Вісник Харківського національного університету
Серія «Математичне моделювання. Інформаційні технології. Автоматизовані системи управління»
№ , 2004, с.

УДК 519.6

Semantically Reinforced Web Services for Wrapping Autonomous Information Resources

Vadim Ermolayev, Natalya Keberle, Vladimir Shapar, Vladimir Vladimirov
Intelligent Systems Research Group,
Dept. of Information Technologies, Zaporozhye State Univ.,
66, Zhukovskogo st., 69063, Zaporozhye, Ukraine
{eva, kenga, wws, vvlad}@zsu.zp.ua

Abstract: The paper presents the use of semantically enhanced web services in the field of distributed intelligent information retrieval. The main idea of the approach is that a web service is used as an intelligent wrapper for an information resource (IR). These autonomous IRs become available for querying within distributed information system with centralized mediator through IR registration. IR wrapper web services provide homogeneous semantically reinforced query interface for the mediator trough the use of machine-processable ontologies. The paper reports on the architectures of the mediator systems in two projects that exploit IR wrapper web service approach. The architecture of the IR wrapper web service, the generic wrapper and its bindings is then presented. The technique is evaluated by experiments with implemented tester for the web service which wraps "University Entrant" IR of Zaporozhye State University.

1. Introduction

The importance of adding semantics to web service descriptions is becoming widely acknowledged. Reinforcing current industry standards for web service discovery and description, like UDDI and WSDL, with properly aligned and machine-processable specifications of web service semantics in the form of ontologies is one of the key targets of Semantic Web enabled Web Services research community. "Clarity in semantics together with a rich formalization are especially important for ontologies describing web services because they enable complex tasks involving multiply agents" (cf. [17]). As outlined also in [21], [8] reasonably formal ontological descriptions of web services may become a catalyst for automated web service discovery, composition, and orchestration on the Semantic Web.

The paper presents the use of web services reinforced with ontologies in the field of distributed intelligent information retrieval (I2R). This research is run in frame of the RACING[1] and UnIT-Net[2] projects. Both projects aim to implement intelligent mediator systems for querying distributed heterogeneous and disparately structured information resources (IRs) in the terms of a domain ontology which is also the common mediator ontology. The main idea of the approach is that a web service is used as an intelligent wrapper for such an IR and provides homogeneous semantically reinforced query interface for the mediator. The only function of such a web service is

[1] **RACING**: Rational Agent Coalitions for Intelligent Mediation of Information Retrieval on the Net. Project funded by Ukrainian Ministry of Education and Science. http://www.zsu.zp.ua/racing/

[2] **UnIT-Net**: IT in University Management Network. TEMPUS/TACIS project MP-JEP-2010-2003. http://www.unit-net.org.ua/

to translate and to perform queries to the wrapped IR. Hence, it is considered that instead of registering a web service it is more reasonable to semi-automatically register the wrapped IR by aligning, mapping and merging its information resource ontology (IRO) to the mediator domain ontology (MDO). The mappings are collected at the mediator side in IR – domain mapping ontology (IRDMO). IRDMO is further used by the mediator for automatic extraction of sub-queries (to specific registered IRs), for the translation of a user query to the terms of respective IROs, and for finding out which wrapper web service is capable to perform the sub-query. The discovery of a web service to perform the sub-query is more straightforward in UnIT-Net IEDI – IRDMO stores the URIs of the wrappers for the registered IRs and sub-queries are generated for all IRs possessing relevant semantic capabilities. More sophisticated technique is used in RACING. RACING wrapper web services are the capabilities of resource wrapping agents (RWA). The decision on which IRs to query by a specific sub-query is taken as the result of negotiation among the query planning agent (QPA) and the RWAs with similar semantic capabilities.

The paper focuses on the web service implementation of the wrapper side of the mentioned mediator systems leaving the solutions of the mediator side problems for other publications (e.g., [10]). The reminder of the paper is structured as follows. Section 2 surveys the related work in the field of distributed I2R and semantically enhanced web services. Section 3 sketches out the architectures of RACING and UnIT-Net IEDI mediator-wrapper systems. Section 4 provides more details on the architectural solutions for web services wrapping IRs. Section 5 reports on the proof-of-concept web service wrapper implementation. Section 6 gives conclusions and outlines the future work.

2 Related Work

Examples of projects developing formal, algorithmic, architectural frameworks, deploying software prototypes for distributed I2R reinforced with the use of ontologies are BUSTER [22], DOME [4], InfoSleuth [2], KRAFT [13], MOMIS [3], OBSERVER [16], Ontobroker [5], PICSEL [15], SIMS [1], TSIMMIS [12]. Although all these projects use different techniques, approaches, and software paradigms they identify similar pitfalls for the domain. The first group of possible pitfalls is the way in which semantic heterogeneity is resolved in the processes of ontology-based information retrieval. As outlined in [4], this includes the aspects of developing ontologies (bottom-up and top-down approaches), mapping between ontologies, and relationships between ontologies and IRs.

