**УДК 004.8**

# *UNIT-NET IEDI: AN INFRASTRUCTURE FOR ELECTRONIC DATA INTERCHANGE*

**Ermolayev V.A.,**
**Zaporozhye National University,**
**Spivakovsky A.V.,**
**Kherson State University,**
**Zholtkevych G.N.,**
**V. Karazin Kharkiv National University,**
**Bulat A.V.,**
**Kherson State University,**
**Keberle N.G.**
**Zaporozhye National University**

*У статті розглядається еталонна архітектура інфраструктури взаємообліку електронними даними (Infrastructure for Electronic Data Interchange) IEDI – багаторівневе розподілене інтелектуальне програмне забезпечення, яке складається з серверів додатків, сервісів, компонентів та засобів, що забезпечують інтелектуальне подання даних, які керують онтологією з розподілених, гетерогенних ресурсів для організаційної сітки Української системи вищої освіти.*

*The paper presented the reference architecture of Infrastructure for Electronic Data Interchange. IEDI is the multi-layered distributed intelligent software system comprising software servers, services, components and tools providing intelligent ontology-driven information retrieval from distributed, heterogeneous, legally and physically autonomous Informational Resources (Irs) for the organizational network of the Ukrainian National Higher Education System.*

**Introduction.**

To achieve and sustain dynamic improvement, service-oriented organizations like universities, need an infrastructure that underpins flexible and robust management of their activities and decision making support. To a large extent the activities within Universities as well as their coordination and control at national level involve the processing of enterprise data and knowledge. As far as the organizations involved in the Educational framework are rightfully independent, they own and maintain their data and knowledge sources autonomously – i.e. independently from each other and, to a high degree, from the coordination body, like a National Ministry. The fact that these information resources are autonomous implies serious complications for their integration. Developing a framework for intelligent integration of autonomous information resources for Ukrainian Universities was one of the objectives of UnIT-NET project[1]. The project resulted in the development of a research prototype of the Infrastructure for Electronic Data Interchange (IEDI). 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.

[1] UnIT-NET: IT in University Management Network is TEMPUS TACIS multiplier project (2002-2005) funded by ETF, http://www.etf.eu.int/. Its objectives were: creating the Network for disseminating best practices in IT for University management;
establishing the Software Testing Laboratory to support collaborative activities in frame of Unit-Net Network.

The reminder of the paper is structured as follows. Section 2 outlines the related work and the principles used in IEDI architectural design. Section 3 sketches out the architecture of UnIT-NET IEDI. Section 4 focuses on the family of ontologies which drive query decomposition, query translation information retrieval and query results mark-up in IEDI. In the Section 5 a walkthrough exampel is given to understand the usage of UnIT-NET IEDI. Section 6 gives concluding remarks and outlines the directions of the future work.

**2. Related Work.**

In the outlined context the genre of the IEDI falls down to the Distributed Intelligent Information Retrieval (I2R) domain within the broader area of Intelligent Information Integration (I3). The research activities within this domain have been intense in the past decade, especially within the Information Society Technologies Key Action Line of the EU FP6 and similar national and international frameworks. Examples of R&D projects developing the formal, algorithmic, architectural frameworks, deploying software prototypes for I2R from distributed, heterogeneous IR-s and Intelligent Information Integration (I3) are BUSTER [STU00], DOME ([CJO01], [CJO02]), InfoSleuth [BAY97], KRAFT [GRA97], MOMIS [BCD98], OBSERVER [KS00], Ontobroker [DEF99], PICSEL [LR00], SIMS [AKS96], TSIMMIS [GAM95], and others. A good survey of ontology-based approaches to I2R and I3 may be found in [WAC01].

Although all these projects use different techniques, approaches and software paradigms for the task, 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 integration. As outlined in [CJO01], this includes the questions of developing ontologies (bottom-up and top-down approaches), mapping between ontologies, and relationships between ontologies and information resources as data providers.

