Seminar Informationsintegration und Informationsqualität

Produkte und Prototypen

Matthias Käppler

Technische Universität Kaiserslautern

01. Juli 2006

Gliederung

Gliederung

Gliederung

Gliederung

Gliederung

Abgrenzung von Begriffen

Schema Matching: Auffinden von Korrespondenzen zwischen Schemaelementen

Schema Mapping: Formulierung von Abbildungen zwischen Schemaelementen

Schema Merging: Zusammenführen mehrerer (Quell-)Schemata in ein überlappungsfreies (Ziel-)Schema

Schema Integration =

$$\text{Schema Matching} \cup \text{Schema Mapping} \cup \text{Schema Merging}$$

Gliederung

Global-as-View vs. Local-as-View

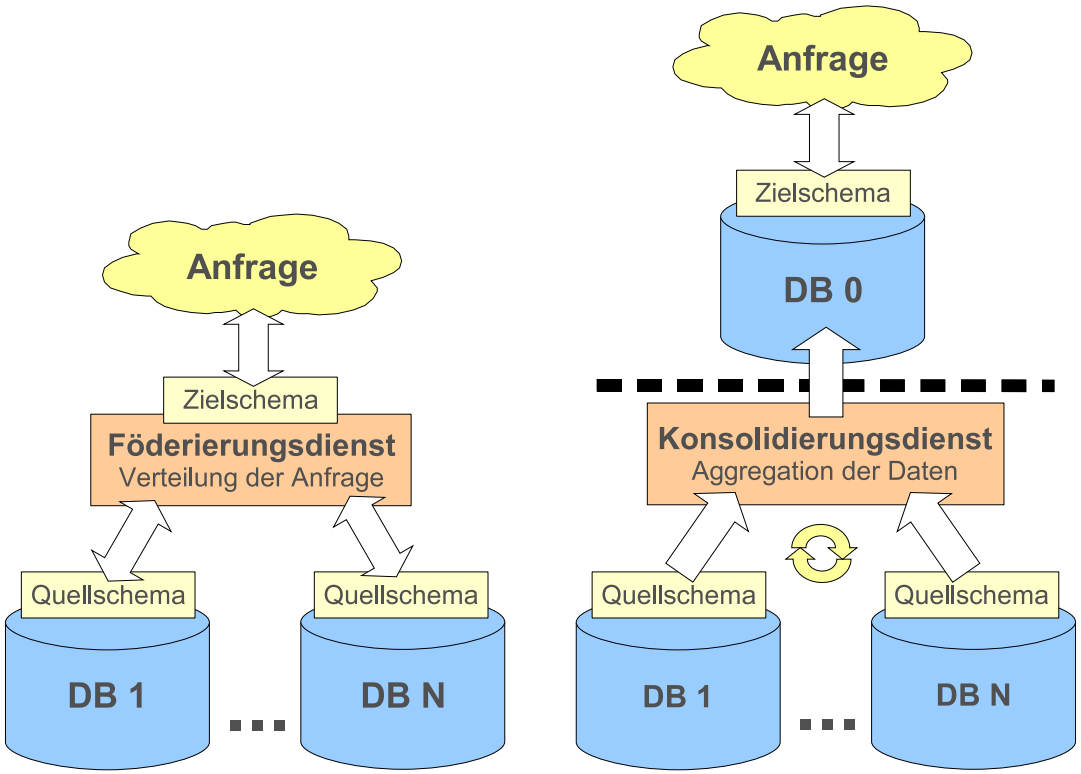
- **Global-as-View (GaV)**
  - Globales Schema wird mittels *Sichten* über die Quellschemata gebildet
  - Anfragetransformation durch *View-Unfolding* $\longrightarrow$ Einfache Handhabung

Global-as-View vs. Local-as-View

- **Global-as-View (GaV)**
  - Globales Schema wird mittels *Sichten* über die Quellschemata gebildet
  - Anfragetransformation durch *View-Unfolding* $\longrightarrow$ Einfache Handhabung
- **Local-as-View (LaV)**
  - Lokale Schemata werden mittels *Sichten* über das globale Schema formuliert
  - Verfahren notwendig, ursprüngliche Anfrage durch Query auf lokales Schema zu beantworten $\longrightarrow$ komplexes Problem
  - Vorteil durch „Entkopplung“ der Quellen vom globalen Schema?

Gliederung

Konsolidierung vs. Föderierung (1)



Konsolidierung vs. Föderierung (2)

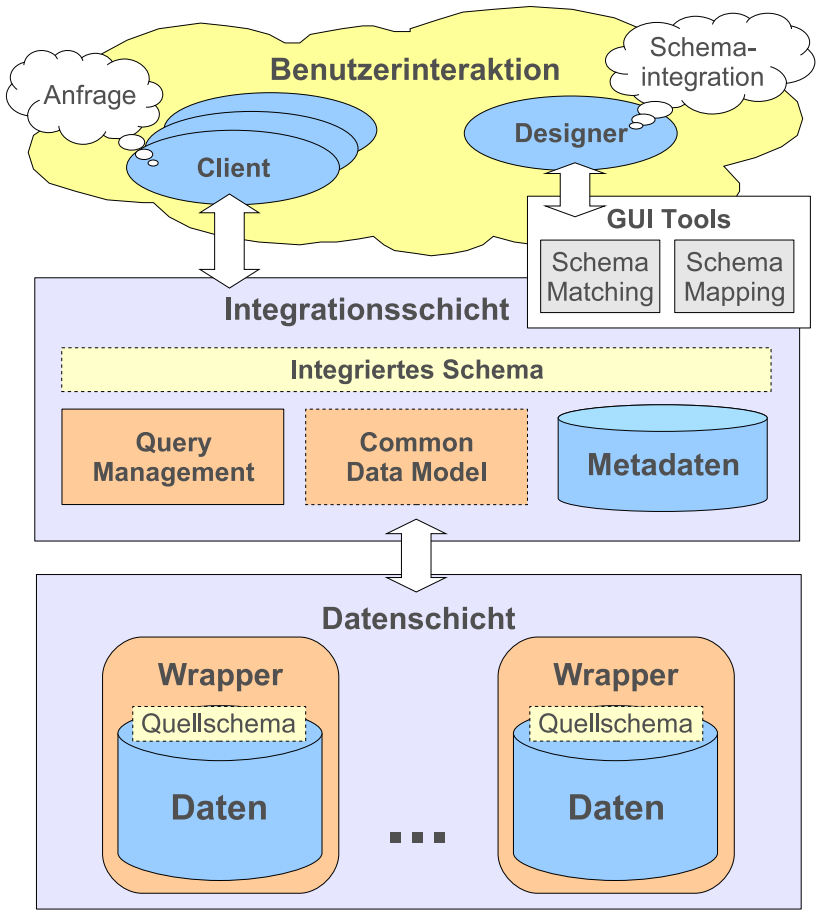
- **Föderierung**
  - Keine Materialisierung des Zielschemas, Datenquellen als logischer Verbund
  - Transformation der Anfragen durch *Mediator*
  - + Aktualität der Daten durch direkte Anfrage an den Quellen
  - − Hohe Latenz durch Verteilung der Quellen
  - − Belastung der Quellen (Ausfall?)

