EU-IST Project IST-2003-506826 SEKT

SEKT: Semantically Enabled Knowledge Technologies

D4.5.3 Prototype of the ontology mediation software V1

Moritz Weiten,

Dirk Wenke, Mika Meier-Collin (all ontoprise GmbH)

Abstract

This document describes the functionality of the mediation software developed in SEKT with emphasis on step-by-step guides. Mediation as an important aspect of ontology management is briefly introduced followed by an overview of the mediation components in SEKT. The main part of the document describes the usage of the mediation software.

Keyword list: ontologies, semantic technologies, ontology management, ontology mediation, ontology mapping, semantic integration

WP4 Ontology Mediation

Prototype	PU
Contractual date of delivery 30/06/05	Actual date of delivery 12/07/05

SEKT Consortium

This document is part of a research project partially funded by the IST Programme of the Commission of the European Communities as project number IST-2003-506826.

British Telecommunications plc.
Orion 5/12, Adastral Park
Ipswich IP5 3RE
UK
Tel: +44 1473 609583, Fax: +44 1473 609832
Contact person: John Davies
E-mail: john.nj.davies@bt.com

Empolis GmbH
Europaallee 10
67657 Kaiserslautern
Germany
Tel: +49 631 303 5540
Fax: +49 631 303 5507
Contact person: Ralph Traphöner
E-mail: ralph.traphoener@empolis.com

Jozef Stefan Institute
Jamova 39
1000 Ljubljana
Slovenia
Tel: +386 1 4773 778, Fax: +386 1 4251 038
Contact person: Marko Grobelnik
E-mail: marko.grobelnik@ijs.si

University of Karlsruhe, Institute AIFB
Englerstr. 28
D-76128 Karlsruhe
Germany
Tel: +49 721 608 6592
Fax: +49 721 608 6580
Contact person: York Sure
E-mail: sure@aifb.uni-karlsruhe.de

University of Sheffield
Department of Computer Science
Regent Court, 211 Portobello St.
Sheffield S1 4DP
UK
Tel: +44 114 222 1891
Fax: +44 114 222 1810
Contact person: Hamish Cunningham
E-mail: hamish@dcs.shef.ac.uk

University of Innsbruck
Institute of Computer Science
Techikerstraße 13
6020 Innsbruck
Austria
Tel: +43 512 507 6475
Fax: +43 512 507 9872
Contact person: Jos de Bruijn
E-mail: jos.de-bruijn@deri.ie

Intelligent Software Components S.A.
Pedro de Valdivia, 10
28006
Madrid
Spain
Tel: +34 913 349 797
Fax: +49 34 913 349 799
Contact person: Richard Benjamins
E-mail: rbenjamins@isoco.com

Kea-pro GmbH
Tal
6464 Springen
Switzerland
Tel: +41 41 879 00
Fax: 41 41 879 00 13
Contact person: Tom Bösser
E-mail: tb@keapro.net

Ontoprise GmbH
Amalienbadstr. 36
76227 Karlsruhe
Germany
Tel: +49 721 50980912
Fax: +49 721 50980911
Contact person: Hans-Peter Schnurr
E-mail: schnurr@ontoprise.de

Sirma AI EAD, Ontotext Lab
135 Tsarigradsko Shose
Sofia 1784
Bulgaria
Tel: +359 2 9768 303, Fax: +359 2 9768 311
Contact person: Atanas Kiryakov
E-mail: naso@sirma.bg

Vrije Universiteit Amsterdam (VUA)
Department of Computer Sciences
De Boelelaan 1081a
1081 HV Amsterdam
The Netherlands
Tel: +31 20 444 7731, Fax: +31 84 221 4294
Contact person: Frank van Harmelen
E-mail: frank.van.harmelen@cs.vu.nl

Universitat Autonoma de Barcelona
Edifici B, Campus de la UAB
08193 Bellaterra (Cerdanyola del Vall` es)
Barcelona
Spain
Tel: +34 93 581 22 35, Fax: +34 93 581 29 88
Contact person: Pompeu Casanovas Romeu
E-mail: pompeu.casanovas@uab.es

Executive Summary

Mediation is an important aspect of ontology management. The interoperability of models or schemas is crucial for the integration of data sources, the reuse and evolution of models. In industrial applications semantic technologies are hardly applied in form of isolated solutions. Semantic integration plays an important role, either as the main goal (e.g. single-view integration for efficient retrieval and access to information) or as an additional requirement (e.g. for an advice system used on existing data sources).

A frequent form of ontology mediation is ontology mapping, which the development in the SEKT project has been focused on. The ontologies involved remain unchanged, while entities are mapped onto each other. This can be done either unidirectional or bidirectional. A typical application would be the data integration problem mentioned above.

As a basic prerequisite languages used for the representation and grounding need to support the definition of mappings in a flexible and efficient way. Especially for the grounding a well-defined semantics for mappings must be given. However, this is not enough for typical applications of ontology mapping. A major issue is an appropriate tool support and a solid conceptual base. The mediation of large and/or complex ontologies requires a significant effort. An expert of knowledge representation cannot be assumed as the general case for mediation approaches in enterprises. Therefore intuitive usage and efficient user support in form of semi-automated processes are important aspects influencing the productivity of semantic technologies.

The set of tools that has been developed for mapping addresses this issue. As part of the ontology-management platform OntoStudio the mapping plugin OntoMap allows the comfortable creation and management of mappings in a graphical environment (see [Wenk2004]). The actual representation (grounding) of mappings is encapsulated, while users are supported by drag-and-drop functionality, form-based creation of mapping conditions and consistency checks on attribute mappings.

Users can export mappings from OntoMap to a mapping-store. The latter supports the store and retrieval of mappings in a neutral language, which has been developed based on the idea of mapping-patterns (see [deBrui05]). While not all mappings that can be defined in the mapping-language are OntoMap-compliant, a great portion of typical use-cases is covered by the mappings that can be exchanged.

The Framework for Ontology Mapping and Alignment (FOAM) predicts mappings for given ontologies based on similarity values (see [Ehri2004b]). It increases the productivity of manual mapping approaches, as it can be used for mapping-suggestions. OntoMap uses this capability to create mappings automatically which users can then accept or withdraw.

This deliverable has its focus on mapping, while ontology merging as well as the grounding and execution of mappings will be addressed in the next version of the mediation software. The work for grounding and execution of mappings has already started, based on an interface between the KAON2-ontology server and OntoMap.

Contents

1 Introduction

Mediation is a major aspect of information and knowledge management. As semantic technologies are starting to be established as a base for enterprise solutions, the relevance of interoperability between different ontologies (or generally “schemas”) increases. In modern IT-infrastructures, semantic solutions do not rely on isolated knowledge based systems but on flexible architectures, supporting the integration of existing data sources. They provide a significant added value (such as efficient retrieval and access, evaluation of constraints or knowledge-based support of business processes) rather than replacing existing components. Therefore the “bottleneck” of those solutions is often not the acquisition of instance data but the appropriate semantic description of metadata to access existing data sources based on ontologies.

Data integration is not the only “driving force” for mediation. Ontologies follow the idea of reuse of knowledge models. In an ideal case reusing existing ontologies would imply a modular combination of models for different domains without overlaps and redundancies. As this is hardly the case in practice, efficient methodologies for ontology mediation are required. Other use-cases for mediation include for example the evolution of ontologies (see [Ehri2005]) or the (semi-) automatic generation of ontologies, such as in the SEKT case studies (see section 2).

The support for mappings starts with the capabilities of ontology representation languages and evaluation (reasoning) engines. Mappings need to be expressed in a suitable form with a defined semantics and efficiently executed. However, the effort for the creation of mappings is a limiting factor for mediation approaches. Efficient tool-support is therefore crucial. The effort for the creation of mappings can be basically a matter of quantity, when for example large legacy systems with possibly simple structures but a high number of attributes need to be integrated. Complex data structures on the other hand might require some expertise for the creation of efficient mappings. Both aspects are addressed by the components that have been developed.

The following sections provide some background information on ontology mediation and a step-by-step guide of how the mediation software is applied.

2 Background: Mediation of Ontologies

Ontology mediation includes different methodologies, as for instance the **merging** of different ontologies into a new ontology, the **mapping** of ontologies (or parts of them) and the **alignment** of ontologies. The latter occurs in particular in connection with different versions of the same ontology. When an ontology-mapping is created, the participating ontologies remain unchanged (apart from the additional strcutures representing the mapping definition). Mappings can be defined bi-directional or unidirectionally and on the attribute level, on the relation level or on the concept level. In contrast to ontology-alignment does an ontology-mapping not necessarily involve all parts of the respective ontologies. The example given in section 4.1 is based on a case where the intersection of ontologies is to be defined. The rest of this document has its focus on ontology-mapping (which reflects the current status of the mediation software). Use-cases for mappings are given in the following sections.

A detailed description of ontology mediation, associated methodologies and tools is given in [deBrui04b] and [Kalf2005].

2.1 Reuse of Ontologies

One of the substantial advantages of ontology technologies is the concept of reuse. In contrast to other technologies related to information management and storage, ontologies incorporate the idea of knowledge reuse, due to the fact that they rely on explicit and formal semantics as well as on expressive representation formats (languages). Rather than describing domains (or super-ordinate concepts) from the scratch, reusing existing ontologies increases the efficiency of the ontology development process and supports the exchange of information and knowledge. However, the combination of existing ontologies often implies overlaps and redundant specifications based on different structures. In this case, the relation between the ontologies needs to be defined explicitly.