Most projects adopt one of the following approaches to using ontologies [WAC01]: single ontology (SIMS), multiple ontology (OBSERVER), hybrid approach (BUSTER, DOME). Mapping between ontologies is necessary when the ontologies architecture of the system works with 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 re-use of ontologies. Mappings between ontologies and information resources schemas maintain correspondences between ontology elements and elements of the data schemas. As stated in [CJO01], 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 not from the data schemas. Thus a mapping between ontology elements and data schema elements makes for transparent execution of user queries within the system.

Other reasons for using mappings between ontology elements and data schemas of information resources are the requirements for information resource autonomy and openness of the system as a whole.

The second group concerns the questions of supplying autonomy and dynamic nature of the open system elements. The solutions here advocate one of the mediator architectures: centralized and decentralized. A centralized mediator architecture provides for one centre, which stores all the information about ontologies, information resources, mappings between them, and which controls the query formulation and execution. A known realization of this approach is TSIMMIS. A decentralized mediator architecture provides for each information resource a separate agent/wrapper, which stores mappings between global/shared ontology (-ies) and the underlying information resource (RACING [EKP03]). The resource broker communicates with resource agents/wrappers and determines relevant and accessible resources for every query personally (InfoSleuth, SIMS, KRAFT).

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 for solving these tasks are:

- Use knowledge from ontologies (hypernym/hyponym relationships) to reformulate queries containing terms which do not exist in the ontology(-ies) to construct query plans with no loss of information (OBSERVER).
- Use some rewriting techniques together with mapping techniques to produce queries on information resources that most effectively satisfy the input query (PICSEL).

Some of the problems mentioned have received only partial solution, for example, the problem of semantic interoperability is typically partially solved by committing the participating nodes to a kind of a convention, providing the framework for semantic representations. These partial solutions evidently constrain the application domain and the functionality of the deployed software prototypes for I2R. The constraints for IEDI are as follows:

- IEDI is built on the principles of the mediator-wrapper architecture [WIE92] with the centralized mediator.
- IEDI exploits the hybrid approach [WAC01] for knowledge representation.
- IEDI uses information resource registration to allow the resource to become available for querying.
- IEDI does not provide full automation for ontologies' mapping and alignment.
- IEDI components use rewriting techniques with mappings to produce, process, and perform queries.

The solutions for IEDI were not aimed to broaden the horizons of the current state of the art in I3 or, more specifically, in I2R. The task was to design the software prototype to demonstrate the feasibility of the ontology-driven approach to I2R and, further on in EDI between the Universities and the State Bodies at National level.

**3. IEDI Reference Architecture.**

The main purpose of IEDI is to provide for performing queries over the set of pre-registered, but independent, distributed and semantically heterogeneous IRs. This implies that IEDI is naturally a distributed system. A query may demand to retrieve data from several geographically distributed IRs which belong to different legal owners and are physically stored in different places. This is why IEDI processes are composed of a number of tasks and activities performed at distributed nodes. These tasks should of course be executed in a controlled and ordered way. A process normally involves both automated activities performed by the IEDI software and human activities, like ontology merge and alignment, supplied with appropriate methodologies and software tools. Human activities are performed by various user roles: authorized user (AU), mediator ontology engineer (MOE), IR ontology engineer (IROE), IR provider (please refer to [BGE04] for more details).

An important factor which seriously influenced the design of IEDI architecture is semantic heterogeneity of the IRs which are registered to IEDI mediator. This implied the use of the hierarchy of ontologies which actually drive the performance of distributed queries to different IRs. The tasks of merging and alignment of the ontologies describing the semantics of the IRs and the common ontology of the mediator – Mediator Domain Ontology (MDO) are performed manually. IEDI provides reference ontologies and tools for this ontology engineering activities. However, this thorough preparation work allows to further perform query formulation, sub-query extraction, sub-query execution tasks in a straightforward manner and almost automatically.