Konsolidierung vs. Föderierung (2)

- **Föderierung**
  - Keine Materialisierung des Zielschemas, Datenquellen als logischer Verbund
  - Transformation der Anfragen durch *Mediator*
  - + Aktualität der Daten durch direkte Anfrage an den Quellen
  - − Hohe Latenz durch Verteilung der Quellen
  - − Belastung der Quellen (Ausfall?)
- **Konsolidierung**
  - Materialisierung des Zielschemas durch Replikation
  - Dadurch Entkopplung der Quellen von der Zieldatenbank
  - + Schnelle und zuverlässige Beantwortung von Anfragen
  - − Komplizierte Update-Algorithmen (Verteilte Systeme!)
  - − Hoher Speicherbedarf (aber: KIWI)

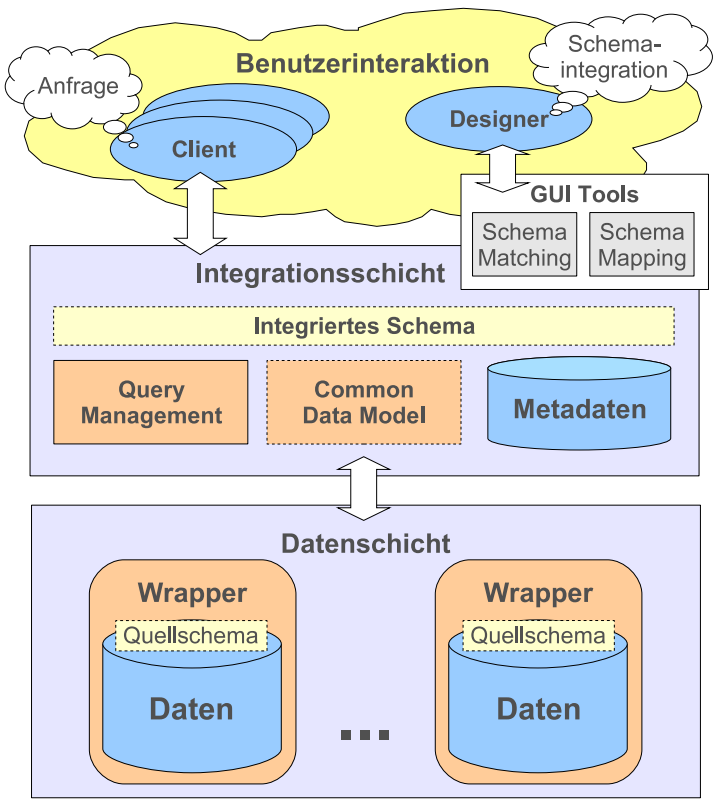
Gliederung

Referenzarchitektur eines föderierten Integrationssystems



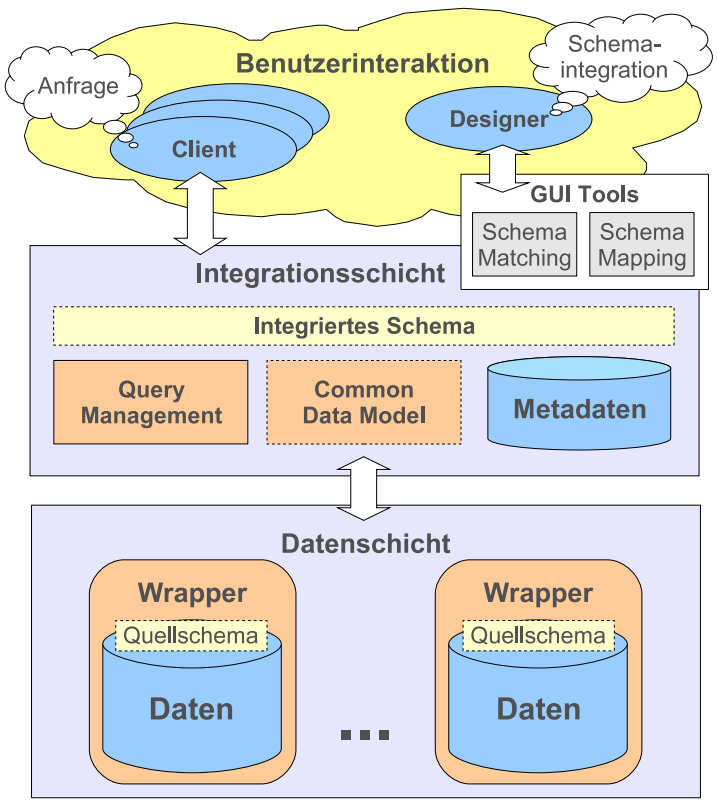
Gliederung

Mögliche Kriterien zur Klassifikation



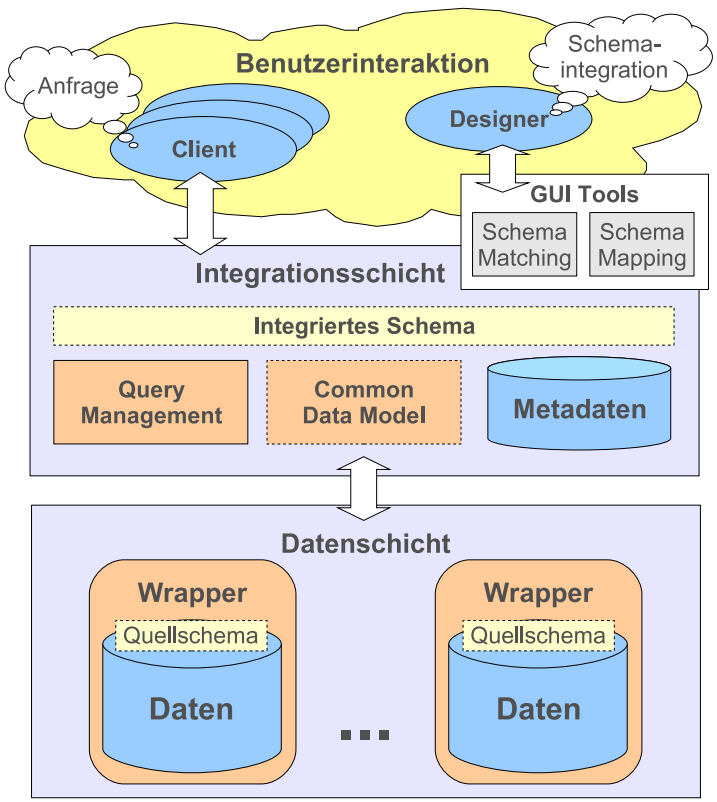
- Common Data Model (CDM)

Mögliche Kriterien zur Klassifikation



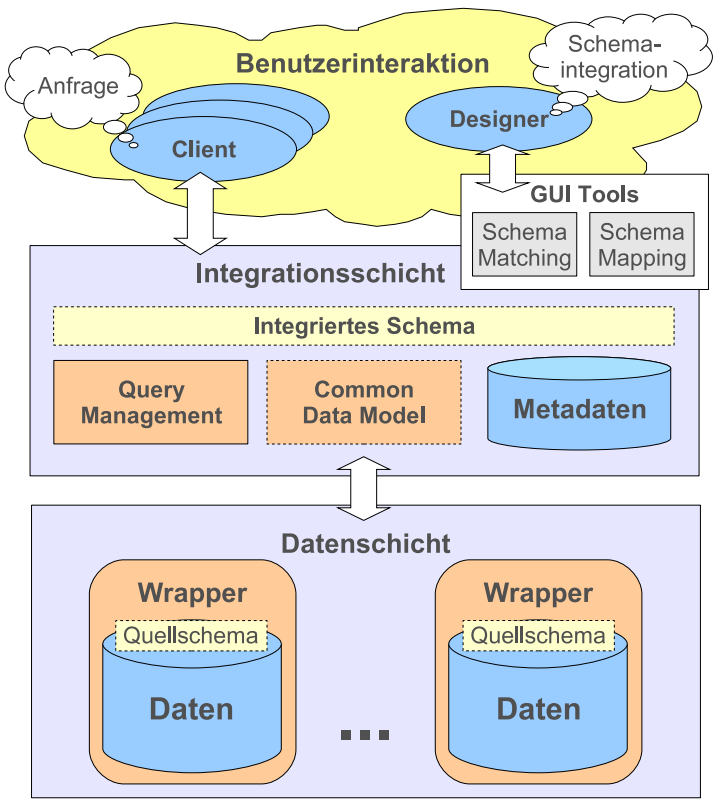
- Common Data Model (CDM)
- Verwendete Anfragesprache

Mögliche Kriterien zur Klassifikation



- Common Data Model (CDM)
- Verwendete Anfragesprache
- Integrationsverfahren

Mögliche Kriterien zur Klassifikation



