Integration

```
<&b(I), buch, {<bewertung B> }>:-
  <produkt { <kategorie "buch">, <isbn I>,
    <bewertung B> }>@verbraucherportal
```

```
<&b(I), buch, {<titel T>, <preis P> }>:-
  <produkt { <bezeichnung T>, <kategorie "buch">, <isbn I>}
  @baumarkt
```

- ▶ Zusammenzuführenden Objekten aus unterschiedlichen Quellen wird der gleiche Objektidentifikator zugewiesen

TSIMMIS: Wrapperspezifikationen

- Im Wrapper müssen die Anfragen in die Quellsyntax übersetzt werden und die Ergebnisse in das Datenmodell OEM übersetzt werden.
- Übersetzung in native Quellenanfragen
Beschreibung der möglichen Anfragen über MSL und assoziierte Aktionen, die den nativen Quellenanfragen entsprechen
 - ▶ produkt { <name X > } :-
// \$\$ = „select * from produkte where name=\"\$X\" //

TSIMMIS: Informationsextraktion

- Für HTML-Quellen
- Regeln der Form **[variables, source, pattern]**
 - ▶ Semantik: {variables} = match(source,pattern)
 - ▶ schrittweise, hierarchische Zerlegung in einzelne Teile

```
[ "root", "get('http://www.verbraucherportal.de/bohrmaschinen.html')",  
  "<table>#</table>" ]
```

```
[ "produkte", "split(root,'<tr>')", "#"]
```

```
[ "produktname,bewertung", "produkte", "<td>#</td><td>#</td>" ]
```

