

Visual Semantic Query Formulation and Execution in UnIT-NET IEDI

Natalya Keberle¹, Vadim Ermolayev¹, Vladimir Vladimirov¹,
Eugeny Dzhurinsky²

¹*Intelligent Systems Research Group,
Dept. of Information Technologies, Zaporozhye National Univ.,
66, Zhukovskogo st., 69063, Zaporozhye, Ukraine*

{eva, kenga, vvlad}@zsu.zp.ua

²*JDevelop.com, Zaporozhye, Ukraine,
eugenydzh@jdevelop.com*

UnIT-NET Infrastructure for Electronic Data Interchange (IEDI) is the multi-layered distributed software system for providing intelligent ontology-driven information retrieval from distributed, heterogeneous, autonomous information resources (IRs). IEDI is constructed according to the ideology of mediator-wrapper architecture with single centralized mediator. The paper presents the results of evaluation of the UnIT-NET IEDI research prototype. Two autonomous distributed IRs - "University Entrant" IR of Zaporozhye National University and "Dean's Office" IR of V. Karazin Kharkiv National University were used as a testbed. Experiments have shown practical applicability of the approach proposed for the IEDI architecture. One of the stages of the evaluation procedure is the assessment of the quality of interface between users and IEDI mediator. The paper analyses lessons learned from the evaluation of UnIT-NET IEDI user interface component – Query Formulation Interface. The analysis proves the usability of the QFI; however there are refinements which have still to be considered in the future work.

1. Introduction

The goal of a data integration system is to provide a uniform interface to various data sources [1], and to enable users to focus on specifying the questions they have.

UnIT-Net¹ IEDI is the software infrastructure providing for the Electronic Data Interchange between the Universities and the State Bodies of Ukraine. More precisely, IEDI is the multi-layered distributed software system comprising the software servers, services, components and tools for providing intelligent ontology-driven information retrieval from distributed, heterogeneous, legally and physically autonomous IR in the frame of the organizational network of the National Higher Education System.

The task of IEDI mediator developed within UnIT-Net project is to provide uniform, consistent and user-friendly interface to retrieve information from distributed autonomous information resources (IRs).

The paper reports on the evaluation of UnIT-NET IEDI research prototype implementation. One of the stages of the evaluation procedure is the assessment of the

¹ **UnIT-Net:** IT in University Management Network. TEMPUS/TACIS project MP-JEP-2010-2003.
<http://www.unit-net.org.ua/>

quality of interface between users and IEDI mediator. The paper presents the study of capabilities of Query Formulation Interface (QFI).

QFI – is a graphical tool which is the UnIT-NET IEDI user interface – uses visual presentation of mediator domain ontology and information resources’ ontologies in the form of a tree and in the form of graph; it allows to add concepts, properties and additional constraints to a user query under construction; it stores user query using the concept of “basket”; it generates initial query to the mediator, initiates sub-queries extraction for the underlying IRs, collects responses and visually presents the responses to the user.

The paper is structured as follows: Section 2 discussed the IEDI framework and its main tasks; Section 3 describes IEDI environment chosen to bring the framework to life; Section 4 describes QFI in details; in the Section 5 presented are the results of experiments with IEDI; Section 6 presents the analysis of experiment results; Section 7 surveys the related work; Section 8 concludes the paper.

2 UnIT-Net IEDI Framework

From the variety of organizational platforms for integration of heterogeneous data sources (mediator-wrapper [1], federated database systems [2] etc) for UnIT-NET IEDI purposes only mediator-wrapper architecture gives the balance of centralized control and to a certain end a freedom in work with autonomous IRs.

Shortly describe the UnIT-NET IEDI architecture (see Fig. 1, also [3], [4]).