- Common Data Model (CDM)
- Verwendete Anfragesprache
- Integrationsverfahren
- Grad der Automatisierung

Gliederung

Garlic

- **Einordnung:** Middleware zur Integration heterogener Datenquellen unter einer einzigen logischen Sicht
  $\longrightarrow$ *Global Schema*

Garlic

- **Einordnung:** Middleware zur Integration heterogener Datenquellen unter einer einzigen logischen Sicht
  $\longrightarrow$ *Global Schema*
- Adaption der Daten in den *Repositories* durch Garlic-Wrapper (XML, Relational, Image Data, . . .)
  $\longrightarrow$ *Repository Schemas*
- **Common Data Model:** Objektorientiertes Modell basierend auf dem ODMG-93 Standard
- **Anfragesprache:** GQL

Garlic

- **Einordnung:** Middleware zur Integration heterogener Datenquellen unter einer einzigen logischen Sicht
  $\longrightarrow$ *Global Schema*
- Adaption der Daten in den *Repositories* durch Garlic-Wrapper (XML, Relational, Image Data, . . .)
  $\longrightarrow$ *Repository Schemas*
- **Common Data Model:** Objektorientiertes Modell basierend auf dem ODMG-93 Standard
- **Anfragesprache:** GQL
- **Im Folgenden:** Welche Maßnahmen ergreift Garlic zur Integration der Quellen?