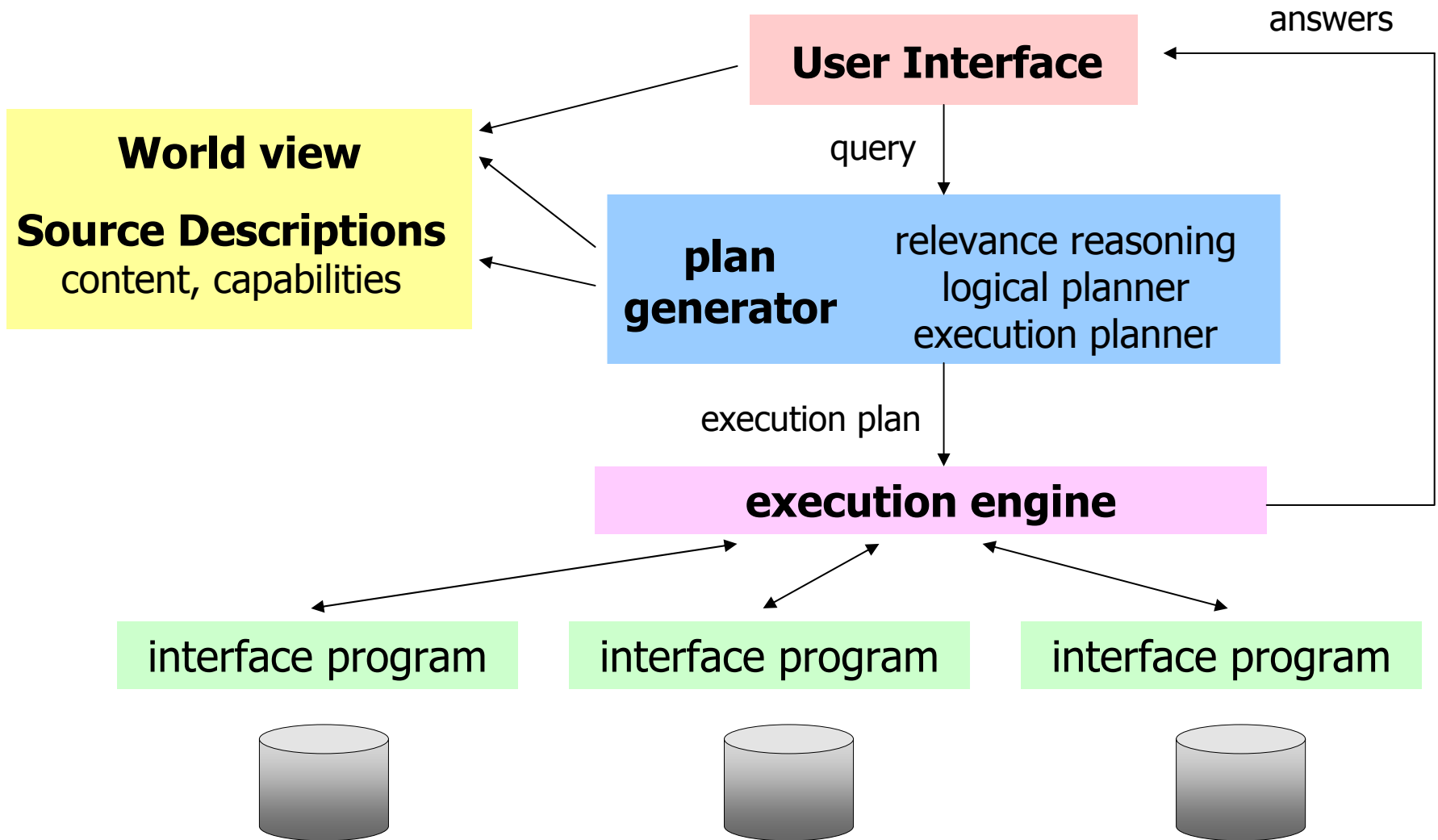
Fallbeispiel: Information Manifold



Information Manifold: Allgemein

- Forschungsprojekt von AT&T 1995/96
- Besonderer Fokus war die Skalierbarkeit des Integrationsansatzes
- Domänenmodellansatz (*world view*)
 - ▶ Modellierung der Domänen unabhängig von den Quellen
 - ▶ dadurch müssen bei der Spezifikation auch weniger Kombinationsmöglichkeiten zwischen den Quellen betrachtet werden