The diagram of IEDI query performance scenario is given in Fig. 1. The diagrams for IR Registration and Ontology Coherence Maintenance are omitted here because of space limitations (please refer to [BGE04]).

IEDI Architectural layering is defined according to the analysis of the IEDI processes and tasks and reflects the mediator-wrapper type of IEDI architecture. The layering represents the overall organization of the IEDI and is outlined according to the following points of view:

- What are the Components, the Tools and the User Roles at the specific IEDI layers?
- How do IEDI Clients and Servers interoperate across the layers of its architecture?

IEDI User Layer is the environment for AUs and AU Clients. IEDI IR Wrapper and IR Layers represent autonomous, heterogeneous, and distributed IR holders.

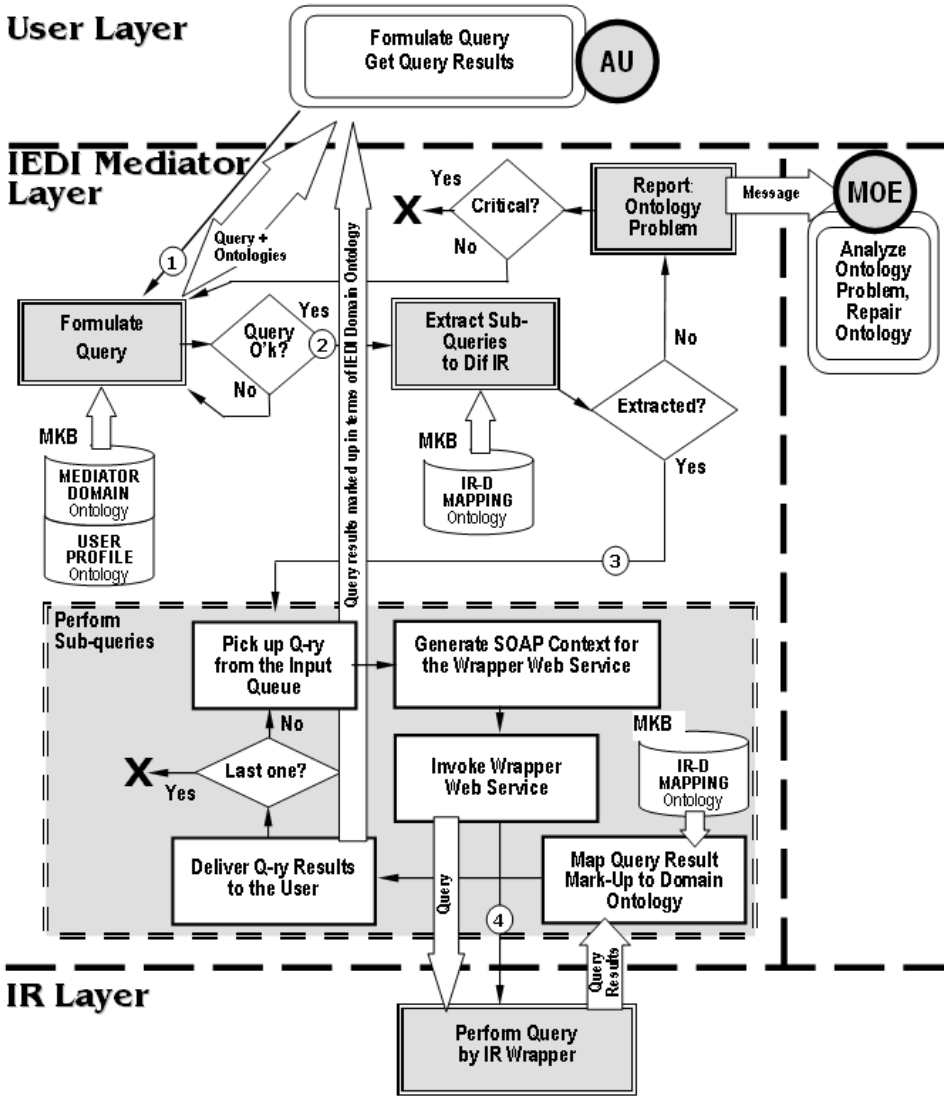



*Figure 1. Process Diagram for IEDI query performance scenario.*

IEDI Mediator Layer is the holder for the components and the tools providing the means for mediation between the AU-s formulating queries and retrieving the results from the registered IR-s and respective IR Wrappers to provide the relevant information. IEDI architectural layering is given in Fig. 2.

The software components of IEDI are split into two categories of Clients and Servers according to their functionality.

IEDI Clients are related to IEDI AU-s and provide the interfaces for their activities. AU client provides IEDI interfaces for an AU. It functions in generic Web Browser environment (+ Java Virtual Machine) at the User Layer of IEDI Architecture and provides the interfaces for the tasks of: User Query Formulation, User Query Approval, Browsing Query Results.

AU Client interoperates with the IEDI Query Formulation Tool [DEKV05] and with the following IEDI components: IEDI Mediator Access Server and Query Formulation Server (the component of IEDI Mediator Server).

MOE Client provides IEDI interfaces for the MOE. It functions in Java Virtual Machine (JVM) environment at the Mediator Layer of IEDI Architecture and provides the interfaces for the tasks of IEDI Ontologies Discussion, Merge, Alignment, Editing and Repair.

IROE Client provides IEDI interfaces for an IROE and is similar to MOE Client.