2.2 Data Integration

The semantic integration of data sources makes it necessary to map existing data-structures to an appropriate ontology. This can be achieved in at least two basic steps:

- Map the given data source onto the semantic representation form of the target system via a defined bridging of the target representation language and the access layer of the underlying data source;
- Map the resulting ontology onto the target ontology/ontologies based on the representation language of the target layer.

[Wenk2004] describes the schema-import of OntoStudio, which provides the base for a semantic integration of relational databases. If for example a single-view integration of different relational databases has to be established, the schema-import would have to be repeated for all relevant database schemas and then mappings to the

ontology/ontologies would have to be created. The result would be an online integration of the selected databases in form an access layer for instance data of the mapped ontology.

A slightly different form of integration is the motivation for the use-case in section 4.1. In that case there is no “top layer” ontology for the integrated description of the underlying data sources but an existing model (possibly bound to a particular application, such as a data analysis tool for example) through which external sources need to be accessed. Rather than directly mapping the models an intermediate ontology is created, which can be seen as the intersection.

2.3 Application in the Case-Studies

The case studies involve an infrastructure of heterogeneous, possibly overlapping ontologies. Support for mediation therefore is a basic requirement.

In the legal case study, the system to support inexperienced judges includes different domain-specific ontologies that need to be aligned or merged. Those ontologies are partially built automatically from a number of databases containing information about sentences and merged into a one ontology providing a single view. The “Ontology of Legal Professional Knowledge” on the other hand represents the knowledge related to a repository of frequently asked questions. This ontology is to be aligned with the ontology representing the information of the databases.

The digital library use case ontology learning and metadata extraction is used to categorize documents. Mediation techniques are needed to mediate between the different ontologies that are provided for the documents. At later stages of the project, “personal ontologies”[1] of library users might be considered. Those will have to be aligned with the library ontology.

[1] This is -strictly spoken- a contradiction, since the definition of ontologies implies that there is a common view and a commitment within a community. However, a “personal ontology” in this context could be an ontology of a certain community a particular user is a member of.

3 Mediation Components in SEKT

This section describes the mediation components and the architecture for the interaction of those components. Section 4 describes the usage of the components based on OntoMap as end application. It does therefore not cover all the capabilities the components provide on their own.

Usability versus Flexibility/Complexity

Support for mediation can be provided on different levels. A graphical tool such as OntoMap provides a comfortable, efficient way to create and manage ontology mappings. However, there are limitations for a graphical interface with respect to the flexibility. Complex mappings require a manual coding in the underlying representation language (which would be F-Logic rules in the case of OntoStudio). The suitability of a certain approach depends on the user profile and of course on the application, two aspects that need to match as well (assuming an expert user for complex mediation problems for example).

Mapping patterns provide a promising approach for less experienced users with a basic understanding of ontology mappings to solve certain categories of mediation problem (see [deBrui05] for the description of patterns and the underlying problems). However, only not all patterns are currently supported by OntoMap (see [Wenk2004] for further reference). OntoMap follows the idea of intuitive use, which limits the types of mappings. A wider range of mapping patterns could be supported by a different category such as a source editor for the representation language providing the grounding for mappings (e.g. in form of code-templates with completion-suggestions for entities).

Section 4 assumes a non-expert user not capable of sophisticated mediation solutions but familiar with the semantics of simple mappings between entities. This setting still covers a wide range of scenarios.

3.1 Software Components

3.1.1 Ontology and Mediation Management: OntoMap

OntoMap is a plugin for the ontology-management platform OntoStudio that supports the creation and management of ontology mappings via a graphical interface. Mappings can be specified based on graphical representation, using a schema-view of the respective ontologies. Users just have to understand the semantics of the graphical representation (e.g. an arrow connecting two concepts), they do not have worry about the logical representation mappings. The user of OntoMap is supported by drag-and-drop functionality and simple consistency checks on property-mappings (automatic

suggestion of necessary class-mappings). For concept mappings[2] constraints can be specified on the available attributes, based on a form.

OntoStudio has its own grounding of mappings, based on F-Logic rules (see [Kife1997]). OntoMap supports simple types of mappings. If more complex mappings are needed (possibly using complex logical expressions or built-ins), the have to be encoded manually. However, OntoMap still covers a great number of use-cases. The rules that are currently supported include:

- concept to concept mappings;
- attribute to attribute mappings;
- relation to relation mapings;
- attribute to concept mappings.

In OntoStudio queries on stored ontologies can be formed and executed instantly, which gives users the possibility to test the consequence of mappings they have created (or imported).

3.1.2 Mapping-Store and Mapping-Language

The language for ontology mappings has been developed as language-neutral representation of mappings that corresponds to *mapping-patterns* (see [deBrui05]). Mapping patterns represent a schema for frequent mappings. Persistency of mappings defined in the Mapping Language is supported by a Language API and a Mapping-Store (both Java-libraries, see [owmg05]). The latter supports the storage and the retrieval of mappings.

The mapping language does not prescribe a particular semantics, but it has relations to existing languages (WSML-Flight and OWL DL) that allow the grounding of actual mappings. The focus of the mapping-language is on the conceptual correspondences of entities, not on a cross-language semantics for mappings (even though a reference semantics is available, see [deBrui05]). While expressions in the mapping-language can be used in terms of a template for actual mappings, it is also possible to express actual mappings in the mapping language. The latter process is the base for the interoperability of OntoMap and the mapping-store in its current version.

The framework of mapping-language and mapping-store strongly supports the concept of a library, allowing to benefit from previous work. Even though the approach is very different from the one described in the next section, both have implications for the efficiency of ontology mediation.

[2] By the term “concept mappings” refer to mappings from concept to concept.

3.1.3 Framework for Ontology Alignment and Mapping

The Framework for Ontology Alignment and Mapping (FOAM) has been developed by Ehrig et al. at the AIFB. It consists of a Java-tool which allows the automatic suggestion of possible mappings based on similarity rules (see [Ehri2005] and [Ehri2004b]), which are encoded manually or by means of machine learning. Examples for similarity rules would be:

- *If labels of concepts are similar, those concepts are similar as well.*
- *If the super-concepts of concepts are similar, those concepts are similar as well.*
- *If the sub-concepts of concepts are similar, those concepts are similar as well.*
- *[…]*

A similarity value ranges between zero and one. It indicates the similarity between entities whereas the value one means that the compared objects are identical. FOAM combines similarity rules and uses weights on them to determine the likelihood of a particular mapping. Relevant and irrelevant candidates are separated based on a threshold, which is one of the "tuning parameters" that are available.

A main feature of FOAM is that it provides a confidence measure for the mappings that have been finally identified. This enables third-party tools to decide which candidates should be suggested or stored for further manual processing.

The manual creation of mappings is a time-consuming task and is a potential "bottleneck" of integration approaches for example[3]. Therefore FOAM provides an important complement to the set of tools for ontology mediation.

[3] A bottleneck that is inherent to "non-semantic" approaches in a similar form.

3.2 Architecture

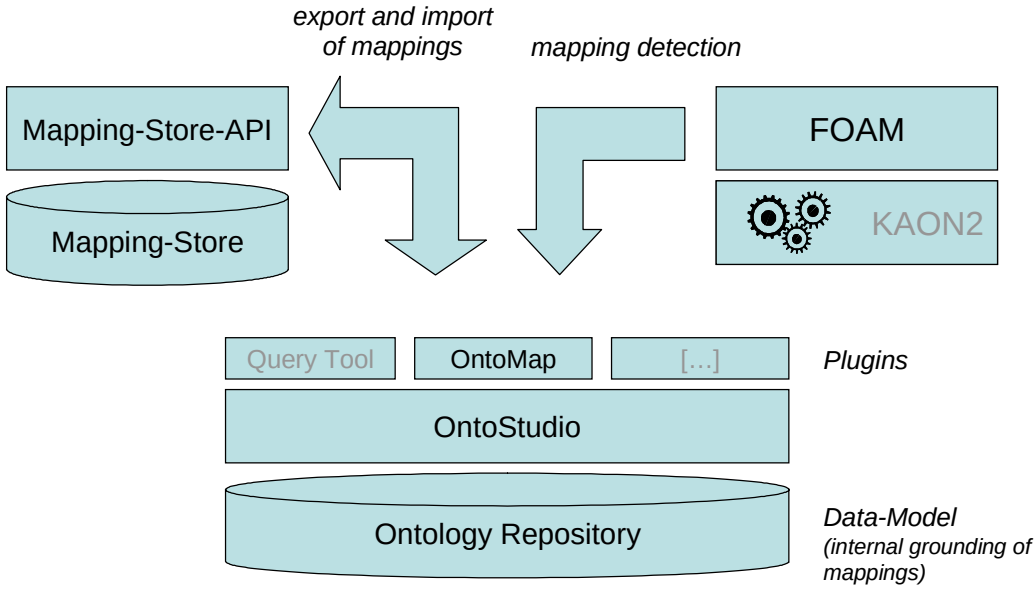

Fig. 1: **Architecture schema**

The current version of the mediation software relies on a direct coupling of the components described in section 3.1.