Information Manifold: Architektur



Information Manifold: Datenmodell

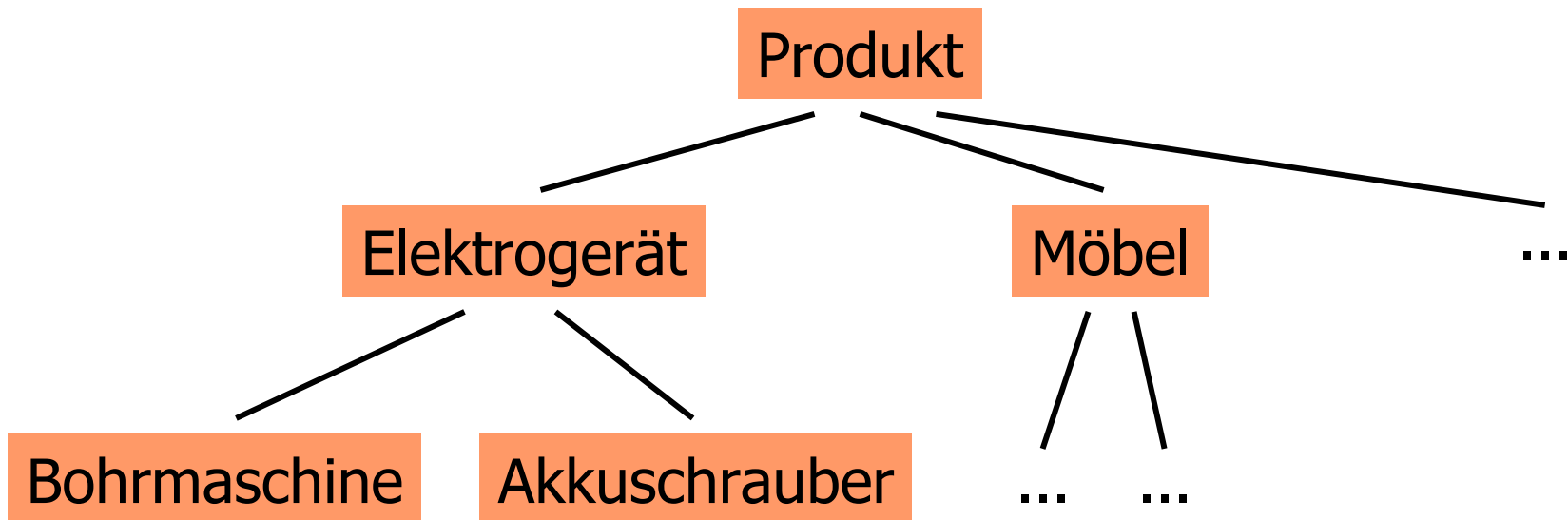
■ Datenmodell

- ▶ objektorientierte Modellierung der Entitäten
- ▶ virtuelle Relationen auf diesen Klassen
- ▶ Description Logic »CLASSIC« für die Definition der Quellsichten

■ Anfrageplanung

- ▶ Grundproblem: Beantwortung von Anfragen mittels Sichten
- ▶ logikbasiertes Suchverfahren

Information Manifold: Domänenmodell-Beispiel



Angebot(herstellername, produkt, preis)
Bewertung(produkt, bewertung)
Hersteller(herstellername, anschrift)

Information Manifold: Quellenbeschreibungen

■ Inhalte (*content*)

- ▶ als Sichten der »world view« mittels konjunktiver Anfragen
- ▶ Quellen werden als unvollständige Sichten modelliert

■ Fähigkeiten (*capabilities*)

(Input, Output, Selectors, min, max)

- ▶ **Input:** Menge der Attribute, für die Bindungen gegeben werden; daraus müssen zwischen [**min,max**] Bindungen gegeben sein
- ▶ **Output:** Menge der Ausgabeattribute
- ▶ **Selectors:** Attribute, auf denen Ungleichheitsprädikate unterstützt werden

Information Manifold: Beispiel-Quellenbeschreibung

■ Baumarkt

- ▶ $v_1(\text{hersteller, produkt, preis})$:
 $\text{Angebot}(\text{hersteller, produkt, preis}) \wedge \text{Elektrogeräte}(\text{produkt})$
- ▶ $[\{\text{hersteller, produkt, preis}\}, \{\text{hersteller, produkt, preis}\}, \{\text{preis}\}, 1, 3]$

■ Verbraucherportal

- ▶ $v_2(\text{produkt, bewertung})$:
 $\text{Bewertung}(\text{produkt, bewertung}) \wedge \text{Produkt}(\text{produkt})$
- ▶ $[\{\text{produkt}\}, \{\text{bewertung}\}, \{\}, 1, 1]$

■ Black&Decker-Händler:

- ▶ $v_3(\text{hersteller, produkt, preis})$:
 $\text{Angebot}(\text{hersteller, produkt, preis}) \wedge \text{Elektrogeräte}(\text{produkt})$
 $\wedge \text{Hersteller} = \text{"Black\&Decker"}$

Information Manifold: Beispiel-Anfrage

■ Anfrage:

$q(H,N,P,B)$: $\text{Angebot}(H,N,P) \wedge \text{Bewertung}(P,B) \wedge$
 $\text{Bohrmaschine}(P) \wedge H = \text{"Bosch"}$

■ Zerlegung:

