Semantic Data Exchange

Mariano Cilia
cilia@informatik.tu-darmstadt.de

Motivation

- Interchange data among applications
  - Cooperate, interact, ...
  - Fundamental for e-Commerce apps.
- 1/2/02
- 1234,00
- 0.5
- 23

Motivation

- Interchange data among applications
  - Cooperate, interact, ...
  - Fundamental for e-Commerce apps.
- 1/2/02
- 1234,00
- 0.5
- 23
- calculate( 21000, 3 )

Contextual information is required

Motivation

- Interchange data among applications
  - Cooperate, interact, ...
  - Fundamental for e-Commerce apps.
- 1/2/02
- 1234,00
- 0.5
- 23
- calculate( 21000, 3 )

Fundamental Semantic Heterogeneity

- Instance/entity identification problem
  - When two different information sources store information on an identical object, but do not share enough common information attributes to identify the object as the same
  - It cannot be solved algorithmically
Fundamental Semantic Heterogeneity (2)

- Two descriptions in different data sources have a similar meaning, but are quite the same
  - Neither data source (and schema) contains enough information to resolve the problem
    - E.g. prices in one DB include sales tax, while in a different DB do not
  - If data sources do not contain information to clarify the discrepancy, it cannot be resolved programmatically

Structural Semantic Heterogeneity

- When the same information is represented in two separate Apps in structurally different but formally equivalent ways. Two main reasons:
  - Normally, consequence of independent creation, design and evolution of autonomous Apps
  - Other reason, rich set of modeling constructors
- Spectrum of heterogeneity
  - Domain conflicts
  - Naming conflicts
  - Type conflicts
  - Structural conflicts

Domain conflicts

- Metadata specification differs and consequently conceptual schemas are different
  - The same entity is described differently in different domains
  - E.g. Paul is known as p123 in domain A and paul in domain B

Naming conflicts

- Objects may be represented in a different manner
  - The same attribute has different labels
  - E.g. Attribute name versus lastname

Type conflicts

- Systems represent low level atomic values differently
- Different types are used to describe related entities
  - E.g. temperature may be of type integer in one system and of type float in another

Structural conflicts

- Data may be managed by different DBMSs
- A different data organization or structure is used to represent the same concepts
  - E.g. An address type is represented as a structure or as a single attribute of type string
Overcoming Semantic Heterogeneity

- These heterogeneities can be overcome when
  - there is enough information in the metadata to clarify the meaning of each of the objects within the data source
- If it’s not the case
  - Resolving these conflicts become difficult
  - and sometimes impossible

Overcoming: Domain conflicts

- By using semantic dictionaries
  - Must contain mapping between domains
    - Dictionary entry with
      - `paul` identifies `p123` in domain A
      - `p123` identifies `paul` in domain B
  - When objects cannot be mapped 1:1 across domains, a more complicated mapping mechanism must be used
    - This can result in information loss
    - E.g. Representation of grades and marks
      - Out of ten and A, B, C, ... +, -

Overcoming: Naming conflicts

- Require mapping functions which simply change the labels of the attributes
  - Easy to overcome
  - A naming conflict can be mistaken for a structural conflict
  - E.g. An attribute price may or may not include taxes

Overcoming: Type conflicts

- Usually can be resolved by applying conversion functions
  - E.g. Programming languages provide functions for converting strings to integers

Overcoming: Structural confl.

- Involves decomposition or composition
  - E.g. The address type
    - represented in a structure composed into a single attribute
    - represented in a single string decomposed into a structure

Overcoming Semantic Heter.

- Ambiguity of data
  - Looking at data and metadata is sometimes not enough to fully understand its meaning
  - Completely removing ambiguity is unrealistic
- Semantic values
  - Facilitate integration of heterogeneous data
  - Help resolve many ambiguities in data
Data Integration - Issues

- Data from different sources/components is represented differently
- Different organizations/departments use different units and representation formats
- Many of the underlying assumptions about the meaning of a given data object are only implicit
- Context information is left implicit and consequently it is lost when crossing institutional boundaries

Why Semantic Metadata?

