Integration with Ontologies

Conference Paper WM2003, April 2003, Luzern

(Note: This is the strongly shortened version adjusted for the conference handbook. We recommend downloading the long version (15 pages) at http://www.ontoprise.de/documents/Integration with Ontologies.pdf.)

author: Andreas Maier (maier@ontoprise.de)
co-authors: J. Aguado (jessica@miramon.net)
A. Bernaras (amaia@miramon.es)
I. Laresgoiti (lares@labein.es)
C. Pedinaci (carlos@miramon.net)
N. Peña (npena@labein.es)
T. Smithers (tim@miramon.net)

Abstract: One of today's hottest IT topics is integration, as bringing together information from different sources and structures is not completely solved. The approach outlined here wants to illustrate how *ontologies* [Gr93] could help to support the integration process.
The main benefits for an ontology-based approach are
- the ability to cover all occurring data structures, for ontologies can be seen as nowadays most advanced knowledge representation model
- the combination of deduction and relational database systems, which extends the mapping and business logic capabilities
- a higher degree of abstraction, as the model is separated from the data storage
- its extendibility and reusability

1. Motivation for an Ontology-based Approach

The goal of integration is to consolidate distributed information intelligently, free of redundancy, processed and operated by the right business logic to deliver the appropriate and condensed answer and offer the end user a simple access to it, without him needing knowledge about the underlying data structures. We believe that with ontologies there's now a model at hand to fit for this goal.

1.1. Defining the Requirements

In our integration process we have
- to cover all existing data **structures** (*requirement 1*), which can be simple table structures up to complex hierarchical structured data with deep inheritance,
- to **map and merge** these schemas among each other (*requirement 2*),
- to define the **logic** for the whole new application (*requirement 3*) (hereby we will be supported by deductive inference mechanisms) and
- to provide a performant data **storage** for the information (*requirement 4*).

In our view ontologies are the best representation model to meet these requirements. In the next chapter we want to prove this statement.

2. Foundations: Enabling the Ontology-based Integration

2.1. Requirement 1: Cover all Data Structures

In [Ma01] we compared several knowledge representation models and discussed the advantages and weaknesses of them. As a conclusion we found that ontologies are the most advanced model of all of them, summing up most of the qualities of the others:

- Like **Taxonomies** [Pe89], ontologies are able to cover *hierarchies*.
- Like **Thesauri** [Me95], **Semantic Nets** [Ho86] and **Topic Maps** [PH02], ontologies contain *relations*. With them, complex contexts can be modelled and visualized in nets. *Linguistic contexts* (i.e. multilingualism or synonym relations), terminologies and classifications can be described, through which the semantic of the integration solution is increased [An01]. Taxonomies, Thesauri, Semantic Nets and the ER-model are comparatively old models, as the dates of the references (chapter 0.) show. Topic Maps are the newest of them and may merge with ontologies.
- Like the **EntityRelationship-Model** (ER) [BCB91] and unlike the others mentioned above, ontologies have a *data model* distinguishing schema information from facts.
- As an **object based model**, ontologies support *inheritance* and *multiple inheritance* of attributes.

2.2. Requirement 2: Mapping and Merging

Before starting the mapping procedure the structures respectively the schemas have to be imported into the ontology. For thus, an ontology modelling tool is required providing various schema import filters for different formats (i.e. for all relevant commercial databases). Such a tool must also support the fundamental mapping types

- concept-to-concept mapping
- attribute-to-attribute mapping
- attribute-to-concept mapping as well as
- conditions and constraints[1] on the mapping rules (which is not explained further here)

2.3. Requirement 3: Deductive Logic

An often asked question is: “Why using logic? Didn’t databases solve all problems decades ago?”

On the one hand applications with lots of logical dependencies (i.e. configuration or variant management systems, solutions representing extensive knowledge domains, expert systems) can be realized much better with rule-based systems.

On the other hand deductive logic reduces complexity. It’s a difference, if you ask

- “Who is the contact person of client ‘Smith’?” or
- “Who is the employee that handles the orders of the product, that client ‘Smith’ has ordered?”.

As this is just a small example, in really complex contexts with many relations between the concepts of the ontology the the effort and complexity to realize in SQL quickly gets too high. As a third reason, the user doesn’t need to know the underlying data structures.

[1] i.e. unit conversions

For example he only knows, that he can ask for “contact persons of clients”, and not the whole conceptual structure that lies behind this question.

