0.1 Introduction

Definition

Provide uniform interface to various data sources

0.1.1 Examples of Data Integration

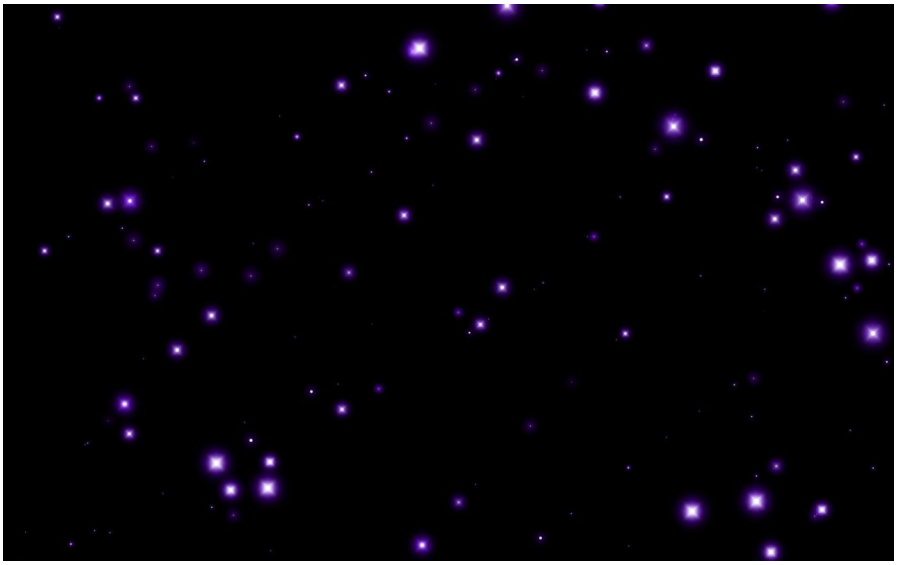
Motivational Relational Example

Schema1 <http://example1.org> Schema2 <http://example2.org>

Student
fullName
takesCourse
creditScore

Student	Enrollment
firstName	semester
secondName	hasCourse
hasEnrollment	
creditScore	

Data Isolation



Data Isolation - Heterogeneity

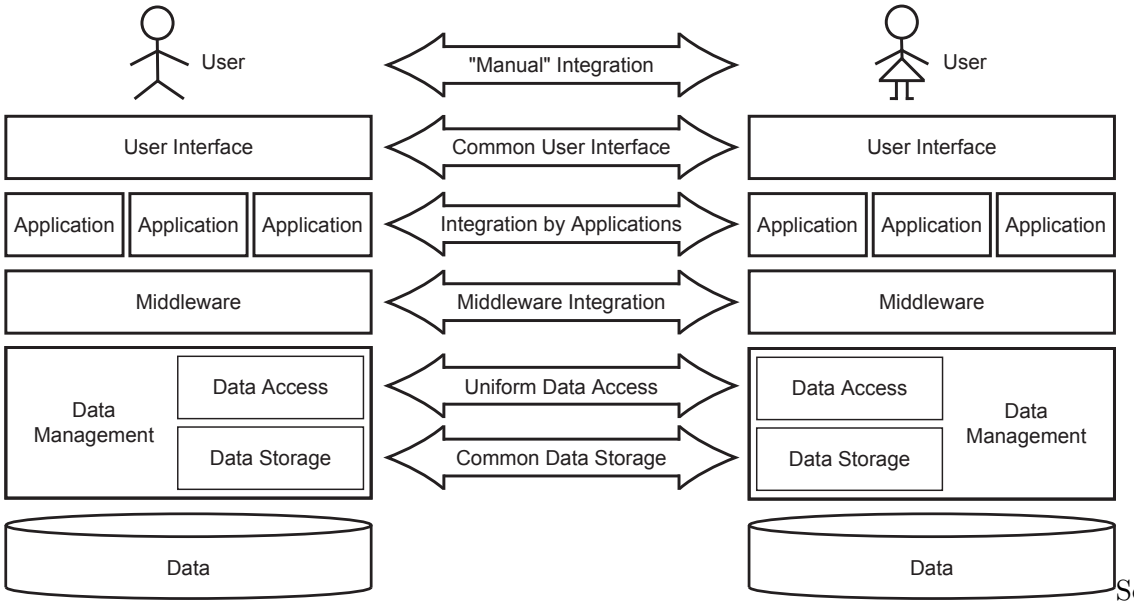
Data isolation is caused mainly by information system component heterogeneity

- hardware and operating systems
- data management software
- **models, schemas, and data semantics**
- **Data modeling methods**
- middleware
- user interfaces
- business rules and integrity constraints.