Most projects adopt one of the following approaches to using ontologies [23]: single ontology (SIMS), multiple ontology (OBSERVER), hybrid approach (BUSTER, DOME). Mapping between ontologies is necessary when a system uses several ontologies either “horizontally” (as in multiple ontologies approach) or “vertically” (as in hybrid approach). Mappings between ontologies within the system provide links between equivalent or related elements of ontologies, thus ensuring ontology re-use. Mappings between ontologies and IR schemas maintain correspondences between ontology elements and the elements of IR data schemas. As stated in [4], the reasons for these mappings are: data schema definitions are not always a good source of domain knowledge for people querying the system, they often play technical role; queries posed to the system are expressed in the ontology-oriented

query language, but not in the terms of data schemas – mapping between ontology elements and data schema elements makes for transparent execution of user queries within the system; the requirements of information resource autonomy and openness of the system as a whole.

The second group of possible pitfalls concerns the aspects of supplying autonomy and dynamic nature of the IRs in an open system. The solutions here advocate one of the types of mediator architectures: centralized and decentralized. A centralized mediator architecture provides for one centre (e.g., TSIMMIS), which stores all the information about ontologies, IRs, mappings between them, and controls the query formulation and execution. A decentralized mediator architecture provides a separate agent/wrapper for each IR, which stores mappings between global/shared ontology(-ies) and the underlying IR. This approach is used in RACING [7]. In other projects (e.g., InfoSleuth, SIMS, KRAFT) the resource broker communicates with resource agents/wrappers and determines relevant and accessible resources for every query.

The third group of possible pitfalls is formed by the tasks of query formulation, effective query decomposition without loss of information and query results merging and refinement. Known approaches to solving these tasks are: use ontologies (hypernymy/hyponymy [16] and meronymy [7] relationships) to reformulate queries containing terms which do not exist in the ontology(-ies) thus constructing query plans with no loss of information; use rewriting techniques together with mappings to produce queries on IRs that most effectively satisfy the input query [15].

Another aspect to be analyzed in relevant research approaches is the use of web services as the interface to IRs. Although some authors claim web services to be appealing technology for the task (e.g. [6]) it is hard to find published research implementations of web service based approaches in distributed I2R, especially for intelligent wrapping of disparate IRs. The projects mentioned above use either agent communication facilities or CORBA as mediator-wrapper interfaces. A possible reason for that is that web services still lack means for appropriate formal specification of the semantics of themselves and the wrapped IR. The means improving semantic representation of web services are recently under intensive research. The few topical examples are OWL-S [19], Web Service Modeling Ontology (WSMO) [24].

The concept and the architectures of RACING and UnIT-Net IEDI use some novelties which, in their combination, distinguish them from their predecessors. Ontologies are specified in W3C emerging de facto standard language OWL DL [18]. Ontology-driven query formulation and transformation [7] is used for query processing at the mediator side. Mediator query language is RDQL [20]. At the wrapper side the semantics of a structured IR (e.g., RDB) is formalized by means of a semi-structured Ontology Specification Language (OWL DL)**.** Web Service technology is used for IR wrappers implementation. Web service registration is substituted by the wrapped IR registration which provides more powerful means to discover an appropriate web service for querying.

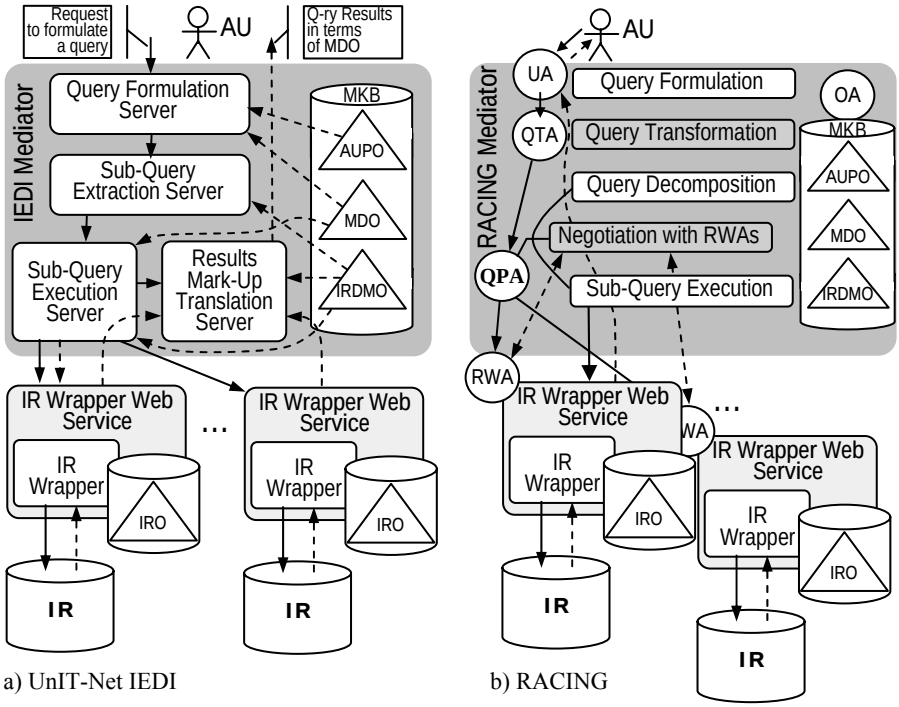


Fig. 1. UnIT-Net IEDI and RACING reference architectures.