4 Usage of Mediation Prototype

This section describes the usage of the ontology medation software in its current version. Note that it does not cover all possibilities of the available components. The focus is on OntoMap as a graphical tool for the creation and management of ontology mappings. The full capability of the mapping-store can be exploited when it is used as a library (see [owmg05] for the details). The same holds for the foam-framework (see [FOAM2005]).

4.1 Manual Creation of Mappings

The example for the mapping scenario the following description is based on is a slightly changed version of the scenarios given in [Mitr00] and [Meis2002]. There are two ontologies for which interoperability is required: A “factory”-ontology and a “carrier”-ontology. An additional ontology, the “vehicles”-ontology, defines the intersection between those ontologies. Note that we do not describe how the “vehicles”-ontology is extracted out of the other ontologies. For further reference refer to [Meis2002].

Mappings in OntoStudio are created in a graphical manner, using “drag and drop”. The mappings are stored internally in form of rules (as described in [Wenk2004]) based on a representation in F-Logic (see [Kife1997]). However, this mechanism is encapsulated by the graphical user-interface. Therefore no knowledge about F-Logic (or other represenation languages) is required to create and manage mappings.

The following steps are required to create ontology-mappings:

- opening the respective ontologies in OntoStudio;
- opening the mapping-view;
- specify source- and target ontologies;
- map concepts, attributes and relations of the ontologies involved using the drag- and drop capabilities of the mapping view;
- specify constraints for conditional mappings (if necessary).

4.1.1 Opening the mapping view

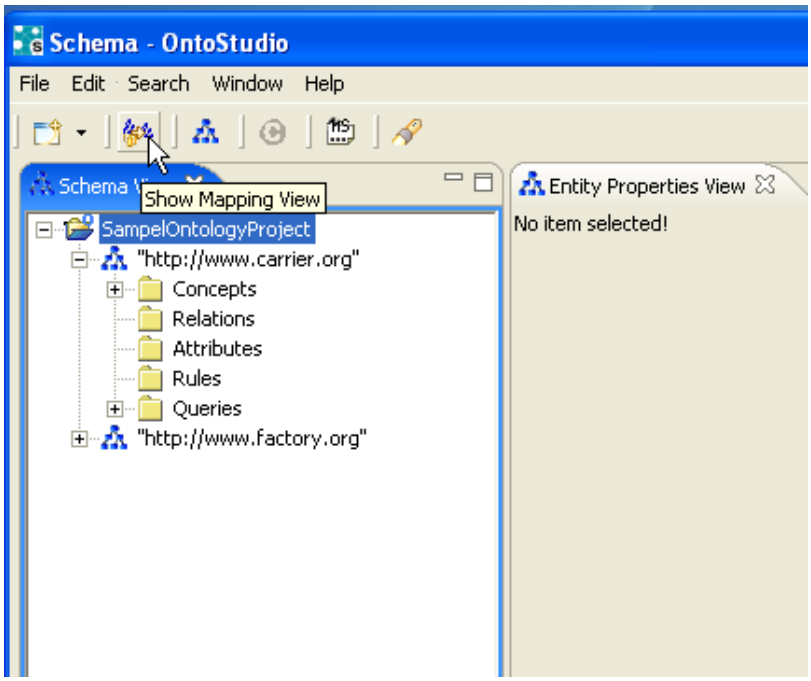

Fig. 2: **Opening the mapping view.**

To start the creation of mappings, the mapping view needs to be opened. To achieve this, the mapping view button needs to be clicked (as shown in figure 2). Figure 3 shows the mapping view. A double click on the blue mapping view tab (indicated by the mouse-pointer) enlarges the view to the maximum. The view can be reset to the original size by another double-click on the tab.

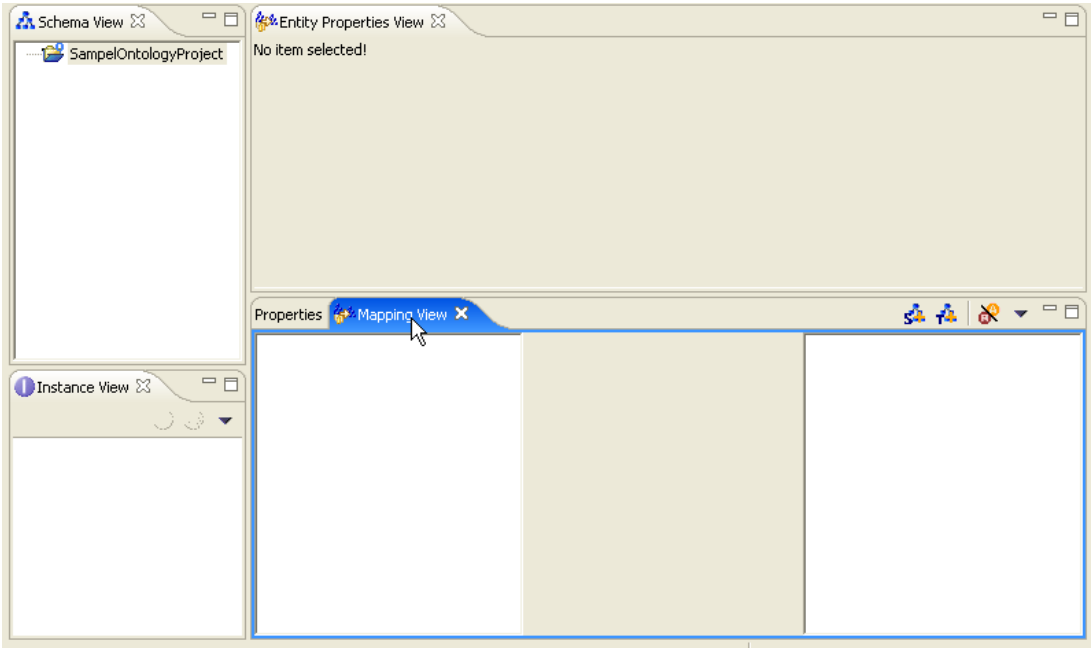

Fig. 3: **Mapping view.**

4.1.2 Specify source and target ontologies

In the mapping view, target and source ontologies for mappings can be specified in two ways:

- via the schmema-view using drag and drop;
- using the selection list of the mapping view.

For the latter, the selection list has to be activated by clicking the “select source ontologies” button in the upper right of the mapping view as shown in figure 4 (“select target ontologies” respectively). In a next step, the source ontologies can be specified using the check-boxes in the selection list (see figure 5). In our example, we select the “carrier-ontology” and the “factory-ontology” as source ontologies.

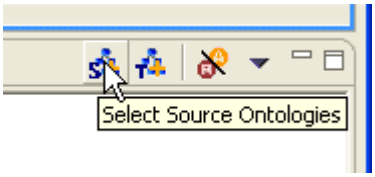

Fig. 4: **Specification of source ontologies (1): open selection list.**

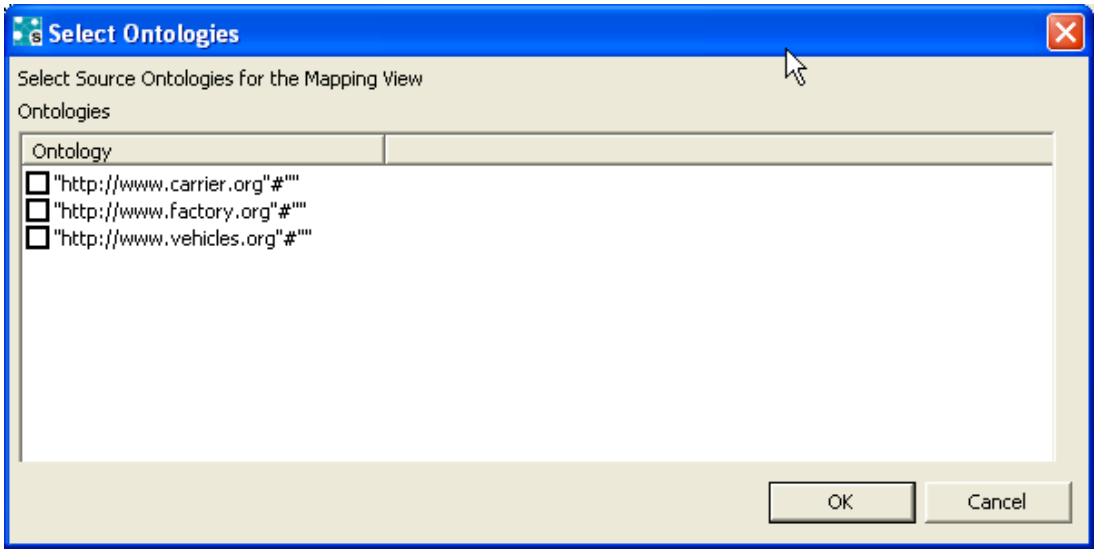

Fig. 5: **Specification of source ontologies (2): selection list.**

The “vehicles-ontology” of our example is specified as target-ontology using drag-and-drop. To achieve this, the target-ontology has to be selected in the schema-view and dragged over to the target field of the mapping view (see figure 6).