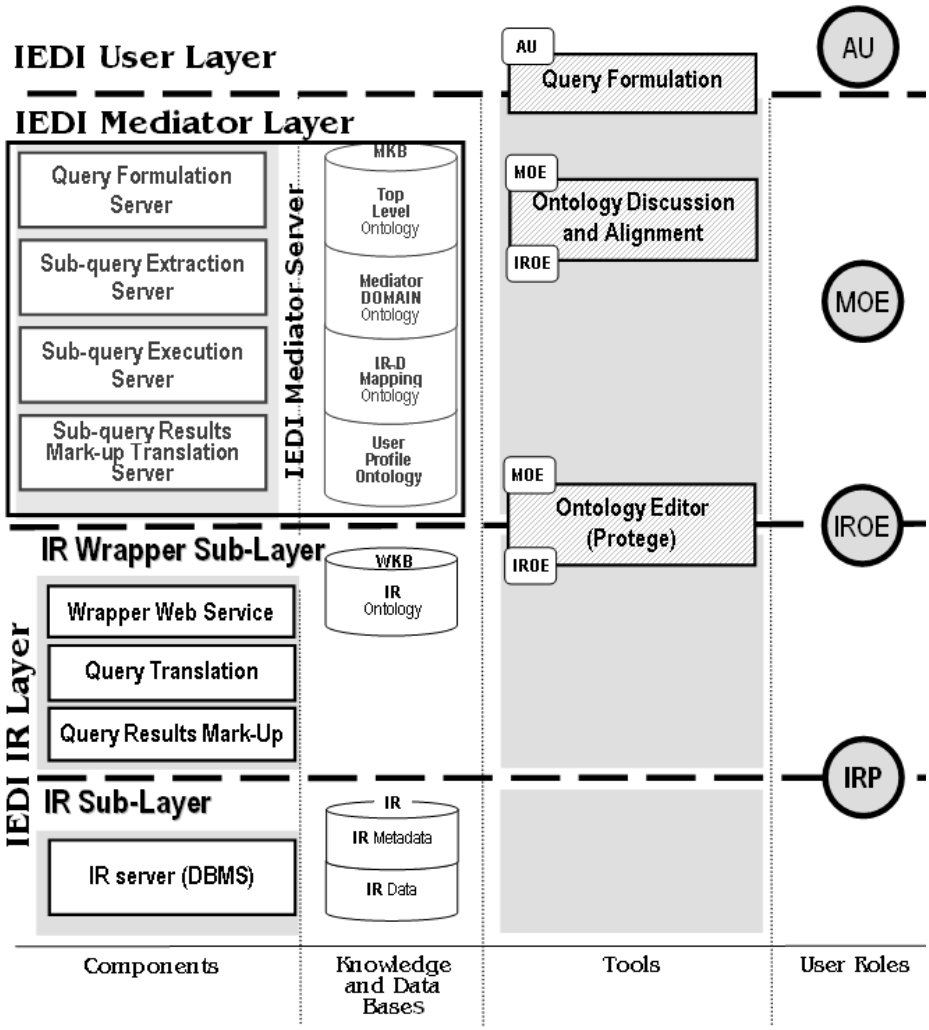



*Figure 2. IEDI Clients and Servers along the layers of the architecture.*

It functions at the Mediator and the IR Wrapper Layers of IEDI Architecture and provides the interfaces for the tasks of IRO Ontology Discussion, Editing and Repair as well as for the Negotiation on IRO – MDO Merge within the IR Registration Process.

MOE and IROE Clients interoperate with the following IEDI tools: Ontology Discussion and Alignment (under development in UnIT-NET), Ontology Editor (Protégé [NSD01]). MOE and IROE Clients interoperate with the following IEDI components: IEDI Mediator Access Server, IEDI Clients and Servers are listed in Fig. 2.

**4 IEDI Ontologies.**

IEDI by its role is a distributed mediator system providing semantic integration of the information retrieved from distributed, heterogeneous, and autonomous information resources. This is why the implementation and the proper usage of semantic descriptions of this information is the critical problem for the overall IEDI system implementation. It is assumed that semantic descriptions within IEDI are formalized and maintained as OWL[2] [OWL03] ontologies at different layers of the architecture. IEDI architecture uses hybrid [WAC01] approach to explicit description of the information resource semantics. Provided are the four types of ontologies: top-level ontology, domain ontology, resource ontology and reference ontology.

**Top-level ontology** (TLO) defines basic top-level elements. These elements according to their definitions are used in the process of mapping resource ontology elements to domain ontology elements. Top-level ontology serves as the foundation for discussion on each concept between MOE and IROE. Top-level ontology allows any two IEDI ontologies to be comparable. IEDI top-level ontology design is based on DOLCE [MBG02].