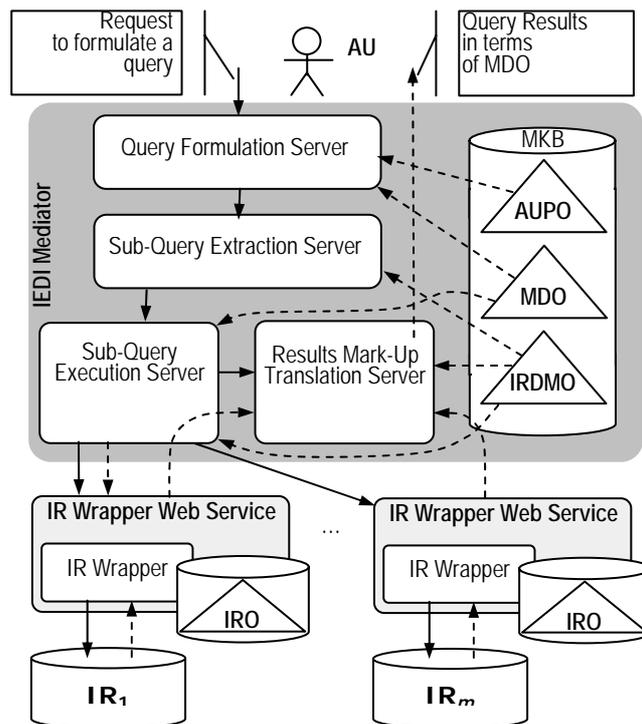


Figure 1. The reference architecture of the UnIT-NET IEDI

There is one IEDI mediator, its main tasks are:

- querying distributed autonomous semantically heterogeneous IRs (including user query formulation, query rewriting, results mark-up)
- registering IR
- maintaining coherent semantic descriptions

IRs are autonomous, distributed, heterogeneous and communicate with IEDI mediator by means of intelligent IR wrappers implemented as web services.

Each IR due to its autonomy has its own representation of the conceptualization of the domain, formally specified by IR conceptual schema. The hierarchy of IEDI ontologies has been developed to integrate these local views into the global view of the mediator.

Two main approaches to integrate data and answer queries without materializing a global schema [5] are “global-as-view” and “local-as-view”. Global-as-view approach prescribes for every domain conceptual schema concept to have a view over the IRs. On the contrary, local-as-view approach provides for each concept of an IR a view over the domain conceptual schema. Both approaches have their advantages and drawbacks; e.g. global-as-view is tolerant to answering complex queries and less flexible for maintaining frequently changed IRs; local-as-view approach is flexible with respect to changes in IRs, but its operation time increases exponentially to the number of IRs involved.

UnIT-NET IEDI explores “global-as-view” approach [6] and query rewriting technique ([7], [8]) to reformulate queries over IRs.

Domain knowledge is structured in the hierarchy of ontologies (see also [4], [9]). Each IR has own Information Resource Ontology (IRO), containing all essential (from the point of view of IR provider) concepts and properties of underlying IR domain. The goal of Mediator Domain Ontology (MDO) is to collect and merge all the knowledge from IROs, possibly adding necessary levels of abstraction.

Rewriting technique [1] makes use of mapping rules applicable to initial query to obtain new queries. In IEDI mapping rules are stored in mapping ontologies. IR-Domain Mapping Ontology (IRDMO, see Fig. 2a) stores mappings between IRO and MDO. IRO-IR Schema Mapping Ontology (see Fig. 2b) is constructed for every IR to store mappings of IR ontology terms to underlying structures of IR.

IRDMO is constructed in a way to contain the minimally necessary mappings for the MDO-IRO pairs. Only the mappings of non-inherited slots are stored for each MDO concept. IEDI mediator uses Late Binding technique [9] to detect all necessary slots of concepts during formulation of sub-queries to IROs.

All ontologies in UnIT-NET were developed with help of ontology editor Protégé 3.0² and are presented in OWL-DL [10], thus they have a sound and formal basis to verify consistence of ontologies.

3 IEDI Environment

All functional components of IEDI architecture are implemented in Java as web services. IR wrapper web services and IEDI mediator web service are run on Apache Tomcat³ Web server.

² Protégé 3.0 is available at the URL: <http://protege.stanford.edu>

³ Apache Tomcat Web server is available at the URL: <http://jakarta.apache.org/tomcat/index.html>

ConceptMappingRule		
MDOConcept	IRO	IROConcept
Person	Faculty	Student
Person	Faculty	Lecturer
Person	Entrants	Profile
Social-Category	Faculty	Category
...

a) IRDMO structure and the fragment of the knowledge base

SlotMappingRule				
MDOConcept	MDOSlot	IRO	IROConcept	IROSlot
Person	first-name	Faculty	Student	Name
Person	first-name	Entrants	Profile	name
Social-Category	descr-of-Social-Category	Faculty	Category	CategoryName
Department	descrOfDepartment	Entrants	Department	descrOfDepartment
Department	descrOfDepartment	Faculty	Faculty	DescrOfFaculty

Rule			
concept	slot	table	field
Student	belongsToCategory	tblStudents	IDCategory
Student	belongsToGroup	tblStudents	IDGroup
Sheet	resultForStudent	tblSheets	IDStudent
Sheet	MarkAtSheet	tblSheets	Mark
...

b) IRO-IR Schema Mapping Ontology structure and the fragment of the knowledge base

RelationshipMappingRule		
primaryTable	foreignTable	expression
tblCategories	tblStudents	tblCategories.ID=tblCategories.IDCategory
tblGroups	tblStudents	tblGroups.ID=tblStudents.IDGroup
tblStudents	tblSheets	tblStudents.ID=tblSheets.IDStudent
...