See [**Ziegler07dataintegration**]

Information System Architecture

Integrating data from different information systems can be performed on different ar-



chitectural levels. See [**Ziegler07dataintegration**]

Integration Solution Categorization

- **Manual Integration** - users need to have detailed knowledge on different user interfaces, query languages, logical data representation and semantics.
- **Common User Interface** - the user is supplied with a common user interface that provides a uniform look and feel. Data is still separately presented so that homogenization and integration of data yet has to be done by the users.
- **Integration by Applications** - custom application merges and presents the data to the end user. Practical only for a small number data sources.
- **Uniform Data Access** - data sources are logically integrated at the data access level. Global applications are provided with a unified global view of physically distributed data. Accessing physically integrated data can be time-consuming since data access, homogenization, and integration have to be done at runtime.
- **Common Data Storage** - data is transferred to a new data storage; local sources can either be retired or remain operational. Physical data integration provides fast data access. Periodical refreshing of the common data storage needs to be considered.

See [**Ziegler07dataintegration**]

Data Integration Specification Aspects

Data integration problems have the same goal but they solution differs depending on the problem specification. Generally, solving data integration problems concerns the following aspects:

- architectural view of an information system
- content and functionality of the component systems,
- kind of information that is managed by component systems (alphanumeric data, multimedia data; structured, semi-structured, unstructured data),
- requirements concerning autonomy of component systems,
- security and privacy
- integrated data quality (freshness, consistency, duplication, data semantics)
- intended use of the integrated information system,
- performance requirements, and
- available resources (time, money, human resources, know-how, etc.

See [**Ziegler07dataintegration**]

Semantic integration

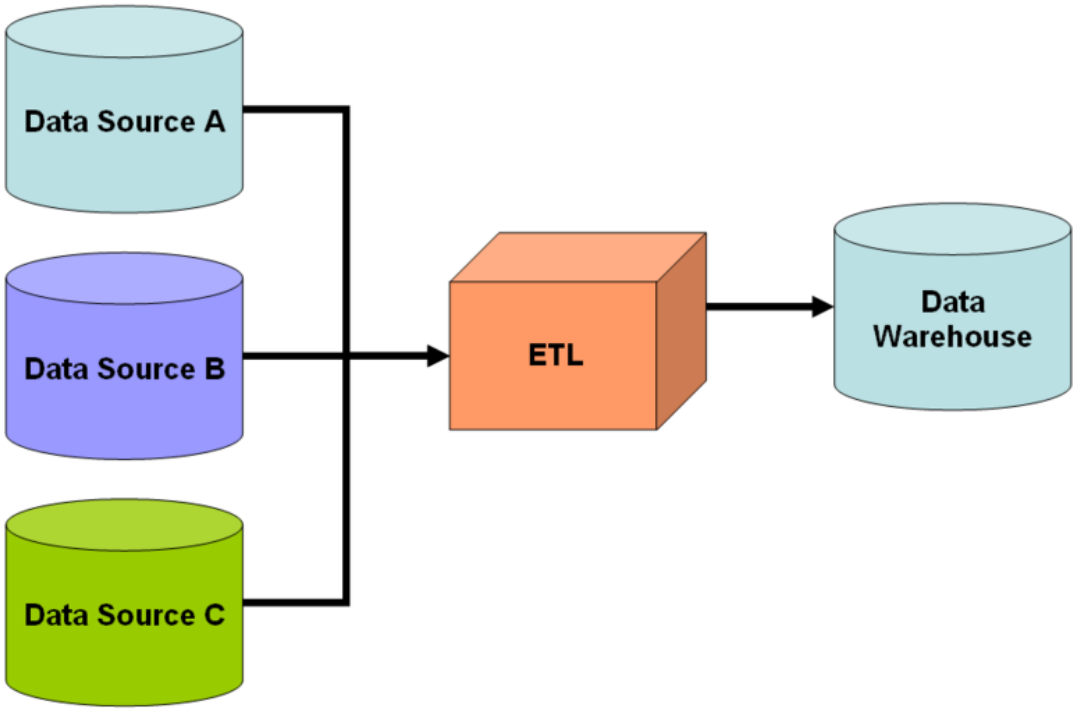
The most fundamental problem of data integration remains the semantic aspect of data. People tend to use terms with an implicitly meaning, e.g. the context specifies the meaning. Here are some examples :