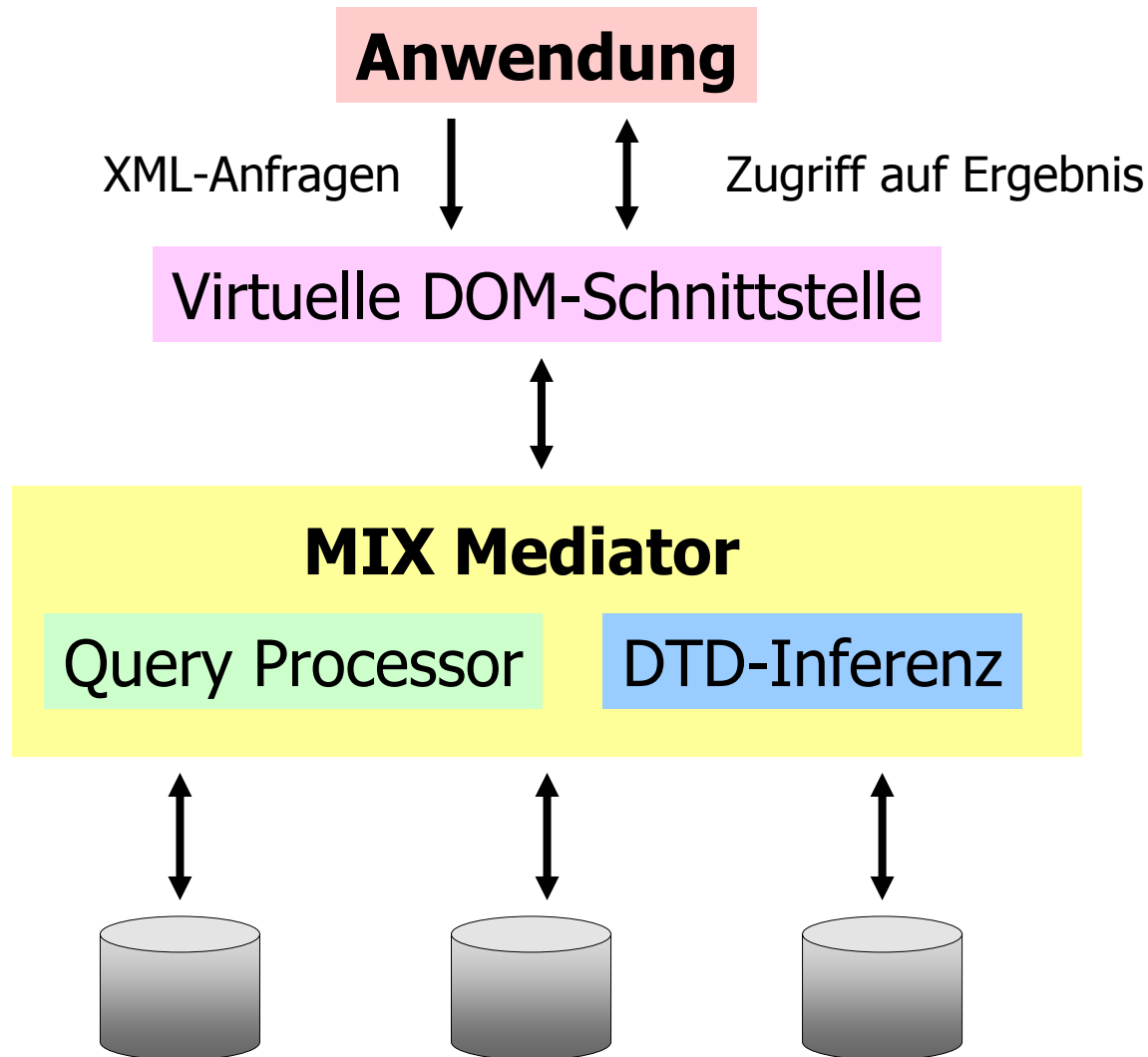
- ▶ nur Baumarkt und Verbraucherportal relevant (Hersteller!)
 - jedes Prädikat wird auf seine Erfüllbarkeit bei Gültigkeit der Anfragefakten überprüft
- ▶ Ermittlung der Kombination: $v_1(H,N,P) \wedge v_2(N,B)$
 - für die Kombinierbarkeit wird überprüft, ob sich die beiden Sichten (zumindest theoretisch) überlappen.
- ▶ Berücksichtigung der Fähigkeiten der Quellen bei der weiteren Planung

Fallbeispiel: MIX



- Zusammenarbeit von UC San Diego und San Diego Supercomputing Center (seit 1999)
- versteht sich als Weiterentwicklung von TSIMMIS
- Nutzung von XML als Integrationsmodell
 - ▶ navigierende Quellenzugriffe auf der Basis von DOM
 - ▶ Ausnutzen von Struktur, wo vorhanden
 - DTD-Inferenz