Garlic-Architektur

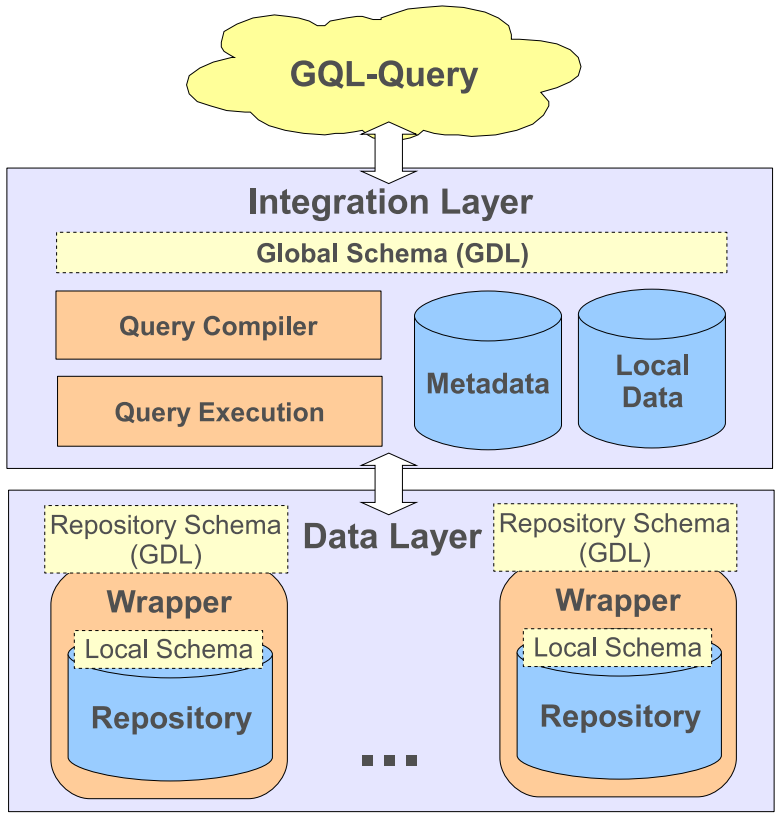


Abbildung an den Wrappern

- Bereitstellung der Daten als *Garlic-Objects* an den Schnittstellen der Wrapper
- Dazu...
  - Formulierung von Interface-Definitionen in der GDL
  - Definition einer oder mehrerer Implementierungen eines Interfaces
- Objekte global eindeutig identifizierbar durch *Garlic-Objekt-ID*

Abbildung an den Wrappern

- Bereitstellung der Daten als *Garlic-Objects* an den Schnittstellen der Wrapper
- Dazu...
  - Formulierung von Interface-Definitionen in der GDL
  - Definition einer oder mehrerer Implementierungen eines Interfaces
- Objekte global eindeutig identifizierbar durch *Garlic-Objekt-ID*

Beispiel relationaler Wrapper

```
create table Fachbereich (
    Fbnr integer primary key,
    Name varchar(30) not null
);
```

```
interface FB_Type {
    attribute long Fbnr;
    attribute string Name;
};
```

Abbildung in der Middleware

- Erweiterung des ODMG-Modells durch *Object-Centered Views*
- *Virtuelle Objekte* bilden Umformungen zugrundeliegender Garlic-Objekte
- Achtung: Virtuelle Objekte sind *immateriell*, da lediglich als Anfragen auf die Quellen realisiert

Abbildung in der Middleware

- Erweiterung des ODMG-Modells durch *Object-Centered Views*
- *Virtuelle Objekte* bilden Umformungen zugrundeliegender Garlic-Objekte
- Achtung: Virtuelle Objekte sind *immateriell*, da lediglich als Anfragen auf die Quellen realisiert

Beispiel

```
create view InfStatistik (Matnr, Semester, Schnitt, self)
as select S.Matnr, S.Semester, avg(Z.Note),
    LIFT('InfStatistik', S.OID)
from Student S, Zensuren Z
where S.fb->Name = 'Informatik' and S.Matnr = Z.Matnr
group by S.Matnr, S.Semester
```

Anfrageplanung

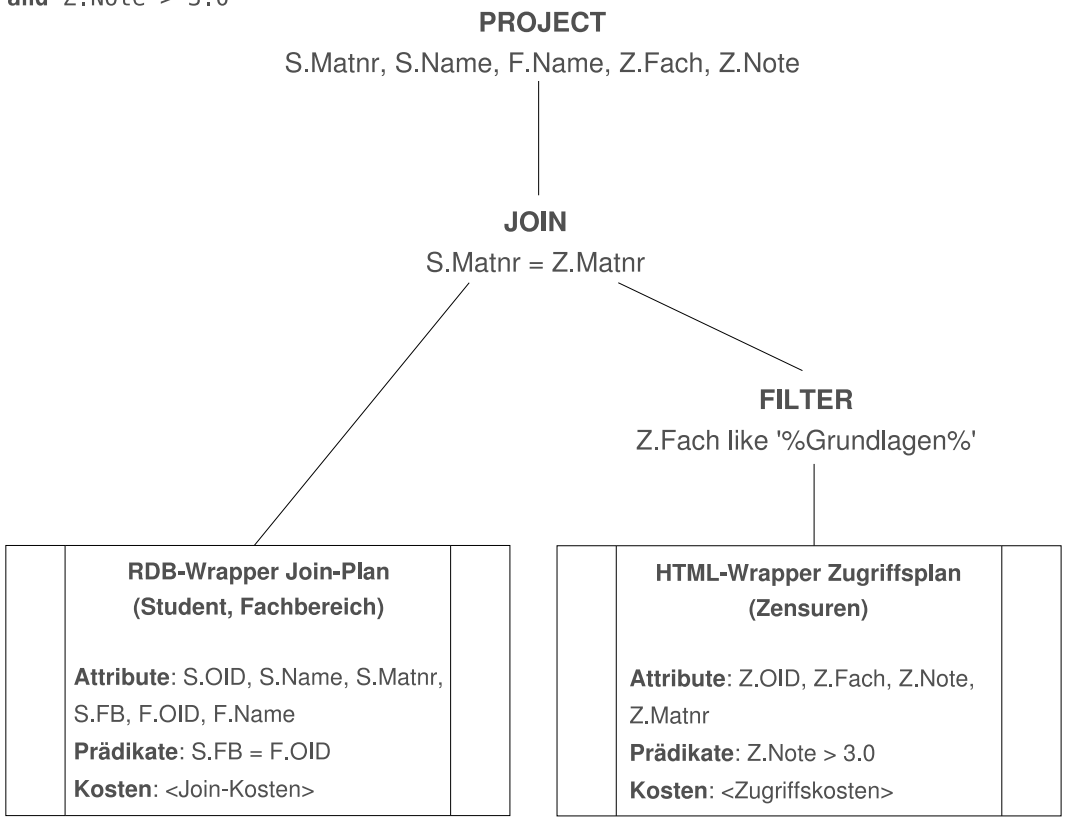
- **Ziel:** Erzeugung mehrerer Anfragepläne und Auswahl des (kosten)effizientesten