Figure 2. The structures of mapping ontologies

Query Formulation Interface is made as separate GUI client desktop application working together with the mediator and operating on Apache Tomcat Web server as well. All the functional components of IEDI architecture (QFI, servers inside the mediator) may be distributed. Communications between functional components are based on SOAP and implemented with help of Apache SOAP processing service⁴. The architectural details of generic IR wrapper server are presented in [11].

Jena⁵ 2 API – Java-based framework for building Semantic Web applications – was chosen to store and query IEDI ontologies. It provides a programmatic environment for RDF, RDFS and OWL, including a rule-based inference engine⁶. Jena is an open-source project and has good support level from developers' team.

⁴ Apache SOAP processing service is available at the URL: <http://ws.apache.org/soap/>

⁵ Jena 2 API is available at the URL: <http://jena.sourceforge.net/>

⁶ Description of Jena features can be found at the URL: <http://simile.mit.edu/reports/stores/index.html>

Internal language for querying IEDI ontologies is RDQL [15], as far as RDQL has implementation in Jena 2 API. The comprehensive survey of all existing query languages for RDF is given in [12].

User interface for query formulation – QFI – was written as Java SWING graphical user interface application.

4 Visual Query Formulation with QFI

In IEDI it is not required that a user codes his or her queries in RDQL notation. Instead, the tool for ontology-driven Query Formulation Interface (QFI) is provided by the IEDI mediator.

The general requirement to QFI was that a user should have a visual ontology-based interface which allows him or her to interactively choose the terms of the MDO, to apply constraints to these terms and, thus, to formulate the query. The user query should be generated automatically from the set of the chosen ontology elements (concepts, properties constrained with the required terms).

The aim of visual query formulation interface in UnIT-Net is twofold.

First, QFI serves as the tool for domain ontologies navigation and learning. This allows user to be aware of the semantics of concepts and questions they will further pose to the mediator.

Second, QFI embeds all the mechanisms for subsequent querying the knowledge stored in mediator/resources. This allows user to pose queries with no extra knowledge on query languages inside the mediator.

Let's walk through the presentation of the following sample query “*Entrants from Zaporozhye, and their schools descriptions*” in QFI (see Fig. 3).