[2] Protégé 2000 Ontology Editor Ver. 1.9. [NSD01] was used to code All IEDI ontologies.

**Domain ontology** (MDO) represents particular domain knowledge. There are several reasons to explore domain ontology in UNIT-NET IEDI mediator. First one is that domain ontology provides the AUs with the opportunity to formulate their queries using concepts, agreed within domain community and to store correspondences between personal user knowledge on the domain and agreed domain ontology in their user profiles (User Profiles Reference Ontology – UPRO). Another reason is that domain ontology presents a vision of the community on the domain, and therefore plays an educational role.

**Information Resource ontology** (IRO) is a kind of domain ontology, which is constructed at the resource side independently of other resources as well as from mediator ontologies. It presents the vision of IROE on the domain. Resource ontology is used in the process of resource registration at the mediator. Each registered information resource should have its own resource ontology.

**Reference ontologies** (IR – Domain Mapping Ontology (IRDMO), UPRO) are mediator ontologies, which store the knowledge on correspondences between concepts in two or more ontologies. IRDMO contains axioms denoting equivalence/subsumption between concepts/slots. The function of the UPRO is to represent the semantics of AU profiles (refer to [EKP03] for more details).

Table 1 summarizes the involvement of IEDI mediator ontologies in IEDI processes.

Table 1.