Anfrageplanung

- **Ziel:** Erzeugung mehrerer Anfragepläne und Auswahl des (kosten)effizientesten
- Bottom-Up-Erzeugung verschiedener Pläne durch Senden von *Work Requests* an die Wrapper:
  - Single Collection Access Plans
  - Join Plans
  - Bind Plans
- Erzeugung eines vollständigen Anfrageplans aus den Teilplänen der Wrapper
- Kompensation evtl. nicht vorhandener oder ineffizienter Funktionalität der Wrapper durch POPs

Garlic-Anfrageplan

```
select S.Matnr, S.Name, F.Name, Z.Fach, Z.Note
from Student S, Fachbereich F, Zensuren Z
where S.FB = F.OID
and Z.Fach like '%Grundlagen%'
and Z.Matnr = S.Matnr
and Z.Note > 3.0
```



Gliederung

IBM DB2 Information Integrator

- **Einordnung:** Analog zu Garlic, IBMs Produktivsystem basierend auf Garlic und der DB2 Universal Database

IBM DB2 Information Integrator

- **Einordnung:** Analog zu Garlic, IBMs Produktivsystem basierend auf Garlic und der DB2 Universal Database
- Zahlreiche **Wrapper** verfügbar zur Integration strukturierter, semi-strukturierter und unstrukturierter Quellen:
  - **Relationale Wrapper:** Oracle, Informix, Sybase, SQL Server, Teradata, ...
  - **Nicht-relationale Wrapper:** XML, Plain Text, Excel Spreadsheets, BLAST, Documentum, ...
- **Features:**
  - Distributed Joins
  - Pushdown von Operationen
  - Kompensierungsmaßnahmen
  - Replikation von Daten
  - Voller Schreibzugriff auf relationale Quellen

Architektur, Komponenten, Begriffe

- Ein durch den DB2II föderiertes DB2-System besteht aus . . .
  - . . . einer DB2 UDB Instanz
  - . . . einem globalen Katalog
  - . . . den Servern
  - . . . sowie den Wrappern

Architektur, Komponenten, Begriffe

- Ein durch den DB2II föderiertes DB2-System besteht aus ...
  - ...einer DB2 UDB Instanz
  - ...einem globalen Katalog
  - ...den Servern
  - ...sowie den Wrappern
- Bekanntmachung der lokalen Objekte der Server durch *Nicknames*
  $\longrightarrow$ Völlige Transparenz für Benutzer

Architektur, Komponenten, Begriffe

- Ein durch den DB2II föderiertes DB2-System besteht aus ...
  - ...einer DB2 UDB Instanz
  - ...einem globalen Katalog
  - ...den Servern
  - ...sowie den Wrappern
- Bekanntmachung der lokalen Objekte der Server durch *Nicknames*
  $\rightarrow$ Völlige Transparenz für Benutzer
- **Wesentliche Unterschiede zu Garlic:** Ablegen der Daten in Relationen; Anfragesprache ist SQL

Der Integrationsprozess im Überblick

- Um eine Datenquelle in die Föderation zu integrieren sind folgende Schritte durchzuführen:
  1. Registrierung des Wrapper-Moduls
  2. Anmeldung des Servers
  3. Anlegen von User Mappings
  4. Testen der Verbindung via Passthru-Sessions
  5. Falls nötig/erwünscht, Anlegen weiterer Daten-Mappings
  6. Anlegen von Nicknames

Der Integrationsprozess im Überblick

- Um eine Datenquelle in die Föderation zu integrieren sind folgende Schritte durchzuführen:
  1. Registrierung des Wrapper-Moduls
  2. Anmeldung des Servers
  3. Anlegen von User Mappings
  4. Testen der Verbindung via Passthru-Sessions
  5. Falls nötig/erwünscht, Anlegen weiterer Daten-Mappings
  6. Anlegen von Nicknames
- **Im Folgenden:** Schritte 1, 2 und 6 am Beispiel einer XML-Quelle

Registrierung von Wrapper-Modulen und Servern

- Wrapperimplementierungen werden dem System in Form von *Wrapper-Modulen* bereitgestellt
- Wrapper-Modul implementiert Routinen zum Verbindungsaufbau und Datenaustausch
- Erzeugung eigener Wrapper-Module durch *Wrapper-Development-Kit* möglich
- Im Anschluss Datenquelle als *Server* dem System bekannt machen

Registrierung von Wrapper-Modulen und Servern

- Wrapperimplementierungen werden dem System in Form von *Wrapper-Modulen* bereitgestellt
- Wrapper-Modul implementiert Routinen zum Verbindungsaufbau und Datenaustausch
- Erzeugung eigener Wrapper-Module durch *Wrapper-Development-Kit* möglich
- Im Anschluss Datenquelle als *Server* dem System bekannt machen

Beispiel

```
CONNECT TO <federated_db_name>;
CREATE WRAPPER XML LIBRARY 'libdb2lsxml.a'
CREATE SERVER MY_XML_SERVER WRAPPER XML
```

Erzeugen von Nicknames

- Nicknames dienen der Referenzierung von Objekten in den Datenquellen
  $\longrightarrow$ ortstransparenter Zugriff
- Abbildung von Nicknames auf relationale Quellen unproblematisch...
- ...und sonst?

Erzeugen von Nicknames

- Nicknames dienen der Referenzierung von Objekten in den Datenquellen
  $\longrightarrow$ ortstransparenter Zugriff
- Abbildung von Nicknames auf relationale Quellen unproblematisch...
- ...und sonst?