- a filed with the title *height* might be in centimeters or inches
- Does *Fish* belong to the category *Meat* - some people think "yes" because both are biologically muscles. Other think "no" and their explanation is from the perspective of food categories where the category *Meat* is more specific.

0.1.2 Solution approaches

Common Data Storage Solutions - Data Warehouse

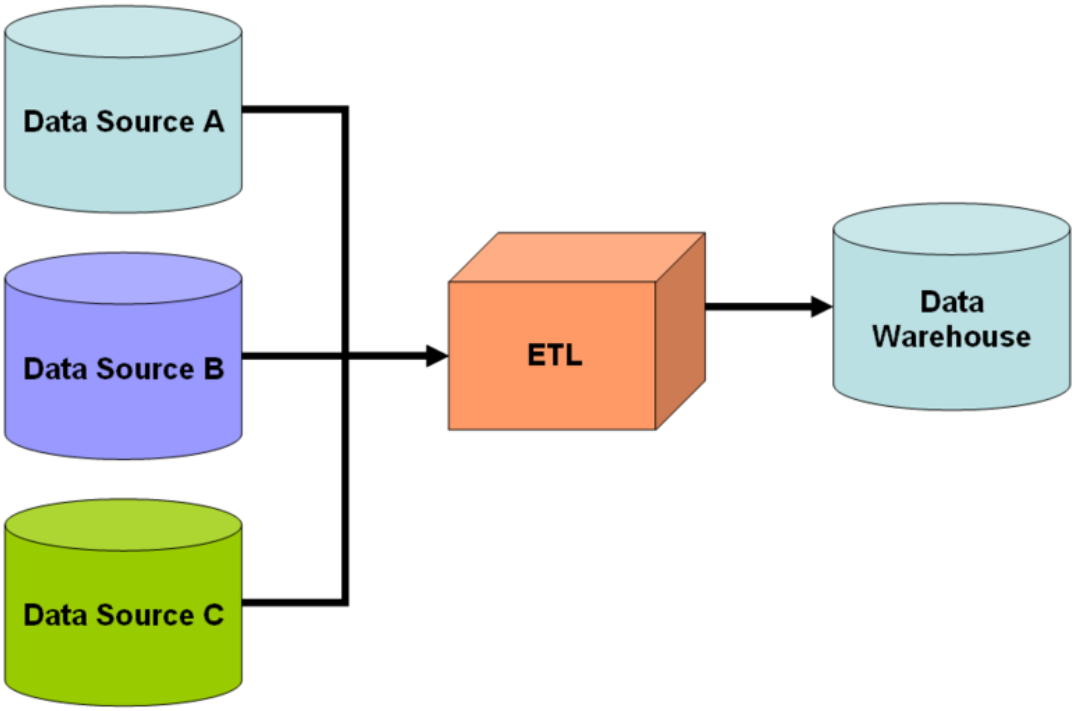
- provides common read-only data storage
- approach - Extract Transform Load (ETL)
- pros - Fast query answering, history snapshot
- cons - data freshness (refreshing, creating new snapshot), reusability