3 UnIT-Net IEDI and RACING Architectures

Both UnIT-Net IEDI and RACING are distributed software systems for I2R. The type of their architectures is mediator-wrapper with centralized mediator and hybrid ontology utilization. The fact which is centric to the topic of the paper is that both systems use semantically enabled web services as IR Wrappers for query processing (Section 4).

The architectures of UnIT-Net IEDI and RACING mediators are outlined in Fig. 1. Both of them are based on the following procedure for the ontology-driven query processing. A human user, who is authorized to pose queries to a system (AU), uses query formulation tool [7] to specify a query in terms of MDO. AU Profile Ontology (AUPO) is used to adjust his or her terminological preferences to the terms of MDO as described in [7]. Query Formulation Server provides for this functionality in IEDI. In case of RACING this task is guided by User Agent (UA) and by Query Transformation Agent (QTA). The output of this initial activity is the RDQL query in terms of MDO.

At the next step the query is automatically decomposed into RDQL sub-queries – one per relevant IR or IR group with similar IROs. In IEDI this activity is performed by Sub-Query Extraction Server. In RACING – it is the first phase of query planning by QPA. Both systems use sub-query extraction algorithm based on late binding technique and IRDMO usage as described in [10].

At next it is determined which IR wrappers will be involved in the processing of the cluster of generated sub-queries. In IEDI the solution is determined by the sub-query extraction. The wrapper web service of each registered IR, for which a non-

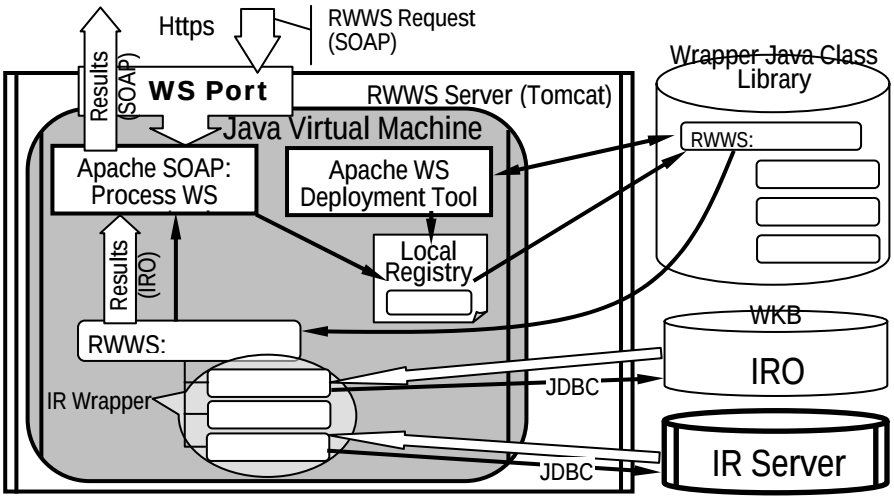


Fig. 2. Generic Wrapper Server architecture.

empty sub-query was generated, will receive the sub-query for execution. In RACING the activity involves intermediate negotiation phase. QPA as the initiator of this negotiation first consults with RACING Matchmaker Agent (MA) to determine the list of negotiation participants – RWAs who are capable to perform the sub-query according to the IROs of their IRs. At next QPA conducts Contract Net negotiation on sub-query performance web service provision with these RWAs [8] and determines the contractor RWA. The negotiation set includes time, price and quality of service conditions of sub-query performance. Contractor RWA supplies QPA with the URI of the requested web service for sub-query performance.

After the sub-queries are performed the results in terms of IR IROs are translated into the terms of the MDO and delivered to AU.

One more difference between IEDI and RACING architectures is in the way of ontology provision. Both systems use IR registration for incremental MDO and IRDMO construction. In both systems the process of aligning and merging the IRO of an IR under registration to MDO is performed by two ontology engineers with the help of the Ontology Negotiation Tool (under development in UnIT-Net) which is the kind of an enhanced ontology editor. However, Mediator Knowledge Base (MKB) in RACING is managed by the Ontology Agent (OA) which provides other agents with the portions of the ontologies on their requests. In IEDI the access of mediator servers to MKB is performed through Jena API [14].

4 Web Services for IR Wrapping

As it was outlined before the task of an IR Wrapper Web Service (RWWS) is to perform RDQL queries received from the Mediator. RWWS, together with its sub-ordinate functional components, are implemented as Java classes compiled to byte code and are executed by JVM of the RWWS Server. Tomcat servlet container[3] is chosen as the RWWS Server for IEDI and RACING prototype implementation. The architecture of the generic RWWS Server and the process of RWWS execution are presented in Fig. 2.

[3] http://jakarta.apache.org/tomcat/index.html

Before an RWWS could be executed by JVM it should be deployed by Apache SOAP Web Service[4] Deployment Tool. The deployment results in placing the code of the RWWS class(es) compiled to byte code to the Wrapper Java Class Library and in RWWS registration at the local Apache web service registry.

The requests to perform a WS are conveyed by means of the secured protocol ([11], Section 8). When a SOAP request to perform an RWWS comes to the RWWS port of the server it is processed by the Apache SOAP processing service. SOAP processing service extracts RWWS invocation data from the SOAP envelope, checks if the requested service is available at the local registry and invokes the execution of the RWWS at the JVM. The components of the RWWS are the ones of the IR Wrapper (Section 4.1). IR Wrapper components use Jena API to interact with the IR Wrapper Knowledge Base (WKB) through JDBC. They interact with the IR by means of JDBC.