Beispiel

```
CONNECT TO <federated_db_name>;
CREATE NICKNAME XMLSCHEMA.STUDENT (
    MATNR CHAR(6) NOT NULL OPTIONS(XPATH './matnr/text()'),
    NAME VARCHAR(30) NOT NULL OPTIONS(XPATH './name/text()'),
    VORNAME VARCHAR(30) NOT NULL OPTIONS(XPATH './vname/text()')
    FOR SERVER "MY_XML_SERVER"
    OPTIONS(XPATH '//student', FILE_PATH '/exchange/xml/studenten.xml')
```

Gliederung

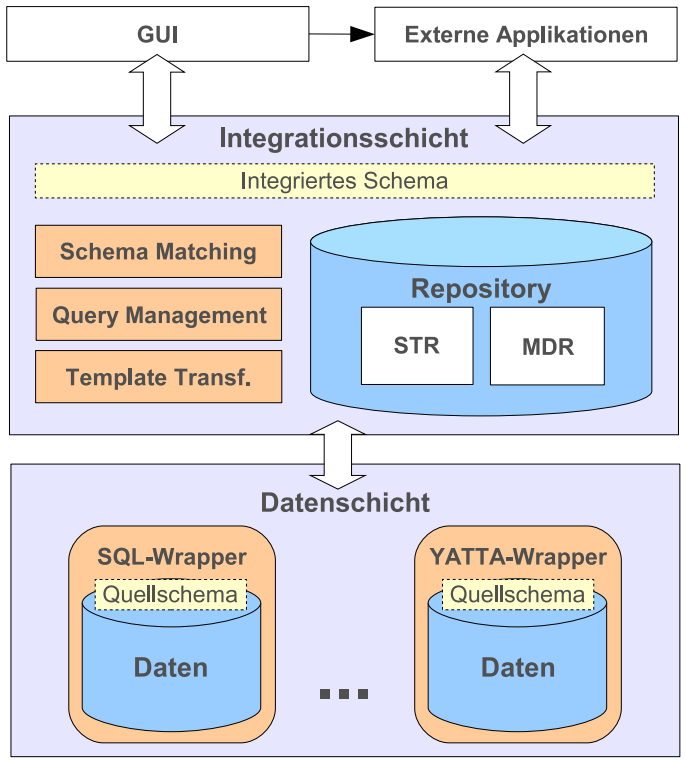
AutoMed

- **Einordnung:** Middleware zur Integration heterogener Datenquellen

AutoMed

- **Einordnung:** Middleware zur Integration heterogener Datenquellen
- Führt neues Integrationsverfahren ein: *Both-as-View*
- Design-Paradigma hinter AutoMed: Kleinster, gemeinsamer Nenner für hohen Grad an Generizität
- **Common Data Model:** HDM (Hypergraph Data Model)
  $\longrightarrow$ Graph-basiert
- **Query Language:** IQL (Intermediate Query Language)
  $\longrightarrow$ Funktionale Sprache

AutoMed Architektur



Both-as-View (BaV) (1)

- Überführung eines Schemas A in ein Schema B durch Formulierung einer Folge *schrittweiser Transformationen*
- Jeder Transformationsschritt beinhaltet eine einzige Transformation (Hinzufügen, Umbenennen, Löschen von Elementen)
- Die Abbildungsfolge ist bidirektional, kann also jederzeit umgekehrt werden

Both-as-View (BaV) (1)

- Überführung eines Schemas A in ein Schema B durch Formulierung einer Folge *schrittweiser Transformationen*
- Jeder Transformationsschritt beinhaltet eine einzige Transformation (Hinzufügen, Umbenennen, Löschen von Elementen)
- Die Abbildungsfolge ist bidirektional, kann also jederzeit umgekehrt werden

Es folgt ein Beispiel . . .

Both-as-View (BaV) (2)

- **Gesucht:** Zielrelation `person` mit Attributen `id` und `name`, die sich vollständig in die Unterklassen `male` und `female` aufgliedert
- **Gegeben:** Relation `staff` mit Attributen `id`, `name` und `gender` (mit Werten 'm' für male und 'f' für female).

Transformationsfolge

1. $renameEntity(\langle\langle staff\rangle\rangle, \langle\langle person\rangle\rangle)$
2. $addEntity(\langle\langle male\rangle\rangle,\ [\{x\} \mid \{x, y\} \leftarrow \langle\langle person, gender\rangle\rangle;\ (=)\ y\ 'm'])$
3. $addEntity(\langle\langle female\rangle\rangle,\ [\{x\} \mid \{x, y\} \leftarrow \langle\langle person, gender\rangle\rangle;\ (=)\ y\ 'f'])$
4. $addGeneralisation(gender, total, person, male, female)$
5. $delAttribute(\langle\langle person, gender\rangle\rangle,$
   $([\{x, y\}|\{x\} \leftarrow \langle\langle male\rangle\rangle; (=)\ y\ 'm'] + +[\{x, y\}|\{x\} \leftarrow \langle\langle female\rangle\rangle; (=)\ y\ 'f']),$
   $(['m','f'] = [\{y\}|\{x, y\} \leftarrow \langle\langle person, gender\rangle\rangle]))$

Gliederung

Clio

- **Bisher:** Nur Transformation der Quell-Datenmodelle durch Wrapper in das CDM der Middleware

Clio

- **Bisher:** Nur Transformation der Quell-Datenmodelle durch Wrapper in das CDM der Middleware
- **Frage:** Wie gehen die Konstrukte der Wrapper in das Zielschema ein? $\rightarrow$ *Schema Mapping*

Clio

- **Einordnung:** Prototyp für semi-automatisches Schema-Matching- und Schema-Mapping-Werkzeug

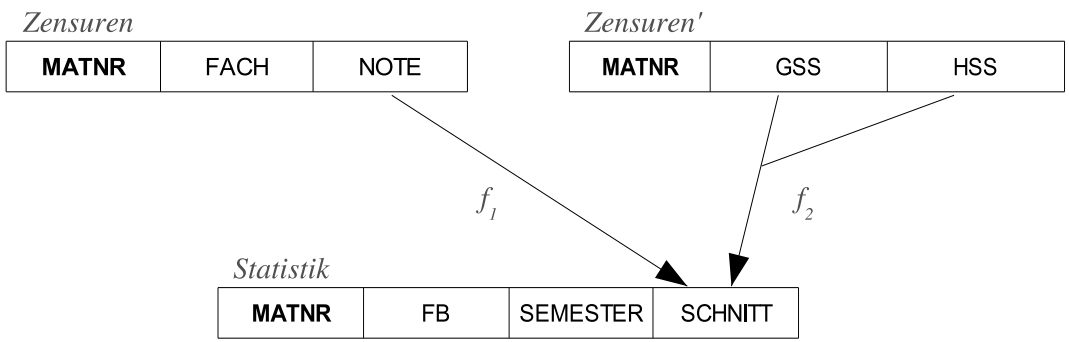
Clio

- **Einordnung:** Prototyp für semi-automatisches Schema-Matching- und Schema-Mapping-Werkzeug
- Semi-automatische Discovery von Matches (Korrespondenzen) als *Grundlage* für ein Schema Mapping
- Erzeugen von Mappings durch Formulierung von Anfragen (GaV)
- Quell- und Zielschemata im relationalen oder XML-Modell $\rightarrow$ Mappings in SQL, SQL/XML, XQuery, XSLT
- Technologische Grundlage für IBM Rational Data Architect