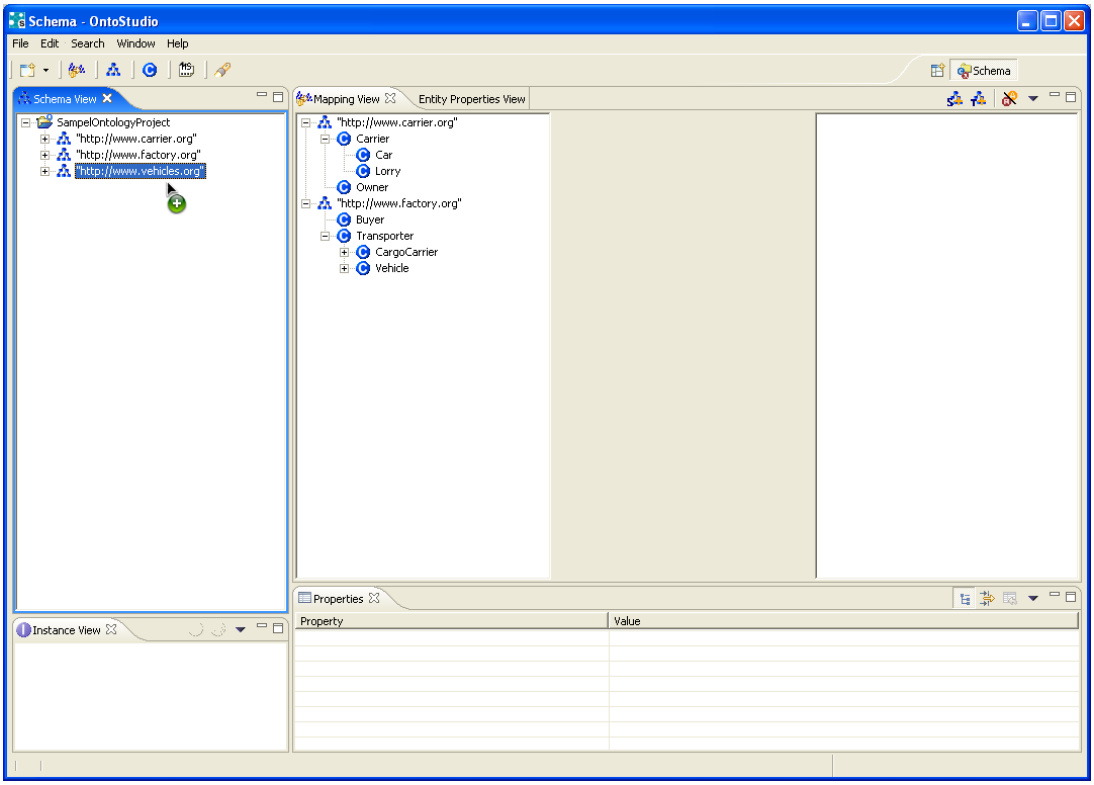

Fig. 6: **Specification of target-ontology using drag and drop.**

4.1.3 Create mappings

In a first step, we create the concept-to-concept mappings by selecting the source-concept and dragging it onto the target-concept. For bidirectional mappings the process has to be repeated in the other direction. The concept “carrier” of the carrier-ontology and the concept “transporter” of the factory-ontology are both mapped onto the concept “carrier” of the vehicle-ontology (bidirectional). The result is shown in figure 7.

We create all remaining concept-mappings but leave out the necessary mappings from “lorry” (carrier-ontology) to “truck” (vehicles-ontology). Once all concept-mappings have been set up, we start to create the property-mappings. In the mapping-view, we activate the visualization of properties by clicking the “show/hide properties”-Button in the upper right of the mapping view (see figure 8). The property-mappings are then created in the same way as described for concept-mappings.

OntoMap provides consistency checks for attribute-mappings. We try to map “max_weight”-attribute of lorry onto “max_weight” of truck without having mapped the concepts “lorry” and “truck”. OntoMap shows a warning message (see figure 10). The missing concept-mappings can be created automatically by clicking the Ok-Button. If the Cancel-Button is chosen, the attribute mapping will not be stored.

Figure 9 shows the result when all necessary mappings of the “vehicles-example” have been constructed.