Common Data Storage Solutions - Operational data stores

Operational Data Sources - warehouse with fresh data

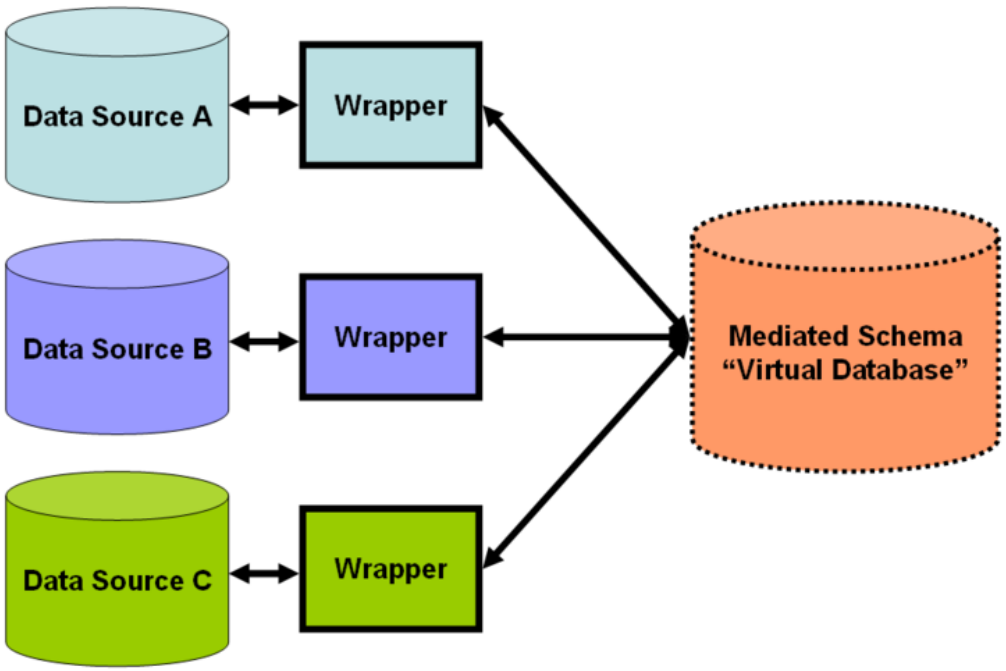
- provides common read-only data storage
- approach Extract Transform Load (ETL)
- pros - Fast query answering,
- cons - data freshness, reusability



Uniform Data Access Solutions

Data is integrated on logical and, or conceptual schema.

- Logic Programming - extensional assertions
- Description Logics - intentional assertions
- pros - data freshness, reusability
- cons - query answering



0.1.3 Technologies

Technologies

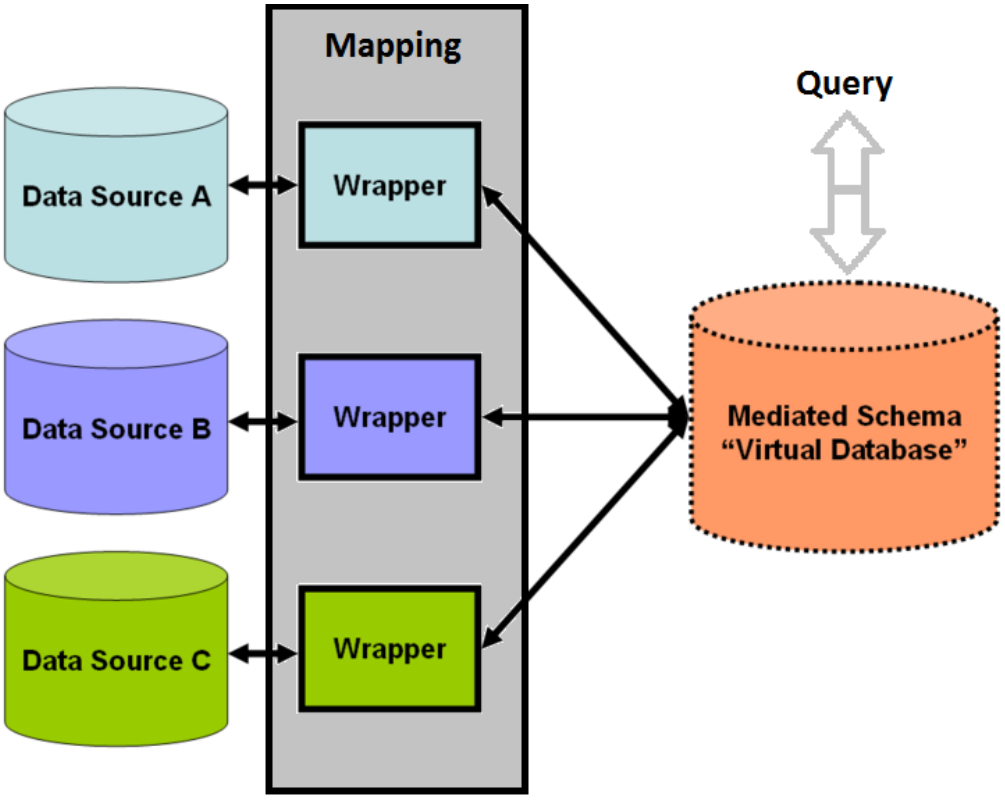
- Ontology based - Conceptual modeling + SWRL rules
- SPARQL DESCRIBE clause
- Expressive query languages - SQWRL, SPARQL-DL
- AL-log: integrating Datalog and description logics [**dlns1998aliddl**]
- OWLIM - OWL DLP (intersection between DL and Datalog) [**Kiryakov2005**]
- D2R - Relational data as RDF [**d2r**]