Wertbasierte Korrespondenzen

- Ermöglichen Definition komplexer Mappings zwischen zwei Schemaelementen
- Dienen als Input für das logische Mapping (in Form von Assertions/Constraints)
- Definition durch eine Funktion f und einem Filter F (hier $F = true$):

$$f_1 : avg(Zensuren(Note)) \rightarrow Statistik(Schnitt)$$
$$f_2 : (Zensuren'(GSS) + Zensuren'(HSS)) * 0.5 \rightarrow Statistik(Schnitt)$$



Physisches Mapping für eine Zielrelation (4 Phasen)

1. Bildung von *Potential Candidate Sets (PCS)*, die (nicht notwendigerweise disjunkt) jeweils *eine* Möglichkeit bilden, die Zielrelation (teilweise) zu erzeugen

Physisches Mapping für eine Zielrelation (4 Phasen)

1. Bildung von *Potential Candidate Sets (PCS)*, die (nicht notwendigerweise disjunkt) jeweils *eine* Möglichkeit bilden, die Zielrelation (teilweise) zu erzeugen
2. Erzeugen von *Candidate Sets (CS)* durch Eliminieren von „schlechten“ PCS

Physisches Mapping für eine Zielrelation (4 Phasen)

1. Bildung von *Potential Candidate Sets (PCS)*, die (nicht notwendigerweise disjunkt) jeweils *eine* Möglichkeit bilden, die Zielrelation (teilweise) zu erzeugen
2. Erzeugen von *Candidate Sets (CS)* durch Eliminieren von „schlechten“ PCS
3. Finden von Untermengen der Menge aller CS, die alle definierten Korrespondenzen vollständig und minimal überdecken → Minimal Cover (MC)
   - Mehr als ein MC: Durchführen eines Rankings, um möglichst „gute“ Abdeckung zu finden

Physisches Mapping für eine Zielrelation (4 Phasen)

1. Bildung von *Potential Candidate Sets (PCS)*, die (nicht notwendigerweise disjunkt) jeweils *eine* Möglichkeit bilden, die Zielrelation (teilweise) zu erzeugen
2. Erzeugen von *Candidate Sets (CS)* durch Eliminieren von „schlechten“ PCS
3. Finden von Untermengen der Menge aller CS, die alle definierten Korrespondenzen vollständig und minimal überdecken → Minimal Cover (MC)
   - Mehr als ein MC: Durchführen eines Rankings, um möglichst „gute“ Abdeckung zu finden
4. Erzeugen der Anfragen (`SELECT-FROM-WHERE`) und `UNION ALL` der Ergebnisse

Gliederung

IBM Rational Data Architect

- Datenmodellierungs- und Integrationswerkzeug von IBM auf Basis von Clio
- RDA ermöglicht:
  - Modellierung,
  - Annotation und
  - Integration

  von Datenquellen
- Verwendet populäre Technologie: Eclipse Plattform, JDBC

IBM Rational Data Architect

- Datenmodellierungs- und Integrationswerkzeug von IBM auf Basis von Clio
- RDA ermöglicht:
  - Modellierung,
  - Annotation und
  - Integration

  von Datenquellen
- Verwendet populäre Technologie: Eclipse Plattform, JDBC
- **Begrifflichkeiten:**
  - *Logisches Modell:* Modell eines DB-Schemas als Entity-Relationship-Diagramm (nicht DB-spezifisch)
  - *Physisches Modell:* Konkrete Realisierung eines logischen Modells (DB-spezifisch)

Typischer Integrationsprozess des RDA

- Ablauf einer Integrations-Session beinhaltet i.d.R. folgende Schritte:
  1. Annotieren der zu integrierenden Schemata
  2. Auffinden/Definieren von Matches zwischen den Quellschemas
  3. Modellierung des Zielschemas
  4. Auffinden/Definieren von Matches zwischen Quellschemas und Zielschema
  5. Erzeugen der Mappings in Form von Anfragen

Typischer Integrationsprozess des RDA

- Ablauf einer Integrations-Session beinhaltet i.d.R. folgende Schritte:
  1. Annotieren der zu integrierenden Schemata
  2. Auffinden/Definieren von Matches zwischen den Quellschemas
  3. Modellierung des Zielschemas
  4. Auffinden/Definieren von Matches zwischen Quellschemas und Zielschema
  5. Erzeugen der Mappings in Form von Anfragen
- **Im Folgenden:** Schritte 1, 2 und 5 im Detail