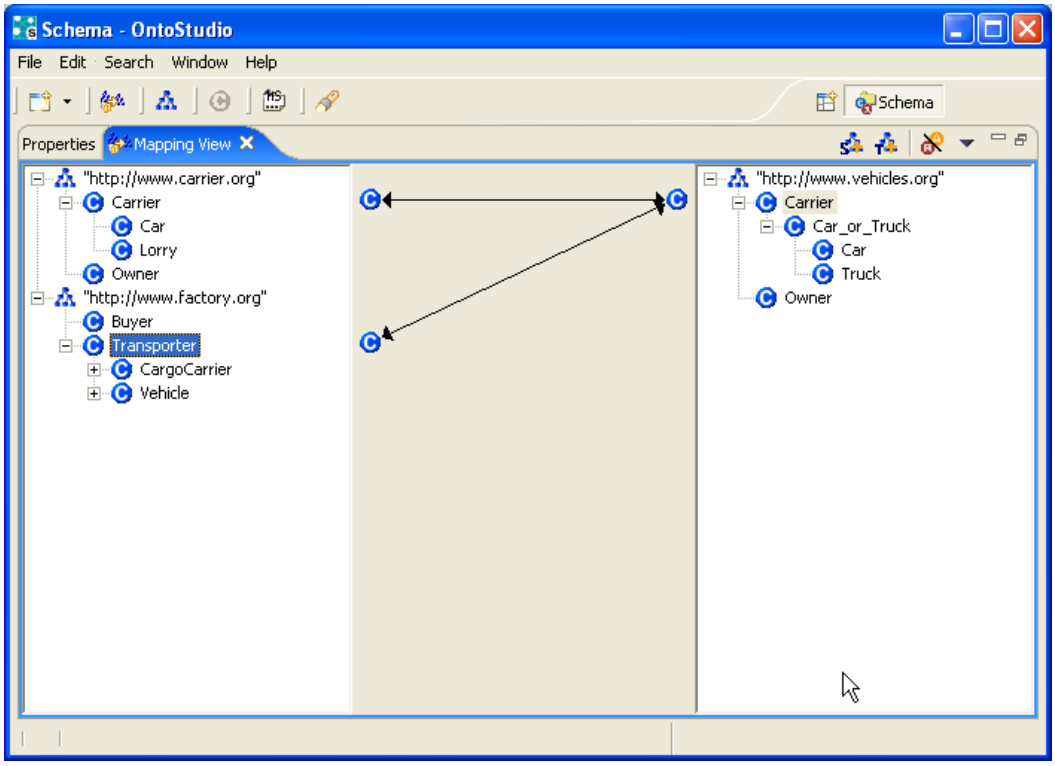

Fig. 7: **Mapped concepts representing carrier/transporter.**

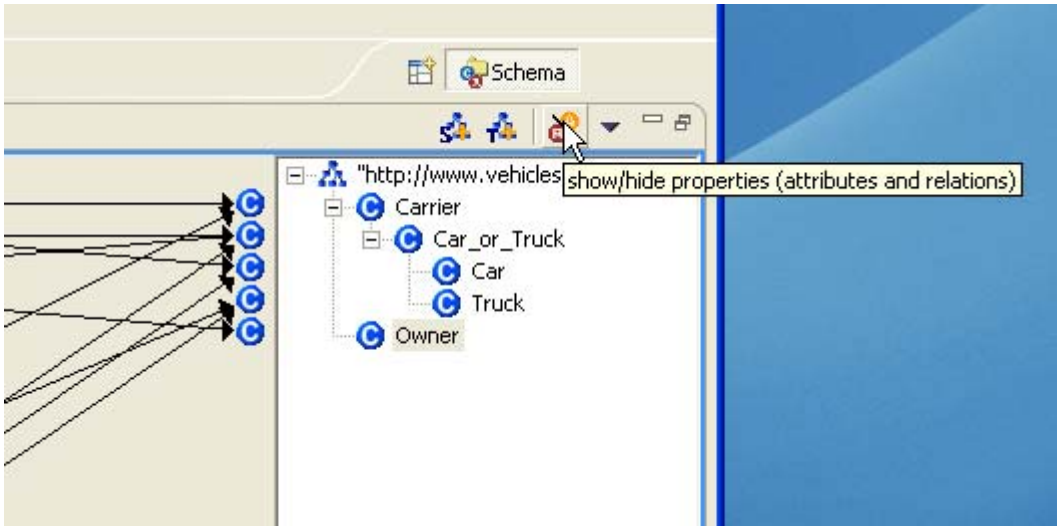

Fig. 8: **Show properties of source and target ontologies.**

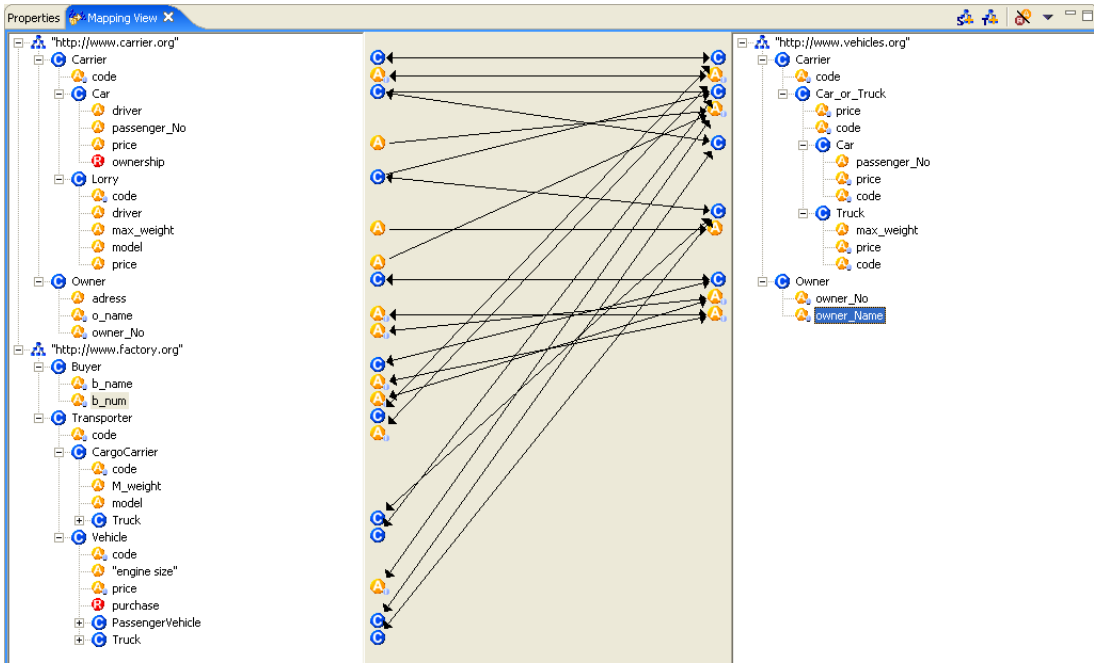

Fig. 9: **All mappings of the "vehicles"-example.**

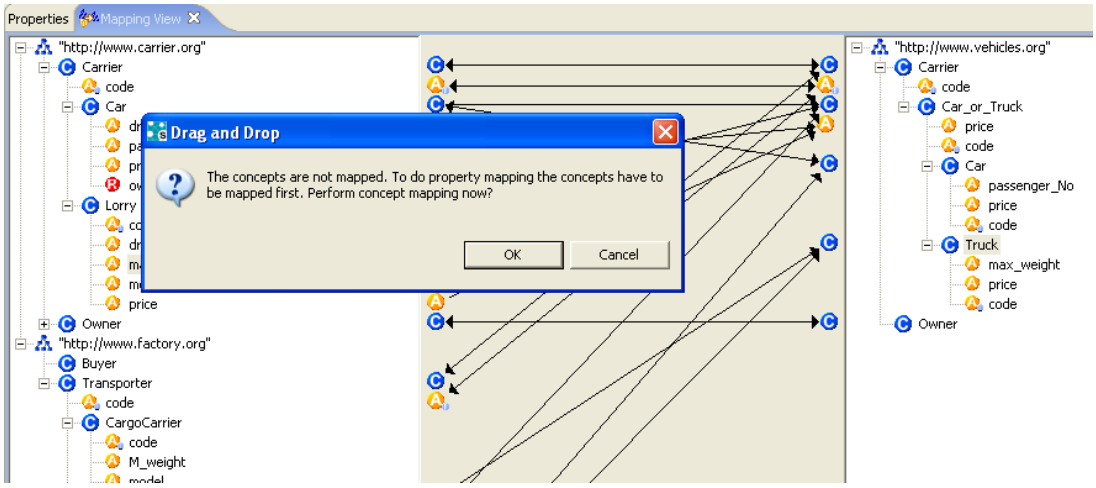

Fig. 10: **Warning-message for missing concept-mappings.**

4.1.4 Specify conditional mappings

OntoMap provides the possibility to specify constraints for concept-mappings. Currently constraints on attributes can be stated. Let's take the mappings onto the "Car_or_Truck" concept of the vehicles-ontology as an example. Those mappings are defined only in one direction, since every "truck" is a "Car_or_Truck" but not every "Car_or_Truck" is a "truck". In some countries, the definition of a truck depends on

the own weight of the vehicle. Let's assume that's the only constraint here (neglecting other aspects such as the number of axes for example). In such a case it would be possible to map "Car_or_Truck" on "car" under the constraint that the own weight is over the limit for "non-trucks".

To achieve this, we select the mapping of "Car_or_Truck" onto Truck (factory-ontology) by clicking on the arrow representing that mapping in the mapping view. OntoMap will display the form for mapping-constraints ("filters"), as shown in figure 11. To specify a filter, we select the respective attribute in the most left combo-box of the "Filters" row. Then we specify the constraint on that attribute using the next combo-box and the input-field. By pressing "Apply" we add the new constraint to the list.

The mapping view indicates mappings using constraints with a filter symbol (see figure 19).

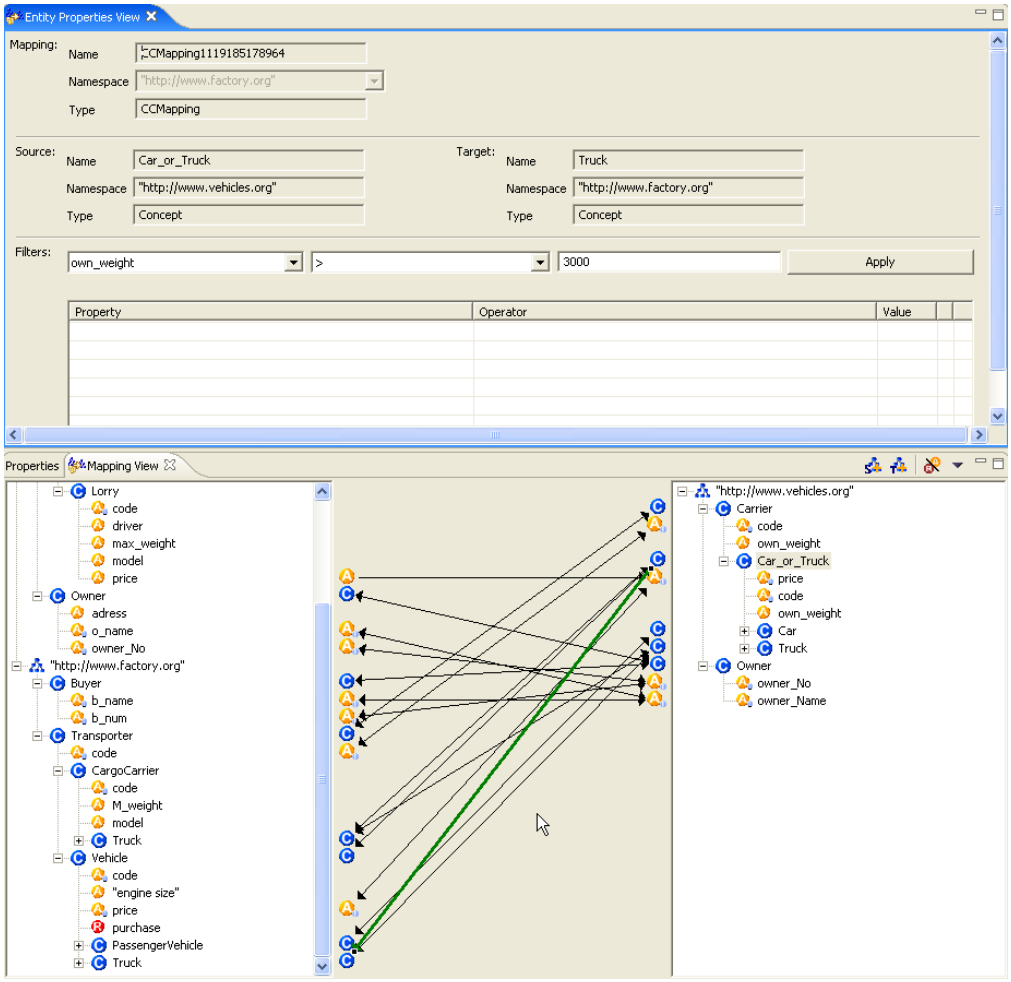

Fig. 11: **Specification of Constraints for conditional mappings.**

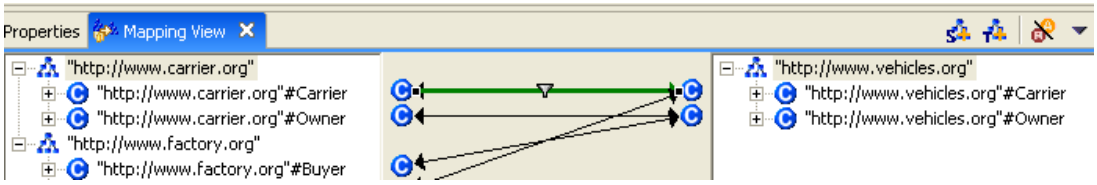

Fig. 12: **Filter symbol for conditional mappings**

4.1.5 Check the resulting model

Checking the result of the created mapping is quiet straightforward. We create instances of the mapped concepts in the different ontologies. In a second step we create queries on those concepts and check the results.

We create for example a “lorry”-instance in the carrier ontology and a “truck”-instance in the factory ontology. Then we create a new query for the “lorry”-concept. We select the concept in the schema-view, press the right mouse-button and choose “New Query” from the menu that appears (see figure 13). The new query appears in

the query-section of the schema view. We press the right mouse button and select "Execute Query". In the result view the instances for "lorry" and for "truck" should appear.