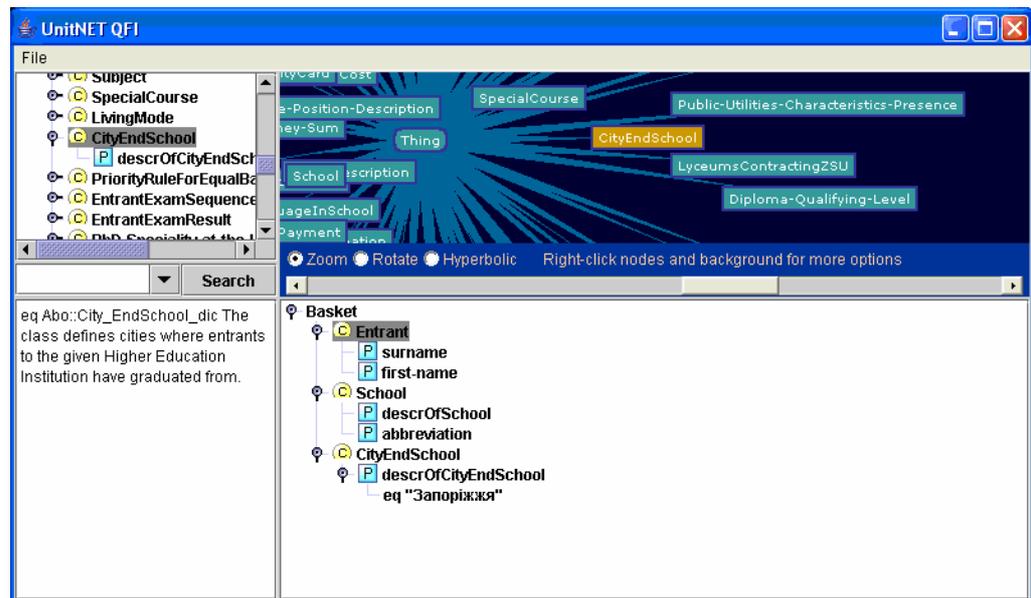


Figure 3. Visual presentation of the query “*Entrants from Zaporozhye and their schools descriptions*”


```

<set><model><concepts><basketclass>
  <cls>
    <className>CityEndSchool</className>
  </cls>
  <children>
    <basketproperty>
      <parent class="basketclass" reference="...">
        <property>
          <propertyName>descrOfCityEndSchool</propertyName>
        </property>
      </basketproperty>
      <constraints>
        <basketconstraint>
          <constraintAction>eq</constraintAction>
          <constraintValue>3anopixxxx</constraintValue>
          <parent class="basketproperty" reference="...">
        </basketconstraint>
      </constraints>
    </basketproperty> </children> </basketclass> ...
  </concepts>
  <ontology>http://unitnet.education.zp.ua/ont/MD0.owl</ontology>
</generatedRDQL>...</generatedRDQL></model></set>

```

Figure 5. Mark-up for Visual Queries basket content

Visual Query (VQ) is constructed by the user with help of visual interface of QFI. We borrow the concept of a “basket” from e-commerce applications to store elements of the visual query. Serializing Java objects into XML is done with help of XStream⁷.

The user adds necessary concepts to the basket, assigns additional restrictions on the values of properties. The fragment of the basket content for query “Entrants from Zaporozhye and descriptions of their schools” is shown in the Fig. 5.

After the query is built visually, QFI generates initial RDQL query. The main problem here comes from the fact that the user is free to choose arbitrary concepts from the ontology. It may happen that the graph constructed over these concepts is unconnected. As it was shown in [9] such graph corresponds to a “bad” initial query, resulting in redundant responses from IRs and rather artificial content of these responses. To avoid this situation QFI makes “bad” queries “good” by automatically searching the paths between concepts and incorporating necessary parts of ontology (concepts, object properties) into the resulting Initial Query (IQ). Dijkstra algorithm⁸ for the search of the shortest path in a graph has been used to implement this intelligent functionality. This algorithm applies to ontology represented as directed graph – concepts are treated as vertexes, object properties – as edges, and tries to find shortest paths between concepts from user query.

QFI allows to make changes directly to the RDQL presentation of the query, and then – to save a new query. This feature is useful in case when a user really needs “bad” query to be executed (e.g. to receive the Cartesian product of instances of several concepts), or when the user is experienced in the domain ontology and is able to describe the path between directly unrelated concepts in RDQL himself.

Fig. 6 shows the RDQL query automatically constructed in QFI for “Entrants from Zaporozhye and their schools descriptions” example. The following triples were automatically added into the resulting query:

⁷ XStream available at URL: <http://www.xml.com/pub/a/2004/08/18/xstream.html>

⁸ Realization of Dijkstra algorithm for shortest path search was taken from JDSL - Data Structures Library in Java, available at the URL: <http://www.cs.brown.edu/cgc/jdsl/>

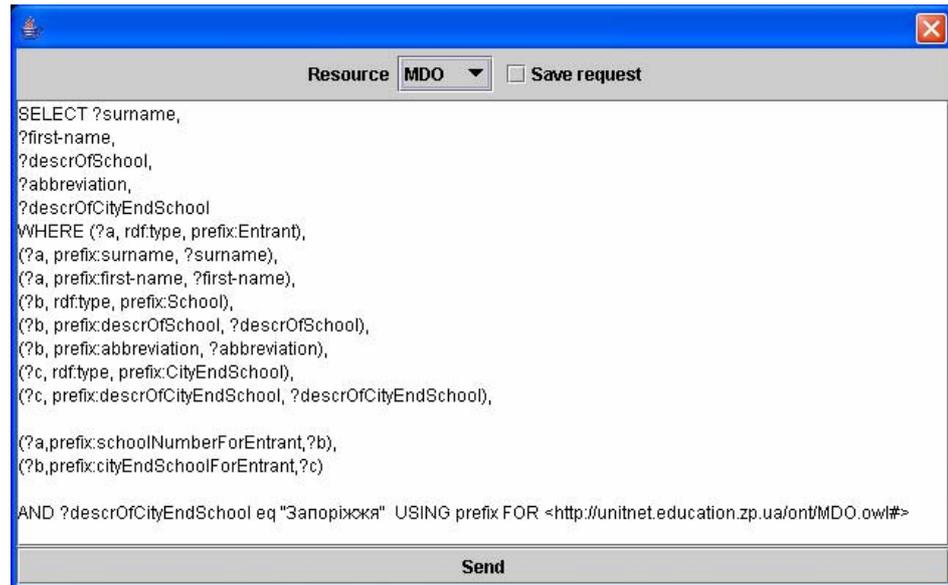


Figure 6. RDQL query generated with QFI

(?a, prefix:schoolNumberForEntrant, ?b) and
 (?b, prefix:cityEndSchoolForEntrant, ?c)

