ToMAS: A System for Adapting Mappings while Schemas Evolve

Yannis Velegrakis[†] Renée J. Miller[†] Lucian Popa[‡] John Mylopoulos[†]

[†]University of Toronto
{velgias,miller,jm}@cs.toronto.edu

[‡]IBM Almaden Research Center
lucian@almaden.ibm.com

1. Problem Description

A broad variety of data is available on the Web in distinct heterogeneous sources, stored under different formats: database formats (relational), document formats (SGML/XML), browser formats (HTML), scientific data, etc. Merging and coalescing data from such sources continues to be an important problem for modern information systems and e-commerce applications. Such merging is often achieved through a set of mappings that specify how data instances of one or more source schemas correspond to instances of a target schema. Mappings are often specified in a declarative way, e.g., as queries or view definitions from the source schema(s) to the target schema.

In most existing data exchange and integration systems schemas are considered to be relatively static. However, in highly dynamic environments with no centralized authority, such as the Web, sources may evolve without prior notice. This evolution may involve not only their content but also their schemas and their query capabilities. When this happens, mappings that depend on these schemas may become invalid or inconsistent and will have to be adapted to conform to the new schema structures and semantics. For small schemas, browsing a short list of simple mappings to perform the required changes is a feasible option, but as the structure of the data and the schemas become more complex, the effort involved in that task is considerable, since it requires rewriting of large complex transformation queries and programs. To ensure that mappings remain correct and consistent, automatic tool support is required.

In this showcase, we demonstrate the *Toronto Mapping Adaptation System (ToMAS)* [5], a tool for automatically detecting and adapting mappings that have become invalid or inconsistent due to changes in either data semantics or schemas. It differs from other approaches in multiple ways:

1. ToMAS considers changes not only to the structure of the schemas but also to the schema semantics. ToMAS detects mappings that are affected by a schema constraint change even if the change does not make any of the mappings syntactically incorrect.
2. ToMAS generates rewritings that are guaranteed to conform to the modified schema structure and constraints.
3. ToMAS supports changes to either the source or the target schemas. This is analogous to adapting views after changes to the base table definitions or the view interface.
4. To generate the right rewritings, ToMAS exploits knowledge about user choices that is embodied in the existing mappings. Hence, the generated rewritings are consistent with the semantics of the original mapping.
5. The supported changes are not restricted to atomic type schema elements but can be applied to more complex schema structures such as relational tables or complex type XML-Schema elements.
6. The supported changes are not restricted to one schema element but may span elements located in different places in the schema. Such changes include copying and moving complex structures. These two changes cannot be simulated by element deletions and creations since the moved/copied elements need to carry with them constraints and mappings that were previously defined on them.

Due to its modular architecture and its stand-alone nature, ToMAS can easily be applied to numerous scenarios and can interoperate with many other tools. A list of the areas where a tool like ToMAS can be used includes (but is not limited to): Data Integration [3], Modeling Source Descriptions through Local-As-View mappings [3], Data Exchange [4], Physical Data Design [2], Model Management [1]. To the best of our knowledge, no other tool can correctly maintain the consistency of the mappings under schema changes at the level of complexity supported by ToMAS.

References

[1] P. Bernstein, A. Levy, and R. Pottinger. A Vision for Management of Complex Models. *SIGMOD Record*, 29(4):55–63, December 2000.

[2] P. Bohannon, J. Freire, J. R. Haritsa, M. Ramanath, P. Roy, and J. Siméon. LegoDB: Customizing Relational Storage for XML Documents, *System Demonstration*. In *VLDB*, 2002.

[3] M. Lenzerini. Data Integration: A Theoretical Perspective. In *PODS*, pages 233–246, 2002.

[4] L. Popa, Y. Velegrakis, R. J. Miller, M. A. Hernandez, and R. Fagin. Translating Web Data. In *VLDB*, pages 598–609, August 2002.

[5] Y. Velegrakis, R. J. Miller, and L. Popa. Mapping Adaptation Under Evolving Schemas. In *VLDB*, pages 584–595, September 2003.

Proceedings of the 20th International Conference on Data Engineering (ICDE'04)
1063-6382/04 $ 20.00 © 2004 IEEE