0.2 Data Integration Theory

Aspects of Data Integration

- data sources content specification

- how to query integrated data



Conceptual Level Specification

Data integration - relational databases

Definition
A data integration system $\mathcal{DI}$ is a triple $\langle \mathcal{G}, \mathcal{S}, \mathcal{M} \rangle$, where

- $\mathcal{G}$ is the global schema
- $\mathcal{S}$ is the source schema
- $\mathcal{M}$ is the mapping between $\mathcal{S}$ and $\mathcal{G}$

Semantics of $\mathcal{DI}$: which are the databases (rdf graphs) of $mathcalDI$

Definition
Let $\mathcal{C}$ be a source database $sem^{\mathcal{C}}(\mathcal{DI}) = \{\mathcal{B} | \mathcal{B}$ is legal wrt $\mathcal{G}$ and staisfies $\mathcal{M}$ wrt $\mathcal{C}\}$

Specification of the Schema Mapping $\mathcal{M}$

- **Local as View (LAV)** - the source schemas are represented as views using the concepts and relationships in the global schema. This approach is the most flexible. New data sources are integrated by simply representing them in terms of the global schema.
- **Global as View (GAV)** - the global schema is represented as views using the concepts and relationships of the source schema. This approach is suitable for simple data integration tasks.
- **mixed apprach GLAV**
- **Peer to Peer (P2P)** - suitable only for small amount of data sources, each pair of sources must have their own mapping.

Quering Integrated Data Sources

Query Asnwering Techniques

- *query rewriting* - reformulates a query Q over the global schema wrt to the mapping and the local schema. The query result is then transformed to match the result format of Q.
- *query answering* - searches for bindings for the query variables using a reasoner over the mappings, the local and the global schema. Reasoners may support querying *complex class expressions*. Reasoner may be extended to support complex semantic query answering, e.g. $SPARQL - DL$, $SPARQL - DL^{NOT}$. Implicit knowledge may materialized in which case simple SPARQL engine is sufficient.

Schema Mapping using SPARQL CONSTRUCT

Listing 1 : graph1

```
@prefix ex1: <http://example1.org#> .

_:a a         ex1:Student;
                   ex1:fullName "Thomas, Edison";
                   ex1:takesCourse "Physics";
                   ex1:creditScore "40"
```

Listing 2 : graph2

```
@prefix ex2: <http://example2.org#> .

_:a a         ex1:Student;
                   ex2:firstName "Albert";
                   ex2:secondName "Einstein";
                   ex2:creditScore "10";
                   ex2:hasEnrolment [
                         ex2:semestr "summer";
                         ex2:hasCourse "Math".
                   ].
```

Schema Mapping using SPARQL CONSTRUCT

Listing 3 : query

```
PREFIX ex1: <http://example1.org#>
PREFIX ex2: <http://example2.org#>
CONSTRUCT
{ _:student ex1:fullName ?name;
                                        ex1:takesCourse ?course.}
WHERE
{ ?student  ex2:firstName ?fname;
                                        ex2:secondNane ?sname;
                                        ex2:hasEnrolment/ex2:hasCourse ?course.
      BIND (CONCAT(?fname, ", " ?sname) as ?name}
```

