Towards Semantic Interoperability in an Evolving Environment

Johann Eder Alps-Adria University Klagenfurt
... et respice finem

1957
Requirements for Interoperability Support

- Semantic expressiveness
- Operational scalability
- Maintainability
Distributed Information Systems

- Today we have the technique to deploy distributed information systems with affordable setup costs.
  - Web Services: interoperability on the syntactic level.
  - Emerging semantic interoperability (Semantic Web Services).
  - Emerging Approaches for formalizing and modeling distributed workflows.

- Typical problems in enterprise-scale applications:
  - Very high number of data instances
  - Demands for efficient computation → Can be a limiting factor for semantic applications.
Sub-Problem exchange of documents

- In an enterprise scenario one major use-case is the exchange of business documents. Due to evolution the content and meaning of such documents can change.

- Situation today: Transformation of documents with hand-written (tool assisted) scripts (i.e. XSLT).

- Task of the programmer:
  - Error-prone writing of mapping-scripts that map the syntactic and the semantic level.

- General idea (semi)-automatic transformation between document versions.
Evolution an different levels

**Schema change without ontology change**

$S \rightarrow S'$; $O = O'$ (Change on the syntactic level)
Example: Rename of XML-tag

**Ontology change without schema-change**

$S \rightarrow S$; $O \rightarrow O'$ (Change on the semantic level)
Example: Data gets new meaning: ICD-code version, German zip-code in 1993.

**Combination of schema and ontology-change**

$S \rightarrow S'$; $O \rightarrow O'$ (Changes on both levels)
Often schema-change induced by ontology change.
Introduction of a new attribute.
Example: Every invoice must now contain a tax number.
Possible Solution I

Semantic Level

- OWL, RDF

- Semantic Type version \( v \)
  - \( v_i \)
  - \( v_j \)

Multi-Ontology-Version Data Migration

Lifting to Semantic Type

Instance Level

- XML, XML Schema

- Data instance \( I \)
  - of Schema \( S \)

- Data instance \( I'' \)
  - of Schema \( S'' \)

Lowering to XML Type
Possible Solution 2: Proposed architecture
Scientific Basis for Evolvable Interoperability

- Required expressiveness
  - Ontologies

- Operational scalability
  - Fast document transformers

- Maintainability
  - Ontology Versioning, Knowledge Compilation
Need for (semi-)automatic adoption

1. There is the need for a system or language to record changes in a powerful way.
   - Goal: Automatic computation of the consequences of changes and decisions for adoption.
   - Can possibly be fulfilled by powerful semantic web techniques with reasoning capabilities.

2. Need for a highly effective and efficient transformation system which is scalable to support the high throughput expected in enterprise-scale applications.

   - 1 and 2 stay in conflict.
Conclusion:

- Evolution is a major concern in information systems.
- Distributed systems make the problem even more complex.
- There is a need for a system or language to record changes and their consequences in a powerful way. This requires reasoning support.
- Industrial-scale applications require a very high scalability.
- Scalability and Reasoning support stay in conflict.
- We propose “Knowledge Compilation” techniques to overcome this issue.
Possible Solution 1

- **Advantages:**
  - Full reasoning support for the transformation on the semantic level. Rules can apply on schema and data.

- **Enterprise-Scale Applications:**
  - Very high number of instance documents
  - Comparably low number of schemata.

- **Drawbacks:**
  - Computation on the semantic level is supposed to be expensive.
  - The transformation is done for every single instance document.