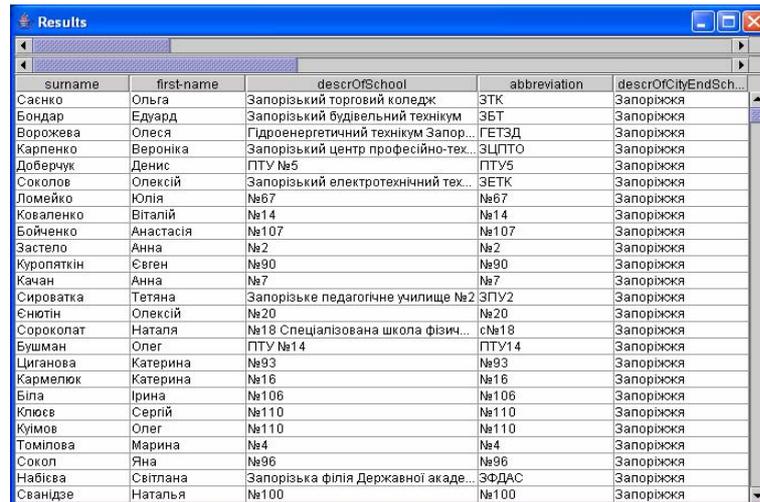
First triple was added to describe the path between concepts “*Entrant*” and “*School*”, and second triple describes the path between “*School*” and “*CityEndSchool*”.

When the initial query is ready, QFI connects to the IEDI mediator, passes this query and waits for the results. Then QFI performs results’ mark-up in terms of the ontology whose concepts were in the query and brings the results to the user.

In addition, QFI stores queries history to allow user to easily pose frequently used queries. Upon the user request QFI may store the particular responses from IRs as well.

As far as initial query is formulated for the whole MDO QFI initiates ontology-driven sub-queries extraction (see detailed description of the ODSQE algorithm in [9]) to obtain sub-queries to the registered IRs. These RDQL sub-queries are partial in the sense that not every MDO concept/property from the initial query should have correspondences in every registered IR. In spite of this, it is guaranteed that each partial RDQL query is correct and produces result. All peculiarities of sub-queries extraction are reported in [9]. Then IEDI mediator transforms each sub-query into the IR query and sends it to the IR. Current implementation of IEDI uses RDQL-SQL transformation algorithm worked out in [11]. IRO concepts and slots are mapped into table names and field names of the underlying relational IR scheme. This mapping according to the IEDI ontologies hierarchy is stored in IRO-IR Scheme Mapping ontology.

IR wrappers accept the sub-queries, initiate their execution and perform results mark-up in terms of respective IRO. The details of IR wrapper algorithms and realization are reported in [11].



surname	first-name	descrOfSchool	abbreviation	descrOfCityEndSch...
Саснко	Ольга	Запорізький торговий коледж	ЗТК	Запоріжжя
Бондар	Едуард	Запорізький будівельний технікум	ЗБТ	Запоріжжя
Ворожева	Олеся	Гідроенергетичний технікум Запор...	ГЕТЗД	Запоріжжя
Карпенко	Вероніка	Запорізький центр професійно-тех...	ЗЦПТО	Запоріжжя
Доберчук	Денис	ПТУ №5	ПТУ5	Запоріжжя
Соколов	Олексій	Запорізький електротехнічний тех...	ЗЕТК	Запоріжжя
Ломейко	Юлія	№67	№67	Запоріжжя
Коваленко	Віталій	№14	№14	Запоріжжя
Бойченко	Анастасія	№107	№107	Запоріжжя
Застело	Анна	№2	№2	Запоріжжя
Куропяткін	Євген	№90	№90	Запоріжжя
Качан	Анна	№7	№7	Запоріжжя
Сироватка	Тетяна	Запорізьке педагогічне училище №2	ЗПУ2	Запоріжжя
Єнютін	Олексій	№20	№20	Запоріжжя
Сороколат	Наталія	№18 Спеціалізована школа фізич...	с№18	Запоріжжя
Бушман	Олег	ПТУ №14	ПТУ14	Запоріжжя
Циганова	Катерина	№93	№93	Запоріжжя
Кармелюк	Катерина	№16	№16	Запоріжжя
Біла	Ірина	№106	№106	Запоріжжя
Клюєв	Сергій	№110	№110	Запоріжжя
Кумов	Олег	№110	№110	Запоріжжя
Томілова	Марина	№4	№4	Запоріжжя
Сокол	Яна	№96	№96	Запоріжжя
Набієва	Світлана	Запорізька філія Державної акаде...	ЗФДАС	Запоріжжя
Свандізе	Наталія	№100	№100	Запоріжжя