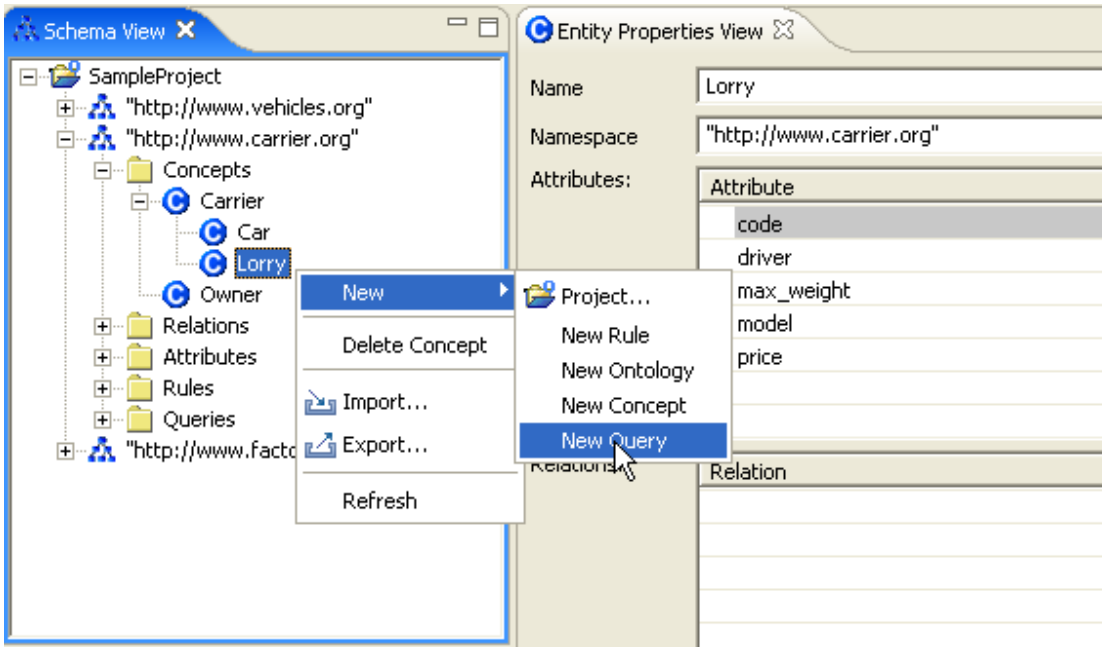

Fig. 13: **Creating a new query for the "lorry"-concept**

4.1.6 Delete Mappings

To delete a given mapping, we select the mapping (as described above) and use the right mouse-button to open a small menu that contains the option "Remove Mapping" as only item. By using the left mouse-button on the menu-item we remove the selected mapping.

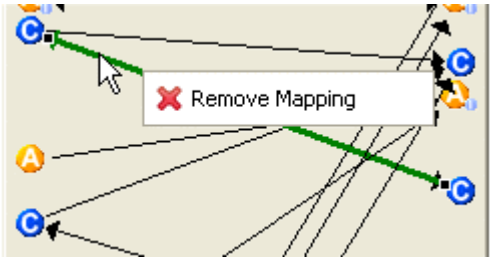

Fig. 14: **Removal of mappings using the graphical selection and the right mouse button.**

4.2 Automatic Mapping based on FOAM

The support for semi-automatic mappings is provided by the FOAM framework (see section 3.1.3). A first integration of FOAM in OntoMap has been realized. Its application is straightforward.

To use the mapping-discovery provided by FOAM, we open the mapping-view and specify source- and target ontology as previously described (see section 4.1.2). In a second step, we directly start the mapping discovery by pressing the “map ontologies automatically”-Button (figure 15). The ontologies are then mapped based on the tuning parameters that specified for FOAM. OntoStudio uses default values for the parameters (such as the threshold for relevant mappings). Currently there is no graphical support for the configuration of FOAM within OntoMap.

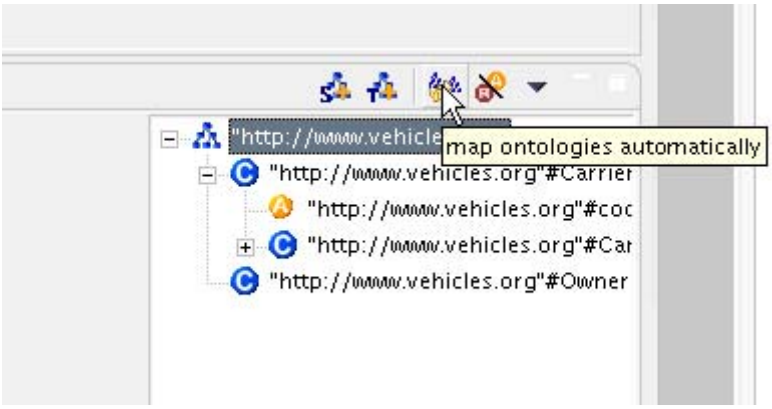

Fig. 15: **Activating the mapping discovery in the mapping view**

4.3 Mapping-Store Support

The connection between the mapping-store (see section 3.1.2) and OntoMap is based on import and export functions. OntoMap does currently not support the full range of mappings that can be represented and stored in the mapping store. This is important for the import of mappings, since all types of mappings a user can define in OntoMap can be represented in the mapping language as well.

The current version of the mapping-store is implemented on a directory- and file basis. For both, the import and the export of mappings (from and to the store) the root directory of the store has to be specified. In the case of an export, a new store can be set up, using an empty directory. To create access and use a mapping-store, the necessary user-rights for the directory-structure have to be provided.

The directory-structure of a mapping-store as well as the files contained in it should not be manually altered. A mapping-store should always be accessed via the mapping-store API or a third party tool (such as OntoMap) using that API.

4.3.1 Import mappings from a mapping store

Mappings of an existing mapping-store can be imported into OntoStudio (OntoMap) using a form-based import-filter. To activate the import, we select a project in the

Schema-View of OntoStudio and use the right mouse-button to open the menu shown in figure 16 and then choose the “Mapping-Store Import Wizard” (see figure 17).

In the next step, we choose the root-directory of the mapping-store that we want to import from as well as the target project for the import (figure 18). As described above, the respective directory has to be visible and the access-rights need to be set accordingly.

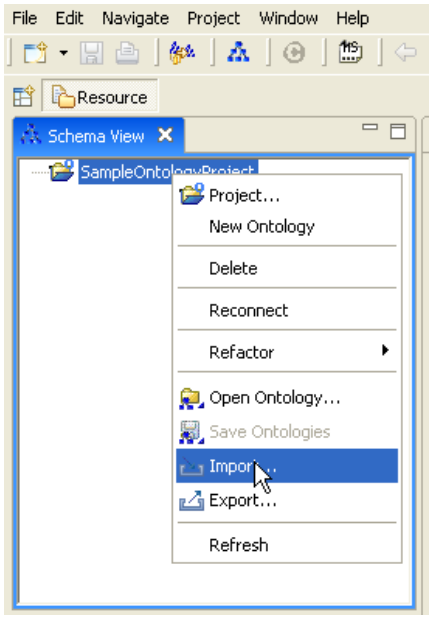

Fig. 16: **Activation of import menu.**

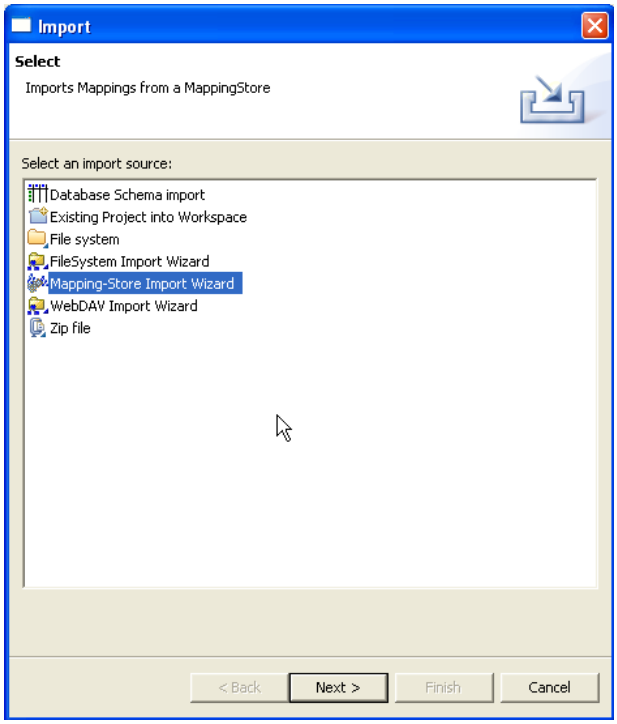

Fig. 17: **Selection of mapping-store import-wizard.**

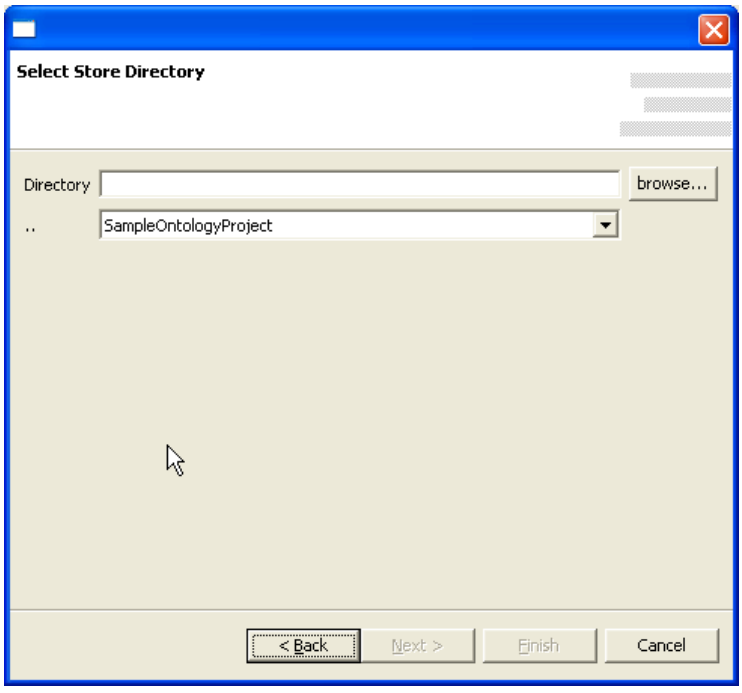

Fig. 18: **Selection of mapping store directory**

When the mapping-store root directory has been set, we press the “Next”-button to proceed with the retrieval of the mappings we want to import. Figure 19 shows the form which is used to specify filters for the mappings that are to be imported.