4.1 IR Wrappers

The function of an IR Wrapper in IEDI and RACING is to provide uniform access to registered IRs. The wrappers for the specific IRs are designed and deployed according to the architectural pattern provided by the Generic IR Wrapper Architecture Specification ([11], Section 4.4.1).

4.1.1 A Generic IR Wrapper and IR Wrapper Binding

Generic IR Wrapper is the architectural abstraction and the software pattern for constructing and deploying IR Wrappers in the process of the preparation to IR registration ([11], Section 2.3). Its main function is to provide the implementation framework for the uniform access to respective IR. The implementation of this function comprises:

- Provision of the IRO (coded in OWL) describing the semantics of the resource and terminological mapping from IR Schema to IRO. IRO constitutes Wrapper Knowledge Base (WKB)
- Provision of the web service as the interface to query the IR by IEDI Mediator
- Provision of the terminology translation component. RDQL query in terms of the IRO should be translated to the RDQL query in terms of the IR schema (if there is the schema: for example, the IR is the relational data base)
- Provision of the query language translation component. An RDQL query should be translated to the Query Language of the IR (IRQL)
- Provision of the component which will actually order the execution of IRQL queries through JDBC interface to the IR Server
- Provision of the component which will mark-up the result of the query in the terms of the IRO

IEDI Generic Wrapper architecture is shown on Fig. 3. It comprises both IR invariant (web service, terminology translation, query result mark-up) and IR specific (WKB, language translation, query execution) components. IR specific components are further on referred to as IR Wrapper Binding. IEDI Generic Wrapper implementation thus provides the skeleton for the specific IR Wrapper implementation. The implementation of IR Wrapper Binding finalizes IR Wrapper deployment. One of the tasks of UnIT-Net project is to collect and to maintain the

[4] http://ws.apache.org/soap/features.html

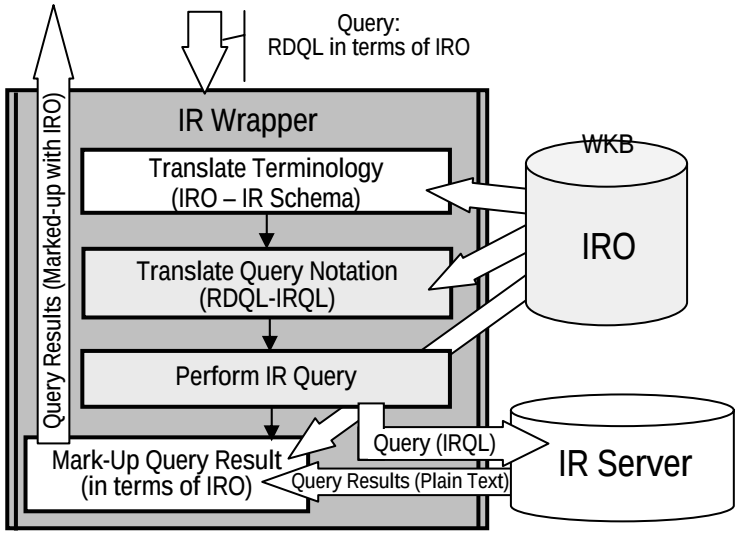


Fig. 3. Generic IR Wrapper architecture.

library of IR Wrapper Bindings for different types of IRs on the principles of Open Source licensing.

4.2 Query Translation Component for MS SQL Server IR Wrapper Binding

A specific IR Wrapper Binding for the IR of Zaporozhye State University containing university entrant information in the Relational Data Base managed by MS SQL Server has been developed in frame of UnIT-Net project. Its query language translation component is implemented according to the following algorithm.

Input: RDQL Query in the terms of University Entrant DB Schema

Output: SQL Query

Pre-conditions: ---

Post-effects: ---

Function: the algorithm performs the translation of the input query to the output query.

1. Form SELECT clause of the output query:

For each variable name in the SELECT clause of the input query replace variable names by corresponding slot names taken from WHERE clause of the input query.

2. Form FROM clause of the SQL query:

Add all table names used in QUERY (SELECT and WHERE clauses) to FROM clause.

3. Form WHERE clause of the SQL query:

For each triple in the triple list section of RDQL WHERE clause (<?x, TableName.FieldName, ?y>) of the input query

If ?y is found in the AND section of RDQL WHERE clause

For each entry of ?y (?y <OP> <value>)

in the AND section of RDQL WHERE clause

Add (TableName.FieldName <OP> <value>) together with the corresponding logical connective (AND or OR) to the WHERE clause

of the output query.
Remove the used entry of `?y` (`?y <OP> <value>`)
together with the corresponding logical connective
from the AND section of the RDQL WHERE clause.
End For
End If
Remove the processed triple from the triple list section
of the RDQL WHERE clause of the input query
End For

4. Form WHERE clause of the SQL query:
For each pair of table names from the SQL SELECT clause
If the pair appears as the **`PrimaryTable`** – **`ForeignTable`** pair
in the **IRO Relationships** Section
Add corresponding **`Expression`**
from ***IRO Relationships*** Section
to the SQL WHERE clause
(connect with AND logical connective)
End If
End For

5. Clean-up:
Remove USING clause of input query.

5 Proof-of-Concept Implementation

A proof-of-concept tester application has been developed to perform evaluation experiments with MS SQL DB RWWS. "University Entrant" database server of Zaporozhye State University was used as the IR for our experiments. The tester has been designed as a lightweight CGI client for a web browser. Its appearance is presented on Fig. 4. Tester application allows to input RDQL queries. It than either translates the input query if `#SQLONLY` operator is found in the input stream, or translates, performs the query and marks-up the result in terms of the IRO otherwise.