Figure 7. Response obtained for the query

E.g. query “Entrants from Zaporozhye and their schools descriptions” had the following response (see Fig. 7).

5 Experiments with QFI

Evaluation experiment was planned as follows. There were chosen two autonomous IRs registered into the IEDI mediator – “University Entrants” IR of Zaporozhye National University and “Dean’s Office” IR of V. Karazin Kharkiv National University. “University Entrants” IR is run on MS SQL Server 2000 platform, “Dean’s Office” IR uses MS Access.

The preparatory stage consisted of construction of experimental query set and installation of IR wrappers on the IRs.

The principle of organizing the query set was that IEDI mediator should correctly answer queries specific to the given IR, and additionally it should be able to answer queries over common concepts of MDO, which have several mappings in IRs.

Each IR support team was asked to prepare a set of queries intrinsic to the IR and formulated in natural language. The size of each query set was approximately 25 queries. As far as frequently asked queries are usually hardwired into the visual interface of information system, most of the queries in the set were taken from that interface. Resulting query sets were formed without any restrictions which might be influenced by the IEDI mediator architecture, ontologies hierarchy and the query formulation language.

Natural language query set over MDO concepts was prepared by MDO support team at ZNU. The main requirement to this set was to intensively use concepts common to both IRs and to explore taxonomy of concepts.

The main stage of the experiment was as follows.

At first, each IR support team has formulated and executed queries from their query sets using QFI and IEDI mediator. The correspondent IR itself was used locally. This has allowed further use the IR query set as a testbed for checking correctness of IRO, IRO-IR Schema Mapping Ontology and of RDQL-SQL transformation algorithm.

At second, MDO support team has re-run IR query sets, when these IR were used remotely. This has proved the reliability of IEDI mediator – wrappers communication through distributed web-services.

At third, MDO support team has formulated in QFI and executed MDO query set to check correctness of sub-queries extraction and IR responses gathering.

For each query sets it was proposed to IRO/MDO support teams to fill in a questionnaire during the main stage of experiments. They were asked to outline the reasons of complexity in query formulation. The results of their answers are shown in Table 1 (several reasons for one query were allowed).

Table 1. Users vision of complexity in query formulation in IEDI

Reason Query sets	IEDI architecture and tasks	RDQL limitations	QFI implementation
IR query sets	0%	36% (18 of 50)	16% (8 of 50)
MDO query set	36% (9 of 25)	36% (9 of 25)	60% (15 of 25)

Both IR support teams were strongly familiar with correspondent domain; however they used QFI with very little respect to its factual capabilities. They expected to get the results in the form similar to their IR native user interface, where the aggregate values (sums, average values etc) are of common use. That's why they've put the restrictions of RDQL on the first place of complexity reasons.

MDO support team was familiar with correspondent domains, and this team was better informed of factual capabilities of QFI and IEDI mediator. Their main task was to pose queries involving concepts from both IRs. They've outlined that IEDI mediator does not merge responses from IRs, but the most complex thing was to compare values from IRs between each other. This has lead to first place of QFI limitations for complexity reasons.

However it should be pointed out that only queries which contained value restriction of non-string data types were failed. The percentage of such queries in the sets was 4% (2 queries of 50).

Maximal delay of response for querying IR situated in the same network with the user and the IEDI mediator was proportional to the response size and did not exceed 5 seconds. For querying IR via remote access both to the mediator and an IR the maximal response delay caused by network connection was 50 seconds.

6 Lessons Learned and Future Research

Experiment with the user interface component of IEDI mediator shows that visual query formulation supplied with native language-oriented interface is very comfortable and easy to use. For novices in the domain it was a bit complex to understand the context of a particular ontology concept, and they were forced to read comments to definitions of concepts and properties. Additional but expected inconvenience was related with the situation when users navigate and query ontology with concepts/properties' names given in a foreign language.

Results obtained during the implementation and testing stages of UnIT-NET project have pointed out current restrictions of the interface between users and IEDI.