Listing 4 : result

```
@prefix ex1: <http://example1.org#> .

_:S ex1:fullName "Albert, Einstein";
                                        ex1:takesCourse "Math".
```

Conceptual Level Mapping

- Enterprise Conceptual Schema - the global schema
- Source Conceptual Schemas
- Domain Conceptual Schema mapping between the source and enterprise schema

Schema Mappings using DL

- used to express intentional statements
- concept inclusion axioms, i.e. $\sqsubseteq$
- example - $Programer \sqsubseteq Employee$

Schema Mappings using Rules

- used to express extensional statements
- example $hasProfession(?X, Programer) \rightarrow Employee$

Example Data Integration

Example 1

- **global schema**
  $Movie(Title, Year, Director, Critique)$, *director* and a sub-class *european*
- **source schema 1**
  $r1(Title1, Year1, Director1)$ - since 1960, european directors

- **source schema 2**

  $r2(Title2, Critique2)$ - since 1990

Listing 5 : query

```
PREFIX g: <http://example.org/global>
SELECT ?T ?R
WHERE
{ ?m a g:movie;
             g:Title ?T;
             g:Critique ?R;
             g:Year 1998
}
```

GAV Mapping

Mapping global as a view of source1 schema

- $Title1 \sqsubseteq Title$
- $Year1 \sqsubseteq Year$
- $Director1 \sqsubseteq Director$
- $r1 \sqsubseteq Movie$
- inverse $Director1$ some $r1 \sqsubseteq european$

Mapping global as a view of source2 schema

- $Title2 \sqsubseteq Title$
- $Critiqu2 \sqsubseteq Critiqu2$
- $r2 \sqsubseteq Movie$

Query over the integrated data

- The query from *Listing 5.* can be answered using query answering. This can be accomplished using for example a reasoner that supports conjunctive query answering, e.g. pellet and extended with $SPARQL-DL$ or $SPARQL-DL^{NOT}$.

LAV Mapping - representing the mapping as global constraints over the local sources

Mapping source1 schema as view of constraints over the global schema

- $european(?D), movie(?M), Director(?M, ?D), Title(?M, ?T), Year(?M, ?Y), greaterThan(?Y$ $\rightarrow r1(?M)$, $Director1(?M, ?D)$, $Title1(?M, ?T)$, $Year1(?M, ?Y)$

Mapping source2 schema as view of constraints over the global schema

- $movie(?M), Title(?M, ?T), Critiqu(?M, ?C), Year(?M, ?Y), greaterThan(?Y, 1989)$ -¿ $r2(?M)$, $Title2(?M, ?T)$, $Critique2(?M, ?R)$

Query rewriting according to LAV mapping

In order to answer the query in *Listing 5.* the query needs to be rewritten to target the individual sources. Generally the rewriting process requires a reasoning step over the mapping, the global schema and the local schema. In our case the the rewriting is simple.

Listing 6 : query

```
PREFIX s1: <http://example.org/source1>
PREFIX s2: <http://example.org/source2>
SELECT ?T ?R
WHERE
{
      GRAPH <http://example.org/source1> {
            ?m a s1:r1;
              s1:Title ?T;
              s1:Year1 1998.
      }

      GRAPH <http://example.org/source2> {
            ?m a s2:r2;
              s2:Title ?T;
              s2:Critique2 ?R.
      }
}
```

Thank you!

Contents

0.1 Introduction 1
0.1.1 Examples of Data Integration 1
0.1.2 Solution approaches 4
0.1.3 Technologies 7
0.2 Data Integration Theory 7

Resources

07dataintegration Patrick Ziegler and Klaus R. Dittrich

Data Integration — Problems, Approaches, and Perspectives

- `http://www.dis.uniroma1.it/~lenzerin/homepagine/talks/TutorialPODS02.pdf`
- `http://www.w3.org/People/Ivan/CorePresentations/DataIntegration/Slides.html`
- `http://en.wikipedia.org/wiki/Data_integration`