The upper section contains the control elements used to set up filters on one of the following properties:

- the name under which a mapping is stored;
- the description of a mapping;
- the name of the source-ontology;
- the name of the target-ontology.

The filters listed in the “Current Filters” list are all applied at the same time, i.e. they are internally combined with an “AND”-operator.

The lower section contains a list-control in which the mappings are displayed that are retrieved using the filters. A listed mapping can be removed by simply selecting the mapping in the list and press the “Remove Selected Mapping”-Button or the DEL-Key.

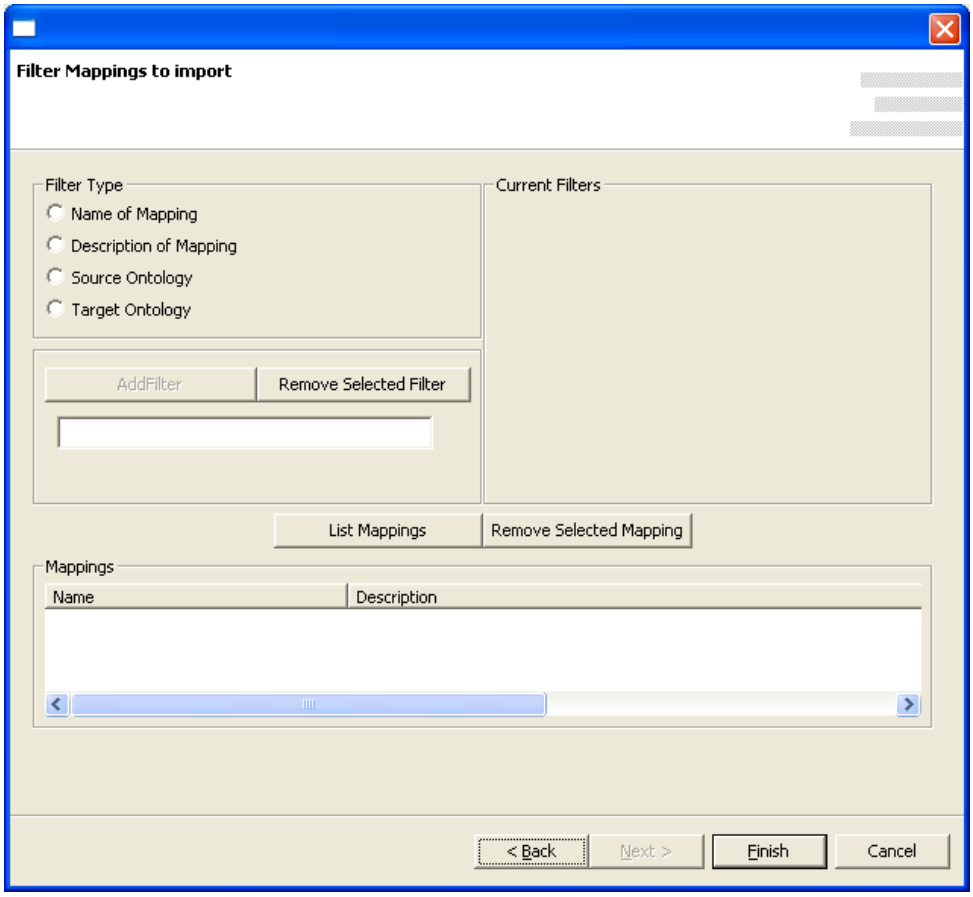

Fig. 19: **Filter for the mappings to be imported.**

For the first two categories, either the exact text or a regular expression for a fuzzy search can be used. Let's consider an example: We know that the mapping-store contains mappings from the carrier-domain and want to retrieve mappings for cars. In a first step, we try to narrow the search via the name.

To achieve this we

- activate the radio button "Name of Mapping",
- type ".*car.*" in the text-field below the "Add Filter"-Button
- and finally press the "Add Filter"-Button.

Then we add another filter to make sure that the mappings to be retrieved concern our target-ontology (which would be the vehicles-ontology in the example given in section 4.1). Therefore we activate the "Target Ontology" radio button. The text field for the filter text changes to a combo box from which we select the designated target ontology. We press the "Add Filter"-Button again and finally press the "List Mappings"-button. The retrieved mappings are then displayed in the list control (see figure 20).

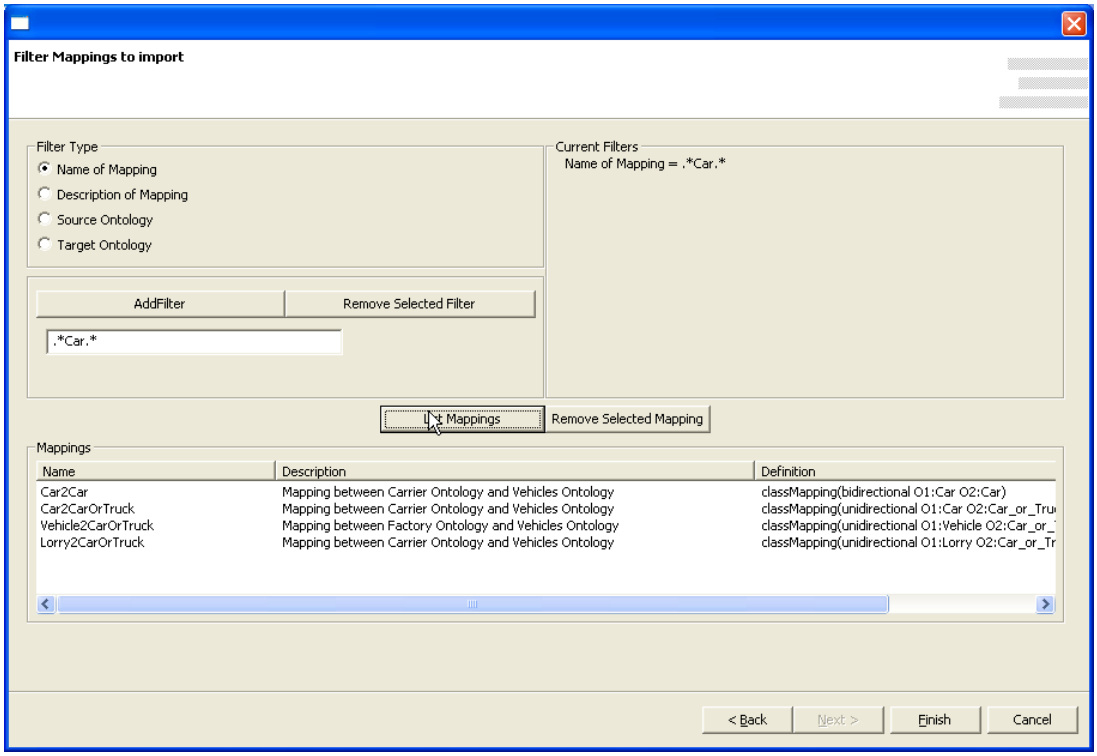

Fig. 20: **Retrieval of mappings containing "car" in the name.**

Another possibility would be to use the regular expression on the description of the mapping. Note that regular expressions provide a powerful instrument for filter-criteria, including logical operators. Examples are given in table 1 below.

Tab. 1: **Examples for regular expressions to be used for mapping retrieval**

Name/description contains the word "car" (anywhere)	.*car.*
Name/description starts with "car"	car.*
Name/description contains the word "car" or the word "truck"	.*car.*|.*truck.*
Name/description contains the word "car" and the word "truck"	.*car.*truck.*|.*truck.*car*

Finally the displayed mappings are imported by pressing the "Finish"-button. Currently the users will be informed about the number of mappings that could be imported successfully. In future there will be a hint about which mappings could not be imported.

4.3.2 Export mappings to a mapping-store

Mappings from a project in the OntoStudio workspace can be exported either to an existing mapping-store or into a new mapping-store which is created automatically. The user specifies the root directory of the mapping store and the ontology for which mappings should be exported. The export wizard is activated by opening the export-

menu similar to the import (see section 4.3.1) and choosing the mapping-store export (see figure 21).