- The Internet as a global marketplace
- Business-to-Consumer:
  - interactive “point-and-click”
- Business-to-Business:
  - proprietary protocols
- Business-to-Business-to-Consumer:
  - need to extract and consolidate data for further electronic processing

Example

<table>
<thead>
<tr>
<th>Source</th>
<th>Flight</th>
<th>City</th>
<th>Departing</th>
<th>Arriving</th>
<th>Time</th>
<th>Seats</th>
<th>Meal</th>
</tr>
</thead>
<tbody>
<tr>
<td>LT</td>
<td>40</td>
<td>FRA</td>
<td>10:35</td>
<td>JFK</td>
<td>1200</td>
<td>0</td>
<td>S/ L</td>
</tr>
</tbody>
</table>

Price Per Adult (Economy Class): DEM 1826

Example

<table>
<thead>
<tr>
<th>Source</th>
<th>Flight</th>
<th>City</th>
<th>Departing</th>
<th>Arriving</th>
<th>Time</th>
<th>Seats</th>
<th>Meal</th>
</tr>
</thead>
<tbody>
<tr>
<td>LT</td>
<td>40</td>
<td>FRA</td>
<td>10:35</td>
<td>JFK</td>
<td>1200</td>
<td>0</td>
<td>S/ L</td>
</tr>
</tbody>
</table>

Price Per Adult (Economy Class): DEM 1826

Example

<table>
<thead>
<tr>
<th>Source</th>
<th>Flight</th>
<th>City</th>
<th>Departing</th>
<th>Arriving</th>
<th>Time</th>
<th>Seats</th>
<th>Meal</th>
</tr>
</thead>
<tbody>
<tr>
<td>LT</td>
<td>40</td>
<td>FRA</td>
<td>10:35</td>
<td>JFK</td>
<td>1200</td>
<td>0</td>
<td>S/ L</td>
</tr>
</tbody>
</table>

Price Per Adult (Economy Class): DEM 1826

The MIX Model

- MIX:
  - “Metadata based Integration model for data X-change”
- Combines two aspects:
  - representation of data plus additional semantic metadata
  - flexible, self-describing data model for the representation of semi-structured data
Ontology

An ontology is a formal specification of a shared conceptualization of a domain of interest

Ontologies in MIX

- Set of concepts and their relationships that model a given domain
- Ontology concept:
  - is an abstraction of a set of real world phenomena
  - has a representation type associated that determines the physical representation of data of a given concept
- Should be based on existing standards, but must be extensible

Simple Semantic Object

- A data item with additional metadata to support its interpretation:

  < Distance, 3850, {<Unit,"mile">, <Scale, 1>} >

  - 3850 is the recorded data value
  - Distance denotes the ontology concept
  - {<Unit,"mile">, <Scale, 1>} represents the interpretation context of the value 3850

Complex Semantic Object

- A heterogeneous collection of semantic objects grouped under a corresponding concept

  < FlightOffer, 
    < ClassOfService, "Economy", 
      < ClassOfServiceCode, "FullServiceClassName" > >, 
    < Price, 1826, 
      < Currency, "DEM" , < Scale, 1 > >, 
    < FlightSegment, 
      < FlightNumber, 400 >, 
      < AirlineIdentifier, "LH", 
        < AirlineIdentifierCode, "TwoLetterAirlineCode" > >, 
    < DepartureDate, "Jun 06 2000", 
      < DateFormat, "Mon DD YYYY" > >, 
    < DepartureTime, "10:35", 
      < TimeFormat, "HH:MM" > >, 
    < DepartureAirport, "FRA", 
      < AirportIdentifierCode, "ThreeLetterAirportCode" > > }, 
  
  ... 

Conversion Function

- A function that converts semantic objects between different contexts

  \[ \phi_{\text{Currency}} ( \{ \text{<Currency,"USD"} >, \text{< Price, 830, <Currency,"USD"} > \} ) = \phi_{\text{Price}} ( \text{<Currency,"USD"} >, \text{< Price, 1826, <Currency,"DEM"} > \} ) \]

  with "1 USD = 2.2 DEM"

- Provide a prerequisite for the integration of data from different sources

Semantic Equivalence

- Semantically equivalent objects represent the same information

  < Distance, 3850, {<Unit,"mile"}, <Scale, 1000} > >
  < Distance, 3850, {<Unit,"mile"}, <Scale, 1} > >

  - Determined through conversion to a common context and the comparison of their data values
  - Depends in general on the context and conversion function used
Semantic Equivalence
- Semantically equivalent objects represent the same information.
  - \(<\text{Distance}, 3850, \{\text{Unit, "mile"}, \text{Scale, 1}\}>\)
  - \(<\text{Distance}, 3850, \{\text{Unit, "mile"}, \text{Scale, 1}\}>\)

- Determined through conversion to a common context and the comparison of their data values.
- Depends in general on the context and conversion function used.

Semantic Identity
- Semantically identical objects represent information about the same real-world object.
  - Complex objects: if all identifying attributes are semantically identical (recursive).
  - Simple objects: if they are semantically equivalent.