Interface restrictions were divided into three types: restrictions imposed with the internal query language – RDQL, simplifications done for the whole IEDI architecture, and simplifications of the current version of QFI.

The first influencing category of restrictions was the internal query language. RDQL does not allow aggregation functions to be applied as at the stage of initial query formulation, as at the stage of results mark-up.

Certain limitations or, better say, simplifications were made to the IEDI architecture. One of them is the simplified structure of IRDMO. Currently the mapping is of the form of equality between concepts and properties from different ontologies. However, practically we were faced with the situation when MDO property may be represented as certain function over IRO properties and concepts. This function may in the primitive case be string concatenation function. E.g. MDO slot “*address*” is atomic and contains street, building number, apartment number and city name, while in one of IRs there is no “*address*”, but separate “*street*”, “*building*”, “*city*”, “*flat*” slots. In more complex case the function should involve the instances of property values. E.g., in one IR the “*gender*” property of a person has the values “*male*” and “*female*”, and in the other IR – correspondent values were “*m*” and “*f*”.

Moreover, as the whole IEDI architecture was not configured for results merging and fusion (it was beyond the scope of UnIT-NET), this has led to the situations when the user had to analyze the responses from the IRs one by one.

QFI limitations were outlined as follows:

- user cannot see object properties of concepts – this leaves the user “outside the kitchen” of initial query formulation
- user unfamiliar with RDQL cannot really approve query prepared by the QFI and he/she is forced to skip the query approval stage
- QFI currently supports value restrictions “equal” and “not equal” for datatype properties. However the whole set of value restrictions provided in the RDQL specification [15] can be easily added to the QFI
- QFI due to limitations of RDQL does not allow aggregation functions to be applied as at the stage of initial query formulation, as at the stage of results mark-up
- QFI currently does not allow to compare one property values with another property values, e.g. in the form “*concept₁.property₁ = concept₂.property₂*”
- QFI does not allow to choose values of property from the list of values stored in the underlying IR
- QFI does not support all data types, which can be in the underlying domain. Currently only string values are allowed in value restrictions.

In spite of the fact that the highest quantity of negative user feedbacks (see Table 1) was obtained because of limitations of QFI, all these limitations are easy to overcome.

The situation with internal query language restrictions is different. At present, RDQL syntax is fixed. One solution is to choose the RDF query language already possessing all necessary features (at least, aggregation). Such languages as SeRQL[12] and SPARQL[16] seem good candidates for the purposes of UnIT-NET IEDI. Another solution for IEDI is to enhance Results Mark-Up Translation Server (see Fig.1). Queries with calculation of maximum/minimum/count/average values etc. may in that

case be performed in two steps: computation of RDQL query result “as is”, without aggregation, and then before presenting the result to the user – to calculate requested aggregation functions (on the mediator side, or at the client side).

Negative user feedbacks concerning the limitations of UnIT-NET IEDI architecture appear when the semantics of a user query requires merging of the IR responses. This problem is also of great importance for the IEDI and is one of future research direction for UnIT-NET consortium.

7 Related Work

Mediator-wrapper architecture [17] provides the syntactic and semantic interoperability [1] with help of intermediate layer (mediator) responsible for unification of user interface to distributed heterogeneous IRs presented with their wrappers.

The number of projects in intelligent information retrieval advocates ontology-based knowledge integration from autonomous heterogeneous distributed information resources. The review of state-of-the-art in this direction one can find in [4], [9], [18] and [19].

Functionality and aesthetic presentation of user interface influences future use of the whole system. As it was mentioned in [14], “...*It is worth noting that most people interacting with computers see only the system interface. Thus, it becomes a very important component of a software system from the design phase onwards.*” There are a lot of investigations focused on visual query system development for databases. The detailed survey and classification of approaches in that field may be found in [14].

Such approaches are rather strict – they assume that user query may consist only of terms already described in the domain.

However, last decade research projects pay more attention to intelligent query interfaces in order to best capture the semantics of a user query. In the variety of proposed techniques outline the following.

User profiles (see a survey in [20]) are often considered as storages of user personal knowledge.

Intensional navigation in SEWASIE ([21], [22]) is used for incremental user query construction through domain ontology navigation. SEWASIE approach allows refinement of existing query terms or even invention of new query terms basing on existing ontology terms and Boolean connectors. Resulting query is then subjected to satisfiability checking w.r.t. domain ontology with help of a reasoning system supporting domain ontology definition language.

Query rewriting techniques [7], [8], [20] apply a set of rules to reformulate user query into terms of underlying resources.