A simplified "University Entrant" IRO was designed for evaluation purposes. Its graphical representation is given on Fig. 5. It was assumed in the evaluation experiments that a user is responsible to formulate and to input RDQL queries in terms of this IRO. However, like having a recipe doesn't yet grant having a meal, having a deployed RWWS doesn't yet ensure that it reasonably correctly performs an arbitrary input query. In our context it was necessary to evaluate if the wrapper web service correctly translates and performs both "good" and "bad" queries – i.e. it is save with respect to the incorrect or partially correct input. By a "good" query we understood an RDQL query in terms of the given IRO which corresponds to the connected Concept Graph (CG). The following **Query (a)** is a "good" query:

Query (a).
In Natural language:
Return records of all University entrants who have got the maximal grade (9.0)
at the entrance examination in Mathematics. Display: surname, name, patronymic.
RDQL:

```
SELECT ?surname, ?secondName, ?aboName
WHERE (?x, abo:aboName, ?aboName),
```

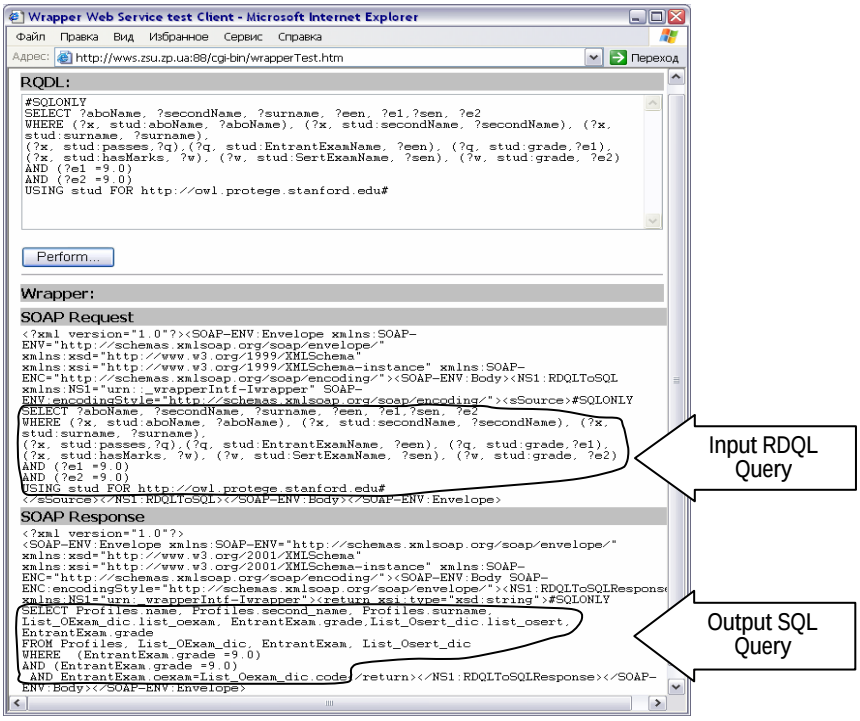


Fig. 4. RWWS tester implementation.
Displayed are the results of RDQL-SQL query translation.

```
(?x, abo:secondName, ?secondName), (?x, abo:surname,
?surname),(?a, abo:forProfile, ?x), (?a, abo:passes, ?y),
(?y, abo:EntrantExamName, ?een),
(?y, abo:grade, ?z)
AND (?z eq '9.0')
AND (?een eq 'MATHEMATICS')
USING abo FOR <http://owl.protege.stanford.edu#>
```

Query CG is given in Fig. 6(a).

Query (b) is an example of a "bad" query because the corresponding CG is disjoint – please refer to Fig. 6(b).

Query (b).

In natural language:

Return all names of the subjects for which the results of School Certification Examinations are accounted for the enrolment decision[5] *with respect to the entrants to Computer Science. Display Speciality Name, Certification Exam Name.*

RDQL:

```
SELECT ?specName, ?sen
WHERE  (?q, stud:CertExamName, ?sen),
(?s, stud:SpecialityName, ?specName)
AND (?specName eq 'COMPUTER SCIENCE')
```

[5] According to Ukrainian National rules for University Entrance procedure the average result of School Certification Exams on several characteristic subjects is used as extra examination mark of a University entrant and, thus, are used to make proper enrolment decision. For example, characteristic subjects for Computer Science entrants are Algebra, Geometry, Informatics, and Physics.