**Use of Ontologies in IEDI processes**

| Ontologies<br>Processes | TLO | MDO Core | MDO | IRDMO | IRO | UPRO |
|---|---|---|---|---|---|---|
| Query distributed autonomous semantically heterogeneous IRs | -- | R | R | R | R | R/U |
| Register new information resource | R | R | R/U | R/U | R | -- |
| Maintain coherence in semantic descriptions | R | R/U | R/U | R/U | R/U | R/U |

Legend: R – used for reference purposes only, R/U – used as a reference and is updated, -- – not used.

**5 A Walkthrough Example.**

Let's demonstrate how IEDI functionality scenarios may be utilized in a practical application to the University Management Domain. Consider a typical query, the results of which might be useful, for example, in the assessment of the quality of the secondary education in the region, or at the National level:

***Retrieve the list of the 1-st year students who had received maximal grade (5) in Mathematics at the entrance examinations and have failed to pass the 1-st Term examination in any basic course in Mathematics (got unsatisfactory grade – 2).***

Suppose, an IEDI grants access to two IR-s: The University Entrant IR based on the MS SQL DBMS and a University Faculty Students IR implemented as MS Access application. The relevant to the example query fragments of the DataBase Schemas for these IR-s are given in Fig. 3 and Fig. 4.

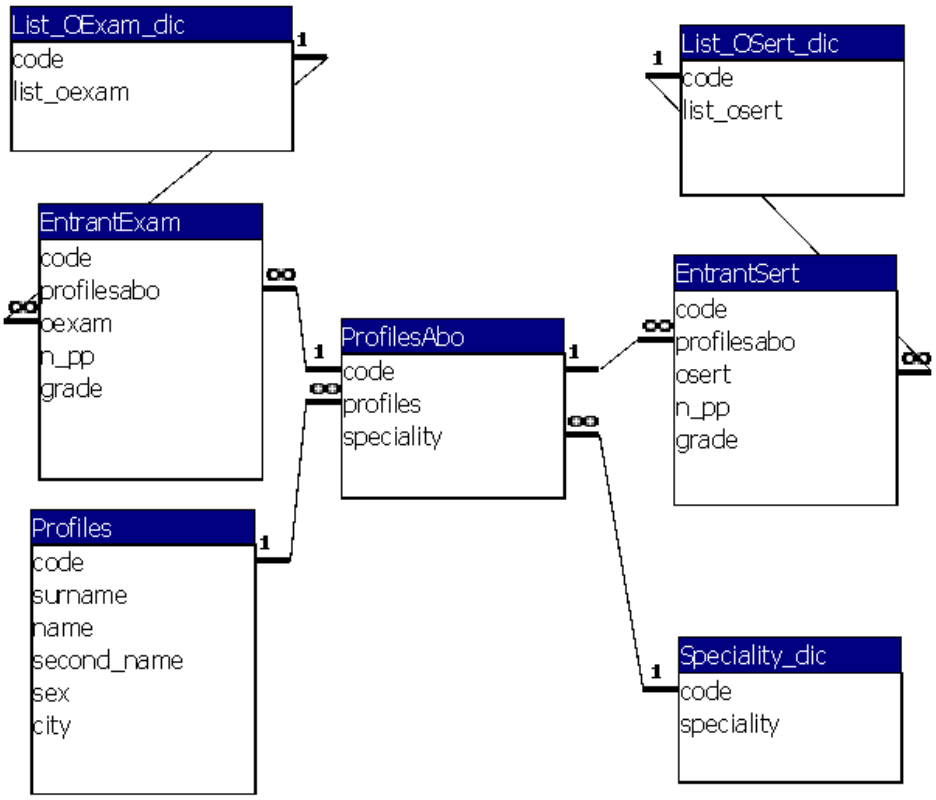



*Fig. 3. The fragment of the DB Schema of the University Entrant IR.*

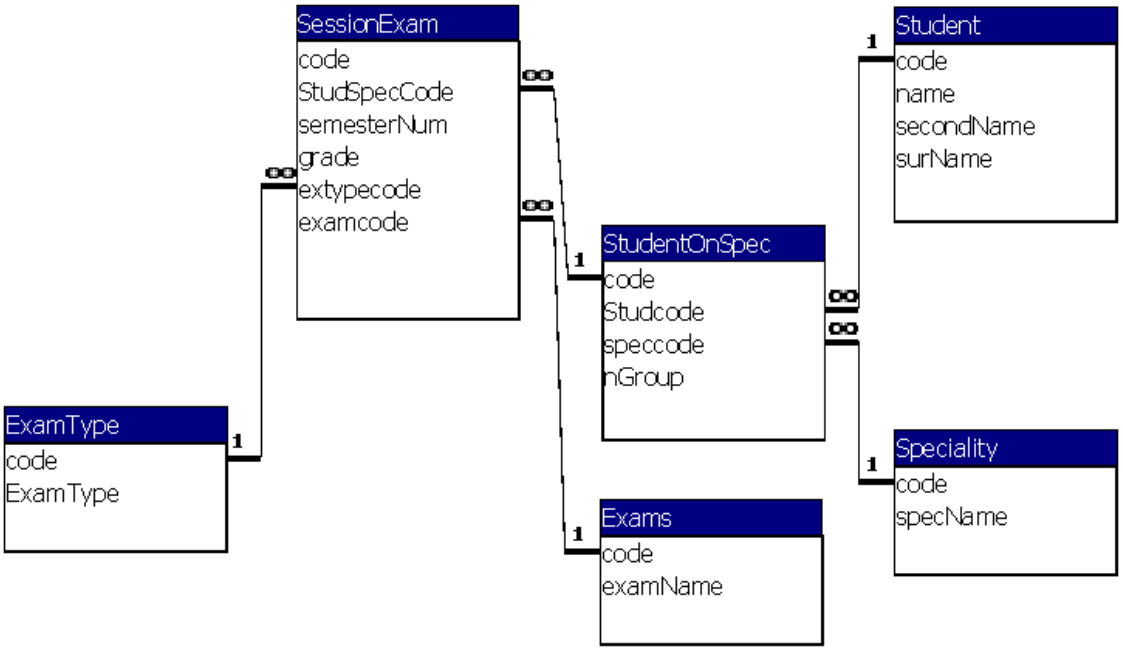
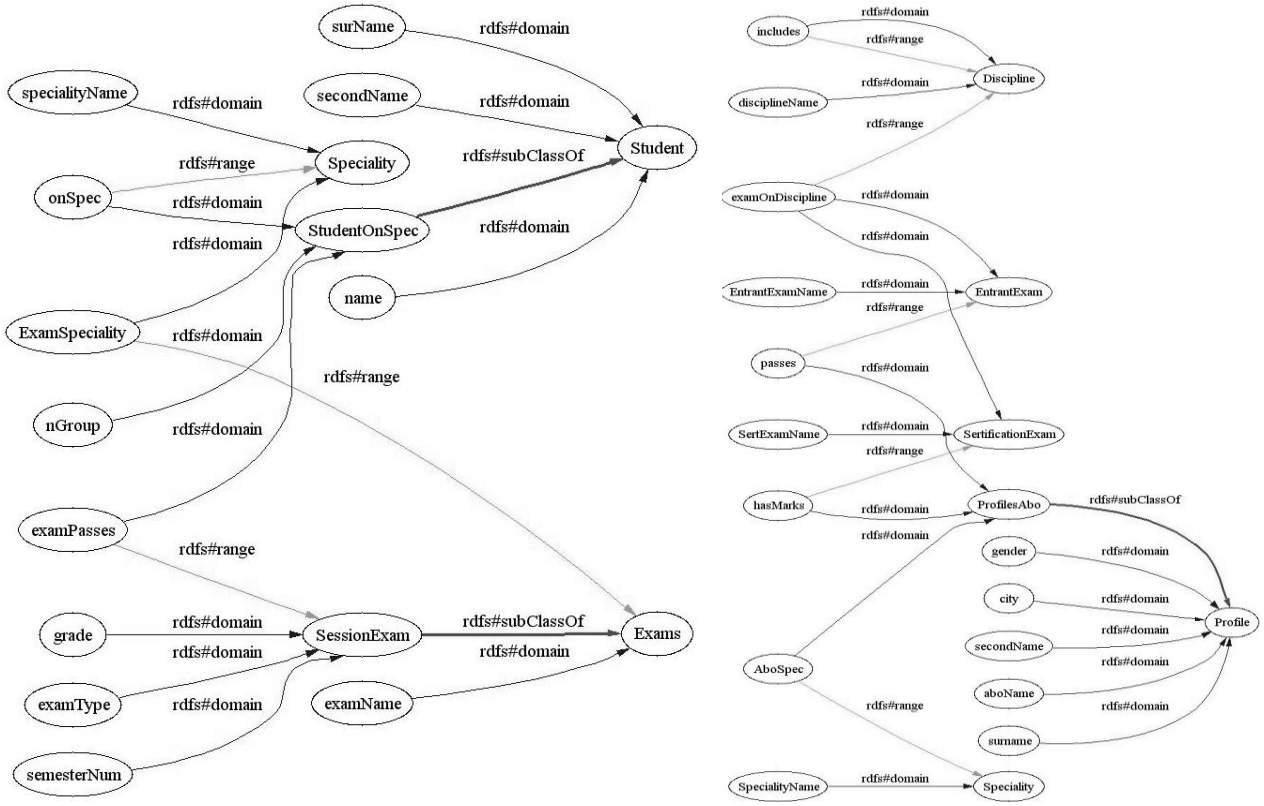