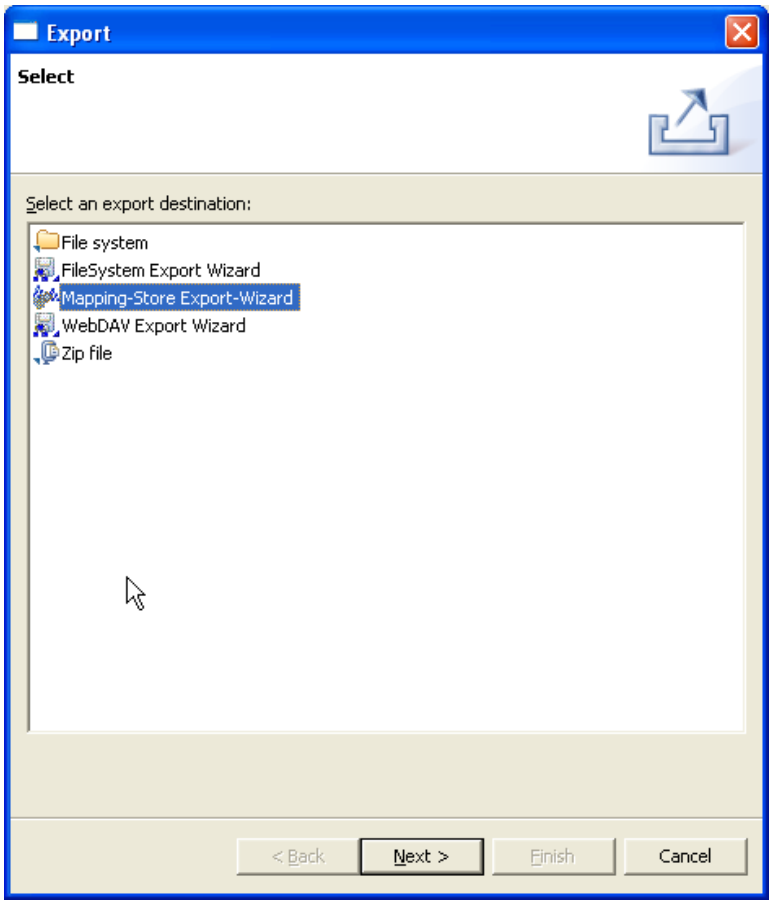

Fig. 21: **Activation of Mapping-Store Export-Wizard**

In the next step, we select the project and the ontology for which mappings should be exported (see figure 22). The root directory of the mapping store has to be specified as well. If the check-box “create a new mapping-store” is activated, a new mapping store will be created in that directory. If the directory is already used, any existing mapping-store will be overridden.

Pressing the “Finish”-Button starts the export of the mappings to the store, from which they are then available in the representation of the mapping language.

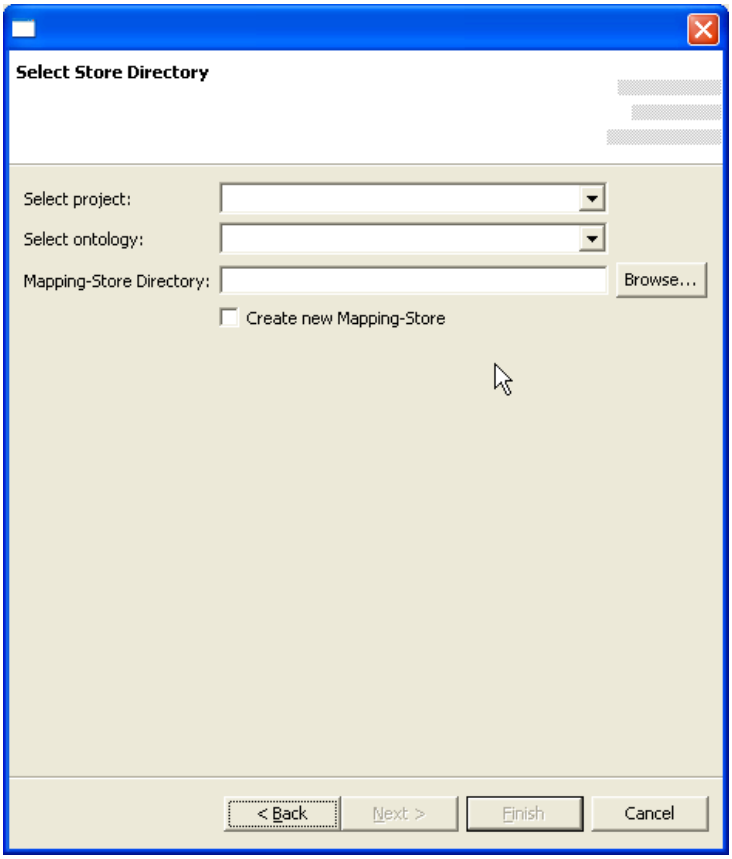

Fig. 22: **Settings for export to mapping-store**

5 Outlook

The current version of the ontology mediation software allows to create and manage ontology mappings in a graphical environment. It supports the import from and export to a mapping repository, which allows to retrieve mappings. The efficiency of the mapping process is increased by the application of the mapping-discovery. This covers basic aspects of ontology mediation. If a higher degree of flexibility is needed than the graphical environment provides, mappings have to be specified directly using either the internal representation language of OntoStudio (declaring mapping rules directly in F-Logic) or the neutral mapping-language.

Future versions of OntoMap might support additional mapping types, in as far as they can be supported by a graphical environment. The base for a support of a wide range of mapping patterns would be provided by a source editor, which is currently not part of OntoStudio. Once a source editor with capabilities such as auto-completion and code-temples is available, such a support will be targeted. Currently mapping rules can be directly defined as F-Logic rules based on the rule-editor that is available (simple text field with validation). However, those rules would not be identified by the systems to be mapping rules (and therefore not visualized accordingly). With the help of code-templates, complex mapping patterns that are hard to represent graphically could be made available to knowledge-engineers.

The current results of WP4 provide a conceptual base for mediation as well as a technical infrastructure for practical mediation problems. The implementation of the mapping language API and the mapping store allow to combine the approaches on the conceptual level (work on patterns) and engineering tools like OntoStudio. One of the next step will be to analyse mappings that have been defined for practical problems and compare them to the patterns developed in [deBrui05].

Further improvements will be made with regard to the interoperability of the different mediation components. A mapping export in terms of a grounding for the KAON2 engine is in development (to allow the execution of mappings). The integration of the mapping discovery currently requires temporary format transformations due to the different representations OntoStudio and the mapping discovery rely on. The project partners will evaluate the possibility to encapsulate the ontology-access within the discovery framework and implement a tighter integration in the modelling environment.

Bibliography and references

[deBrui04] DE BRUIJN, J.; LAUSEN, H.: Active Ontologies for Data Source Queries. In: *Proceedings of the 1st European Semantic Web Symposium (ESWS2004)* Heidelberg, 2004

[deBrui04b] DE BRUIJN, J.; MARTIN-RECUERDA, F.; MANOV, D.; EHRIG, M.: State-of-the-art survey on Ontology Merging and Aligning V1. Project Deliverable D4.2.1. 2004

[deBrui05] DE BRUIJN, J.; FOXVOG, D.; ZIMMERMAN, K.: Ontology Mediation Patterns Library V1. Project Deliverable D4.3.1. 2005

[Ehri2004] EHRIG, M.; STAAB, S.: Efficiency of Ontology Mapping Approaches. ., International Workshop on Semantic Intelligent Middleware for the Web and the Grid at ECAI 04, Valencia, Spain 2004

[Ehri2004a] EHRIG, M.; HAASE, P.; STOJANOVIC, N.; HEFKE, M.: Similarity for Ontologies - a Comprehensive Framework. ., Workshop Enterprise Modelling and Ontology: Ingredients for Interoperability (PAKM) 2004

[Ehri2004b] EHRIG, M.; SURE, Y.: Ontology Mapping - An Integrated Approach. . In: BUSSLER, C.; DAVIS, J.; FENSEL, D.; STUDER, R. (Eds.): *Proceedings of the First European Semantic Web Symposium* Heidelberg, 2004, pp 76-91

[Ehri2005] EHRIG, M.; SURE, Y.: Ontology Mapping by Axioms (OMA). ., Workshop Semantic Model Integration (SMI05), Kaiserslautern 2005

[FOAM2005] www.aifb.uni-karlsruhe.de/WBS/meh/foam/

[Kalf2005] KALFOGLOU, Y.; SCHORLEMMER, M.: Ontology mapping: the state of the art. In: IBFI (Eds.): *Semantic Interoperability and Integration* Dagstuhl, Germany, 2005

[Kife1997] KIFER, M. AND LAUSEN, G.: FLogic: A higher-order language for reasoning about objects. *SIGMOD Record*, Vol. 18 (1997) No. 6, pp 134-146

[Meis2002] MEISEL, H.; COMPATANGLO, E.: ER-ConcepTool: a "reasonable" environment for schema and ontology sharing. . In: IEEE COMPUTER SOCIETY PRESS (Eds.): *14th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'2002)*, 2002

[Mitr00] MITRA, P.; WIEDERHOLD, G.; KERSTEN, M.L.: A Graph-Oriented Model for Articulation of Ontology Interdependencies. . In: *Proc. of the VII Conf. on Extending Database Technology (EDBT'2000)* Heidelberg, 2000, pp 86-100

[owmg05] www.omwg.org/tools/

[Wenk2004] WENKE, D.; ATANASSOV, S; MANOV, D.; MAIER-COLLIN, M.; SPERLIN, W.: Report on Ontology Medation as service component. project deliverable D4.5.1 2004