2.4. Requirement 4: Provide a Data Storage

For maintenance reasons, the data itself should be kept only once, preferably in the origin application. If this application isn’t able to query, we propose migrating it to a database. Although the ER-Model has its weaknesses (chapter 2.1), we suggest using relational databases as storage because of its widely spread and mature solutions. In comparison to other repositories there’s no alternative concerning performance and compatibility. Therefore we need an SQL-Export for creating the database schema out of the ontology.

3. Introducing a Toolbox for the Ontology-based Integration

As we found in (chapter 2.) and (chapter 3.), the following components are needed within an ontology modelling environment [AS2002], meeting the requirements for an integration solution:

- a core modelling component for concepts, attributes, relations, instances, multilingual representations and domain entries
- a schema import and export supporting various formats, particularly SQL
- a mapping tool
- a rule editor
- a rule debugger

The first three points have been realized in recent modelling tools more or less. For the last two points we will introduce a rough idea of visualization, not covering all functionalities coming up.

3.1. Visual Rule Editor

In our example, *rule 1* is a composition relation rule, connecting the three concepts *product*, *order* and *employee*. In our proposal (figure 1: a visual rule **editor**), a user would select them by drag&drop from a left window, where all concepts are listed in a “is-a”-hierarchy, and move them to the center window.

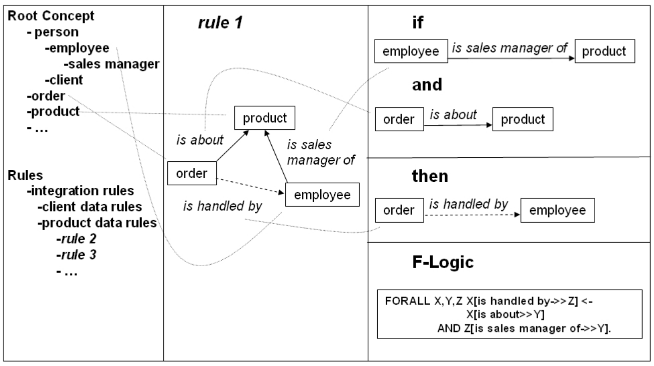

figure 1: a visual rule editor

There the modelled relations (*is about*, *is sales manager of*, *is handled by*) will appear. By moving them by drag&drop into the fields on the right side (*if*, *then*), you would create the rule shown in F-Logic code below. There you could change the rule also by hand.
In figure 1 the ability to define attribute conditions (i.e. employee.name="Miller") or operators (+, -, *, /, NOT, EXISTS, …; i.e. price=quantity*[price per unit]*discount) is missing and has to be added to the draft yet.

3.2. Visual Rule Debugger

The visual rule debugger is an important tool for the IT professional. It's supposed to support him during the rule modelling phase, showing him the outcome of the rules. Thereby it visualizes the inference process for one selected new fact (figure 2: a visual rule debugger). A graph would appear showing the course of conclusion.

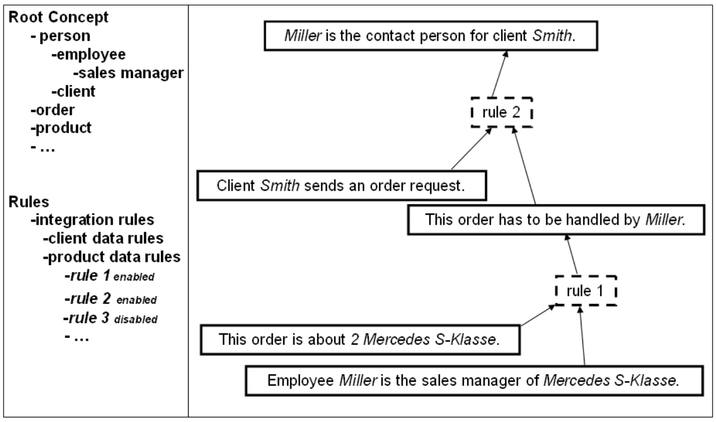

figure 2: a visual rule debugger

4. Closing Remarks

The closing remarks can be found in the long version of this paper at http://www.ontoprise.de/documents/Integration with Ontologies.pdf.

5. References

The references can be found in the long version of this paper at http://www.ontoprise.de/documents/Integration with Ontologies.pdf.