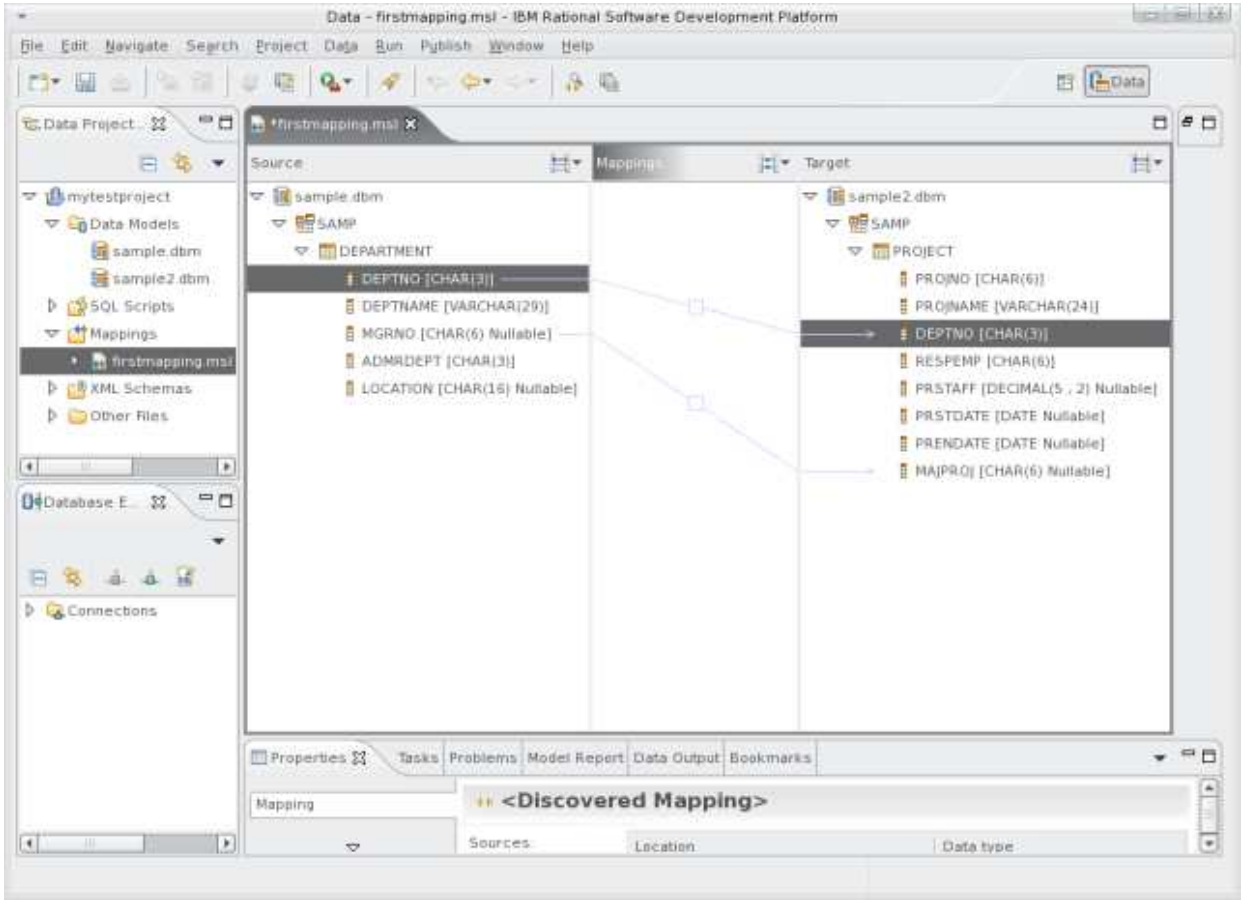
Annotation der Quellschemas

- Aufbau einer Verbindung zu den Datenquellen, anschließend Zugriff über Database Explorer möglich
- Anlegen eines physischen Modells der Datenquelle (kann Abstraktion sein)
- Annotation der Schemaelemente (Tabellen, Spalten, Constraints, Trigger, . . .)
  - Textuelle Beschreibung
  - Ausführlicher Name (im Gegensatz zu Abkürzung)
  - Visuelle Kontextmodelle
  - Glossar

Finden/Erzeugen von Matches zwischen den Quellschemas

- Def. Mapping im RDA: Explizierung einer in den Schemata *nicht* explizit kodierten Korrespondenz zwischen zwei Schemaelementen
- Erzeugen von *Mappings* und *Mapping Models* durch
  - Mapping Discovery (automatisches Auffinden von Korrespondenzen, z.B. durch Heranziehen des Glossars)
  - Manuelle Definition
  - Hinzufügen von Transformationen zu Mappings
- Falls nötig oder erwünscht, weitere Annotationen vornehmen

RDA Mapping Model



Erzeugen der Mappings als Anfragen

- **Voraussetzung:** Es muss ein einzelnes Mapping Model existieren, mit dem Zielschema auf der rechten Seite
  $\longrightarrow$ *Kombinieren aller bisher erzeugten Mapping Models*
- Noch vorhandene Konfliktsituationen müssen aufgelöst werden
- Erzeugen eines SQL- bzw. SQL/XML-Skripts, welches durch Formulierung von Anfragen, Sichten oder Inserts das Mapping realisiert

Ein Rückblick auf die Systeme (1)

- **Common Data Model?**
  - Garlic: ODMG Object Model + X
  - DB2II: Relationenmodell
  - AutoMed: HDM
  - Clio/RDA: Internes Zwischenformat

Ein Rückblick auf die Systeme (1)

- **Common Data Model?**
  - Garlic: ODMG Object Model + X
  - DB2II: Relationenmodell
  - AutoMed: HDM
  - Clio/RDA: Internes Zwischenformat
- **Anfragesprachen?**
  - Garlic: GQL
  - DB2II: SQL
  - AutoMed: IQL als Grundlage
  - Clio/RDA: Impliziert SQL bzw. eine XML-QL (z.B. XQuery, DOM, ...)

Ein Rückblick auf die Systeme (2)

- **Integrationsverfahren?**
  - Garlic: GAV
  - DB2II: GAV
  - AutoMed: BAV
  - Clio/RDA: GAV

Ein Rückblick auf die Systeme (2)

- **Integrationsverfahren?**
  - Garlic: GAV
  - DB2II: GAV
  - AutoMed: BAV
  - Clio/RDA: GAV
- **Grad der Automatisierung?**
  - . . . ist bei allen Lösungen ausbaufähig

Ein Rückblick auf die Systeme (2)

- **Integrationsverfahren?**
  - Garlic: GAV
  - DB2II: GAV
  - AutoMed: BAV
  - Clio/RDA: GAV
- **Grad der Automatisierung?**
  - . . . ist bei allen Lösungen ausbaufähig
- **Nicht betrachtet:**
  - ETL-Werkzeuge
  - Data-Cleaning-Werkzeuge

Wie war das mit...?