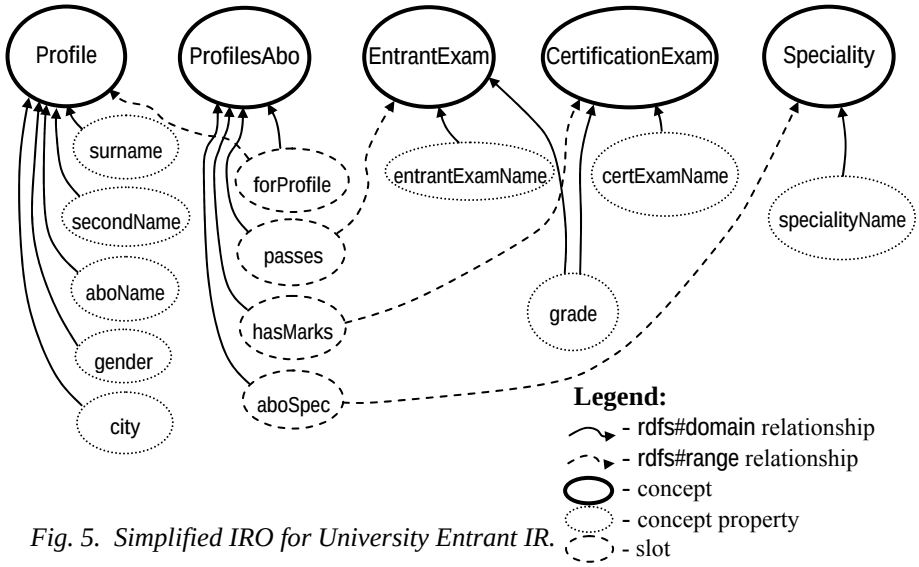


Fig. 5. Simplified IRO for University Entrant IR.

```
USING stud FOR <http://owl.protege.stanford.edu#>
```

This query is "bad" because it doesn't specify the relationship between CertificationExam and Speciality concepts of our IRO. However, IRO provides such a relationship by means of hasMarks and aboSpec slots (see Fig. 5). It is therefore possible to make this "bad" query a "good" one by the following reformulation (Query (c)).

Query (c).

In natural language:

Return all names of the subjects for which the results of School Certification Examinations obtained by the registered University Computer Science entrants are accounted for the enrolment decision ~~with respect to the entrants to Computer Science~~. Display Speciality Name, Certification Exam Name.

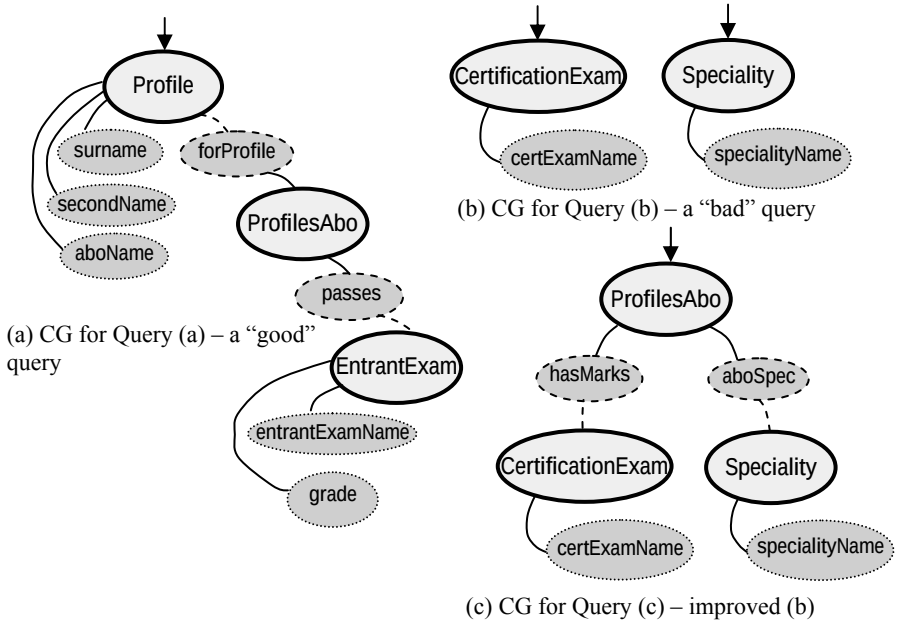
RDQL:

```
SELECT ?specName, ?cen
WHERE  (?x, abo:hasMarks,?q),(?q, abo:SertExamName,
?cen),(?x, abo:aboSpec, ?s), (?s, abo:SpecialityName,
?specName)
AND (?specName eq 'COMPUTER SCIENCE')
USING abo FOR <http://owl.protege.stanford.edu#>
```

The experiments showed that our RWWS has done well both for "good" and "bad" queries. For our example queries the results of query translation from RDQL to SQL are as follows.

Query (a) - SQL:

```
SELECT Profiles.surname, Profiles.second_name, Profiles.name
FROM Profiles, Profilesabo, List_OExam_dic, EntrantExam
WHERE  (EntrantExam.grade='9.0')
AND (List_OExam_dic.list_oexam='MATHEMATICS')
AND ProfilesAbo.profiles=Profiles.code AND
EntrantExam.profilesabo=ProfilesAbo.code AND
EntrantExam.oexam=List_Oexam_dic.code
```



(a) CG for Query (a) – a "good" query

(b) CG for Query (b) – a "bad" query

(c) CG for Query (c) – improved (b)

Fig. 6. RDQL Query Concept Graphs imposed on the "University Entrant" IRO for "good" and "bad" queries.