Representing Ontology Concepts

<table>
<thead>
<tr>
<th>Simple Semantic Objects</th>
<th>Complex Semantic Objects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>Value</td>
</tr>
<tr>
<td>Description</td>
<td>Description</td>
</tr>
<tr>
<td>Semantic Context</td>
<td>Semantic Context</td>
</tr>
<tr>
<td>Functions</td>
<td>Functions</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Simple Semantic Objects</th>
<th>Complex Semantic Objects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>Value</td>
</tr>
<tr>
<td>Description</td>
<td>Description</td>
</tr>
<tr>
<td>Semantic Context</td>
<td>Semantic Context</td>
</tr>
<tr>
<td>Functions</td>
<td>Functions</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Simple Semantic Objects</th>
<th>Complex Semantic Objects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>Value</td>
</tr>
<tr>
<td>Description</td>
<td>Description</td>
</tr>
<tr>
<td>Semantic Context</td>
<td>Semantic Context</td>
</tr>
<tr>
<td>Functions</td>
<td>Functions</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Simple Semantic Objects</th>
<th>Complex Semantic Objects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>Value</td>
</tr>
<tr>
<td>Description</td>
<td>Description</td>
</tr>
<tr>
<td>Semantic Context</td>
<td>Semantic Context</td>
</tr>
<tr>
<td>Functions</td>
<td>Functions</td>
</tr>
</tbody>
</table>
Representing Ontology Concepts

Simple Semantic Objects
- Class
- Value
- Description
- Semantic Context
- Identifier

Complex Semantic Objects
- Class extends Class
- Value
- Description
- Semantic Context
- Identifier

Representation & Domain-specific Ontologies
- Representation Ontologies
  - domain-independent physical representation basis
  - enables exchange and reuse of concepts
  - contains concepts like Numeric-Value, or Character-String
- Domain-specific Ontologies
  - refer to a concrete subject domain
  - provide a consistent conceptualization of this domain
  - contains concepts like FlightOffer, or Distance

Abstract Model

Ontologic
Konversions-funktionen
MIX Modell

Representing Ontology Concepts

Advantages of using Java for the representation of ontology concepts:
- it avoids any impedance mismatch between programming and ontology specification language
- making concepts available as pre-compiled classes allows their shipping, and that of the corresponding data objects, between different platforms

Disadvantages of using Java for the representation of ontology concepts:
- Only Java programs can manipulate concepts
- Representation dependency
- It is not easy to
  - Identify instances
  - Share context among instances
  - Make relationships among concepts explicit
Representation

- Description of semantic objects using OWL (Web Ontology Language)
- Candidate Recommendation of the W3C
- Based on XML and therefore programming language and platform independent
- Use of URI as global identifiers (needed for context-sharing)

Current Implementation

OWL Representation - Sample

```xml
<owl:Thing rdf:ID="Price1">
  <mix:type>Simple</mix:type>
  <mix:concept>EngMath#Price</mix:concept>
  <rep:value>
    <xsd:string rdf:value="99.95"/>
  </rep:value>
  <rep:semanticContext rdf:resource="#CurrencyContext"/>
</owl:Thing>
```

OWL Representation – … Sample

```xml
<owl:Thing rdf:ID="EuroCurrency">
  <mix:type>Simple</mix:type>
  <mix:concept>EngMath#Currency</mix:concept>
  <rep:value>
    <xsd:string rdf:value="Euro"/>
  </rep:value>
</owl:Thing>
```

Current API

- Generic objects abstract from its representation
- Access and manipulation through a clean interface
- Store and load by using the repository
- Serialization and deserialization using factories
Data Integration based on MIX

- Integrated data
- (3) Unification of semantically identical objects
- (2) Conversion to a common context
- (1) Mapping to the MIX model

Heterogeneous data

Ontology-based Infrastructure

- In loosely-coupled (component-based) systems interfaces should consider the issues related with data interpretation
- Specification of arguments must allow the inclusion of metadata in order to correctly interpret them
  - Calculated <Price, 21000, <Currency, "EUR"> >
- Arguments can be converted to the target context before using them

Adapters / Wrappers

Adapters / Wrappers (cont.)

Sem. Data Exchange - Summary

- Common interpretation basis for data
- Implicit modeling assumptions have to be made explicit
- Business-to-Business-to-Consumer: need to extract, prepare and consolidate data for further electronic processing

Sem. Data Exchange – Summary (2)

- MIX:
  - allows to make explicit data semantics by using ontologies and metadata
  - supports integration, and automatic processing of Internet data
  - representation of data plus additional semantic metadata
  - flexible, self-describing data model for the representation of semi-structured data
  - conversion functions