8 Conclusions

The paper reports on the results of evaluation of the UnIT-NET IEDI research prototype. The focus of the paper is the intelligent visual interface – QFI – the graphical tool aiming to assist a user to formulate his/her queries to the resources registered to IEDI. The evaluation of the proposed solutions of IEDI architecture is made experimentally by the members of UnIT-NET consortium. Experiments have shown practical applicability of the approach proposed for the IEDI architecture. One of the stages of the evaluation procedure is the assessment of the quality of interface

between users and IEDI mediator. The paper analyses lessons learned from the evaluation of UnIT-NET IEDI user interface component – Query Formulation Interface. The analysis proves the usability of the QFI; however there are refinements which have still to be considered in the future work.

Acknowledgements

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

REFERENCES

1. Wiederhold G., Genesereth M. The conceptual basis for mediation services. *IEEE Intelligent Systems*, September/October, 1997, 38-47.
2. Sheth A. and Larson J. *Federated Database Systems for Managing Distributed, Heterogenous, and Autonomous Databases*. ACM Computing Surveys, 22:3, 183-236, ACM Press, 1990.
3. 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.
4. 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>
5. Xu Li, Embley D.W. Combining the Best of Global-as-View and Local-as-View for Data Integration. 3-d Intl. Conference on Information Systems Technology and its Applications (ISTA'2004), Salt Lake City, Utah, USA, July 15-17, 2004, 123-136.
6. Chawathe S. et al. The TSIMMIS project: Integration of heterogeneous information sources. Proc. of the 10th Meeting of the Information Processing Society of Japan, Tokyo, Japan, 1994, 7-18.
7. Baader F. et al.: Rewriting concepts using terminologies. 7th Intl. Joint Conf. on Principles of Knowledge Representation and Reasoning (KR 2000), 2000, 297-308.
8. Lattes V.; Rousset M.-C.: The Use of CARIN Language and Algorithms for Information Integration: The PICSEL System. *Intl. J. of Cooperative Information Systems*, 9(4), 2000, 383-401.
9. 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 15-17, 2004, 137-150.
10. OWL Web Ontology Language Reference. W3C Proposed Recommendation, 15 December 2003. URL: <http://www.w3.org/TR/owl-ref/> (last checked: 08.07.2005)
11. Ermolayev. V., Keberle, N., Shapar, V., Vladimirov, V.: Semantically Reinforced Web Services for Wrapping Autonomous Information Resources. *Bulletin of V. Karazin Kharkiv National University*, – 2004. – № 629. Series “Mathematical

- Modelling. Information Technology. Automated Control Systems”, Issue 3. – p.56-69.
12. RDF Query Survey. URL:<http://www.w3.org/2001/11/13-RDF-Query-Rules/> (last checked: 08.07.2005)
 13. Lamping, L.; Rao, R.; and Pirolli, P. A Focus+Context Technique Based on Hyperbolic Geometry for Visualizing Large Hierarchies. Proc. of the ACM SIGCHI Conference on Human Factors in Computing Systems, 401-408. New York: ACM, 1995.
 14. Catarci T., Costabile M.F., Levialdi S., Batini C. Visual Query Systems for Databases: A Survey. Journal of Visual Languages and Computing, 8(2), 1997, 215–260.
 15. RDQL – A Query Language for RDF. W3C Member Submission, 9 January 2004, URL: <http://www.w3.org/Submission/2004/SUBM-RDQL-20040109/> (last checked: 08.07.2005)
 16. SPARQL Query Language for RDF. W3C Working Draft, 19 April 1995. URL: <http://www.w3.org/TR/2005/WD-rdf-sparql-query-20050419/> (last checked: 08.07.2005)
 17. Wiederhold G. Mediators in the Architecture of Future Information Systems. IEEE Computer, 25, 3(March), 1992, 38-49.
 18. Keberle N. Heterogeneous database and knowledge-based integrating systems: the review. Visnyk of the Lviv University. Series Applied mathematics and Computer Science. 2002, No.4, 163-172 (In Ukrainian).
 19. 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.
 20. 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.
 21. Dongilli P., Franconi E., Tessaris S. Semantics Driven Support for Query Formulation. Proc. of the 2004 Intl. Workshop on Description Logics (DL 2004), vol. 104, Whistler, BC, Canada, June 2004.
 22. Catarci T. et al. Usability evaluation tests in the SeWAsIE (SEmantic Webs and AgentS in Integrated Economies) project. Proc. of the 11th Intl. Conf. on Human-Computer Interaction (HCII 2005), 2005.