Query (b) - SQL:

```
SELECT DISTINCT Speciality_dic.speciality,
List_Osert_dic.list_ocert
FROM List_Ocert_dic, Speciality_dic
WHERE (Speciality_dic.speciality='COMPUTER SCIENCE')
```

The experiments with about 20 different "good" queries showed that the evaluated RWWS always translated the query adequately and returned expected results. When a "bad" query was fed to the RWWS it still performed reasonably adequate returning Cartesian Products of the possible correct responses to respective query CG branches. For example the response to Query (b) was as follows:

```
<?xml version="1.0"?>
<SOAP-ENV:Envelope xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/">
<SOAP-ENV:Body SOAP-
ENC:encodingStyle="http://schemas.xmlsoap.org/soap/envelope/">
<NS1:RDQLToSQLResponse xmlns:NS1="urn:_wrapperIntf-Iwrapper"><return
xsi:type="xsd:string">&lt;?xml version="1.0" standalone="yes"?&gt;&lt;!--
Generated by SMExport --&gt;  &lt;DATAPACKET
Version="2.0"&gt;&lt;METADATA&gt;&lt;FIELDS&gt;&lt;FIELD
attrname="SpecialityName" fieldtype="string" WIDTH="80"/&gt;&lt;FIELD
attrname="SertExamName" fieldtype="string"
WIDTH="80"/&gt;&lt;/FIELDS&gt;&lt;PARAMS DEFAULT_ORDER="1" PRIMARY_KEY="1"
LCID="1033"/&gt;&lt;/METADATA&gt;&lt;ROWDATA&gt;&lt;ROW
SpecialityName="COMPUTER SCIENCE" SertExamName="INFORMATICS"/&gt;&lt;ROW
SpecialityName="COMPUTER SCIENCE"
SertExamName="FOREIGN LANGUAGE"/&gt;&lt;ROW
SpecialityName="COMPUTER SCIENCE"
SertExamName="HISTORY"/&gt;&lt;ROW
SpecialityName="COMPUTER SCIENCE"
```

```
SertExamName="ALGEBRA"/&gt;&lt;ROW
…
SpecialityName="COMPUTER SCIENCE"
SertExamName="PHYSICAL CULTURE"/&gt;&lt;ROW
SpecialityName="COMPUTER SCIENCE"
SertExamName="PHYSICS"/&gt;&lt;ROW
DATA&gt;&lt;/DATAPACKET&gt;</return>
</NS1:RDQLToSQLResponse></SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

We may observe that:

- It still contains the correct intended response (INFORMATICS, ALGEBRA, GEOMETRY, PHYSICS)
- But it returns also all other names of Certification Exams for other University specialities

6. Concluding Remarks

The paper presented the use of RWWS reinforced with ontologies in the field of distributed intelligent information retrieval (I2R) in frame of the RACING and UnIT-Net projects. Both projects aim to implement intelligent mediator systems for querying distributed heterogeneous and disparately structured IRs in the terms of the common mediator ontology. The main idea of the approach is that a web service is used as an intelligent wrapper for such an IR and provides homogeneous semantically reinforced query interface for the mediator. One of the novelties which distinguishes RACING and UnIT-Net architectures from many other relevant approaches is the use of Web Service technology for IR wrappers implementation. An RWWS is a very simple web service from the point of view of the interface and invocation (SOAP). However, the interior of an RWWS is the ontologically reinforced IR wrapper which provides the following functional components:

- Terminology translation component: RDQL query in terms of the IRO is translated to the RDQL query in terms of the IR schema (if there is the schema: for example, the IR is the relational data base).
- Query language translation component: an RDQL query is translated to the Query Language of the IR (IRQL), for example to SQL in the Proof-of-Concept implementation.
- The component which actually orders the execution of IRQL queries through JDBC interface to the IR Server
- The component which marks-up the result of the query in the terms of the IRO

It was also reported in the paper that the registration of an RWWS is actually substituted by the registration of the IRO of the wrapped IR to the mediator. It is stated that such registration procedure facilitates to making the description of an RWWS more semantically rich and, hence, making the discovery of the necessary RWWS more simple and precise.

A proof-of-concept tester application has been developed to perform evaluation experiments with MS SQL DB RWWS. “University Entrant” database server of Zaporozhye State University was used as the IR in evaluation experiments. The experiments showed that the RWWS implementation did well both while translating and performing well formulated (with connected CG) and badly formulated RDQL queries in terms of the simplified “University Entrant” IRO.

Acknowledgements

The authors would like to express their gratitude to the members of the RACING and UnIT-Net project consortia for their collaborative help in bringing the reported results to life.

REFERENCES

1. Arens, Y.; Knoblock, C.A.; Shen, W.: Query Reformulation for Dynamic Information Integration. J. of Intelligent Information Systems. 1996.