MIX: Architektur



MIX: Navigierende Zugriffe

- Ergebnisse sollen von den Quellen erst bei Bedarf angefordert werden (*lazy mediator*)
 - ▶ vor allem bei fragmentierten Quellen wie z.B. HTML-Quellen sinnvoll
 - ▶ z.B. Herstellerinformationen
- Modellierung der Mediatoren als Zustandsautomaten
 - ▶ Zustandsübergänge sind Navigationsoperationen auf der DOM-Repräsentation
 - ▶ Aktionen sind Zugriffe auf die Quellen

Fazit



Zusammenfassung

- Unterschiedliche methodische Herangehensweisen an das Problem der Informationsintegration
 - ▶ materialisiert vs. virtuell
- Zentrales Problem: Semantische Integration
 - ▶ materialisierter Ansatz (Globales Schema)
 - ▶ virtueller Ansatz (Domänenmodell bzw. Globales Schema)
 - ▶ Datenaustauschvokabular (vgl. nächste Vorlesung)

Grenzen des Ansatzes auf Informationsebene

- Integration auf Informationsebene ist hervorragend geeignet für Nur-Lese-Zugriffe nach dem Anfrage-Ergebnis-Paradigma
 - ▶ z.B. Integration von Katalogen
- Probleme jedoch bei Schreibzugriffen
 - ▶ z.B. Übertragung von Bestelldaten
 - ▶ Aktualisierungsoperationen auf heterogenen Systemen sehr schwierig auf Datenmodell-/Schemaebene zu lösen
 - ▶ Oft auch Geschäftsprozesse von entscheidender Bedeutung
- Hier braucht man (ergänzend) einen dienstorientierten Ansatz

Weiteres Programm



- Informationsperspektive
 - ▶ Datenaustauschszzenarien
 - ▶ Neuere Ansätze mit Ontologien und Semantic-Web-Technologien
- Dienstperspektive
 - ▶ Dienstorientierte Architekturen und Web Services

■ Zur Vertiefung

- ▶ Abiteboul, Buneman, Suciu: Data on the Web, Morgan Kaufmann, 2000.

■ TSIMMIS

▶ Projekt-Seite

- <http://www-db.stanford.edu/tsimmis/tsimmis.html>

▶ Überblick

- H. Garcia-Molina , Y. Papakonstantinou et al., The TSIMMIS approach to mediation: Data models and Languages, in: Journal of Intelligent Information Systems 1996.
<ftp://www-db.stanford.edu/pub/papers/tsimmis.ps>

▶ Informationsextraktion

- J. Hammer, H. Garcia-Molina, J. Cho, R. Aranha, and A. Crespo: Extracting Semistructured Information from the Web, in: Proceedings of the Workshop on Management of Semistructured Data. Tucson, Arizona, May 1997.
<ftp://www-db.stanford.edu/pub/papers/extract.ps>

■ MIX

- ▶ <http://www.npaci.edu/DICE/MIX/>
- ▶ C. Baru, A. Gupta, B. Ludäscher, R. Marciano, Y. Papakonstantinou, P. Velikhov, V. Chu: XML-Based Information Mediation with MIX, ACM SIGMOD'99, Philadelphia, USA.
<http://www.npaci.edu/DICE/Pubs/sigmod-demo99.pdf>
- ▶ B. Ludäscher, Y. Papakonstantinou, P. Velikhov: A Framework for Navigation-Driven Lazy Mediators, WebDB'99, Philadelphia, USA.
<http://www.npaci.edu/DICE/Pubs/domvxd-webdb.ps.gz>

■ Information Manifold

- ▶ Alon Levy, Anand Rajaraman, Joann Ordille: Query-Answering Algorithms for Information Agents, in: Proceedings of the 13th Nat. Conf. on Artificial Intelligence and the 8th Innovative Applications of Artificial Intelligence Conf., MIT Press, 1996.