2. Bayardo et al.: InfoSleuth: Semantic Integration of Information in Open and Dynamic Environment. In Proceedings of the 1997 ACM International Conference on the Management of Data (SIGMOD), Tucson, Arizona, May 1997.
3. Bergamaschi, S. et al.: An Intelligent Approach to Information Integration. In: Proc. of Formal Ontology in Information Systems (FOIS-98), June, 1998.
4. Cui, Z.; Jones, D.; O'Brien, P.: Semantic B2B Integration: Issues in Ontology-based Applications. SIGMOD Record, 31(1), March 2002. 43-48
5. Decker, S. et al.: Ontobroker: Ontology Based Access to Distributed and Semi-Structured Information. In R. Meersman et al. (eds.): Semantic Issues in Multimedia Systems. Proceedings of DS-8. Kluwer Academic Publisher, Boston, 1999, 351-369.
6. Demetriou, G. Et al.: A Web Services Architecture for Distributed Cross-Language Information Retrieval. Submitted to: J. of Natural Language Engineering, 2003
7. Ermolayev, V., Keberle, N., Plaksin, S., Vladimirov. V.: Capturing Semantics from Search Phrases: Incremental User Personification and Ontology-Driven Query Transformation. In: Proc. of the 2-nd Int. Conf. on Information Systems Technology and its Applications (ISTA'2003), Kharkiv, Ukraine, June 19-21, 2003, 9-20.
8. Ermolayev, V., Keberle, N., Kononenko, O., Plaksin, S., Terziyan, V.: Towards a framework for agent-enabled semantic web service composition. Int. J. of Web Services Research, 1(3), 2004, 63-87.
9. Ermolayev, V., Spivakovsky, A., Zholtkevych, G.: UnIT-NET IIDE : Infrastructure nationale ukrainienne pour l'intraéchange de données électroniques. Colloque National de la Recherche Universitaire dans les I. U. T. Actes de Colloque, Tome 1. Sciences et Techniques de l' Ingenieur, Nice, May, 6-7, 2004, p. 113-121
10. Ermolayev. V., Keberle, N., Shapar, V., Vladimirov, V.: Ontology-Driven Sub-Query Extraction for Distributed Autonomous Information Resources in UnIT-Net IEDI. 3-d Intl. Conference on Information Systems Technology and its Applications (ISTA'2004), Salt Lake City, Utah, USA, July 14-16, 2004 (to appear).
11. Ermolayev, V., et al.: The Infrastructure for Electronic Data Interchange. Reference Architecture Specification. Version 1.0. UNIT-NET Deliverable No D2.2.D.1. URL: http://www.compscipreprints.com/comp/Preprint/eva/20040228/1
12. Garcia-Molino, H. et. al.: The TSIMMIS Approach to Mediation: Data Models and Languages. In: Proc. Next Generation Information Technologies and Systems (NGITS), June 1995.

13. Gray. P. et al.: KRAFT: Knowledge Fusion From Distributed Databases and Knowledge Bases. In: Proc. 8th Intl. Workshop on Database and Expert System Applications (DEXA-97), IEEE Press, 1997, 682-691.
14. Jena – a Semantic Web Framework for Java. URL: http://jena.sourceforge.net/ (last checked: 27.04.2004)
15. Lattes V.; Rousset M.-C.: The Use of CARIN Language and Algorithms for Information Integration: The PICSEL System. Int J. of Cooperative Information Systems, 9(4), 2000, 383-401.
16. Mena, E. et al.: OBSERVER: An Approach for Query Processing in Global Information Systems Based on Interoperation Across Pre-Existing Ontologies. Distributed and Parallel Databases 8(2), 2000, 223-271
17. Mika, P., Gangemi, A., Oberle, D., Sabou. M.: Foundations for Service Ontologies: Aligning OWL-S to DOLCE. In: Proc. 13-th Int. World Wide Web Conf. 2004, New York, NY USA, May 17-22, 2004 (to appear)
18. OWL Web Ontology Language Reference. W3C Proposed Recommendation. 15 December 2003. URL: http://www.w3.org/TR/owl-ref/ (last checked: 27.04.2004)
19. OWL-S: Semantic Mark-up for Web Services. Whitepaper. The OWL Services Coalition. URL: http://www.daml.org/services/owl-s/1.0/owl-s.pdf (last checked: 27.04.2004)
20. RDQL – A Query Language for RDF. W3C Member Submission, 9 January 2004, URL: http://www.w3.org/Submission/2004/SUBM-RDQL-20040109/ (last checked: 27.04.2004)
21. Sivashanmugam, K., Verma, K., Sheth, A. P., Miller, J. A.: Adding Semantics to Web Services Standards. In: Zhang, L.-J. (Ed.): Proc. Int. Conference on Web Services, ICWS '03, June 23 - 26, 2003, Las Vegas, Nevada, USA, 395-401
22. Stuckenschmidt H. et al.: Enabling technologies for interoperability. In: Visser, U., Pundt H. (Eds.): Workshop on the 14th International Symposium of Computer Science for Environmental Protection, Bonn, Germany, 2000, 35-46.
23. Wache, H. et al.: Ontology-Based Integration of Information - A Survey of Existing Approaches. In: (A. Gomez-Perez, M. Gruninger, H. Stuckenschmidt, M. Uschold) Proceedings of the IJCAI-01 Workshop on Ontologies and Information Sharing, Seattle, USA, August 4-5, 2001, 108-118
24. Web Service Modeling Ontology (WSMO). DERI Working Draft 14 February 2004. URL:http://www.nextwebgeneration.org/projects/wsmo/2004/d2/v01/20040214