*Fig. 4. The fragment of the DB Schema of the University Faculty Students IR.*

**5.1. IR Registration.**

It is assumed in IEDI that an IR should first be registered before becoming available to AU queries. The registration process comprises the design and the deployment of the IRO and further merge of the IRO to the IEDI MDO. The graphical representation for the example IRO fragments is shown on Fig. 5.




a) IRO for the Univ. Faculty Students IR b) IRO for Univ. Entrant IR

*Fig. 5. Graphical representation of the fragments of the IRO-s for the example IR-s.*

As it was described in Section 2., the registration of the IR to IEDI comprises the merge of the respective IRO with the MDO. This merge is performed collaboratively by the MOE and the IROE and affects MDO and IRDMO. The changes in MDO are done manually and reflect the consensus obtained by the ontology engineers on the relationships between the elements of the MDO and the IRO. These relationships are automatically stored to IRDMO in the form of mappings. These mappings are further on used for transforming AU queries.

For our example the first registered IR was University Entrant. During the registration the ontology engineers have uploaded the University Entrant IRO to the empty MDO and have agreed that the following ontology elements should be renamed to more adequately reflect the semantics of the domain:

- IRO: <Profile> to MDO: <Person>
- IRO: <AboSpec> to MDO: <PersonOnSpeciality>

In addition it was agreed that <EntrantExam> and <SertificationExam> concepts of IRO should be the subclasses of the <Exam> concept. <Exam> concept also received its datatype property <exam_title>.

When the IRO for University Faculty Students IR was registered the ontology engineers have agreed that the following concepts of the IRO and the MDO were semantically equivalent:

- IRO:<Student> and MDO:<Person>
- The properties of IRO:<Student> (namely <surName>, <secondName>, <Name>) do not bring new semantics to MDO. It was agreed that they are semantically equivalent to (<last_name>, <second_name>, <first_name> respectively).

- The slots for IRO:<Student> were not analyzed as far as the concept does not induce any relationships to another concepts.
- IRO:<Speciality> and MDO:<Speciality>

The properties of IRO:<Speciality> are:

- <specialityName> – datatype property semantically equivalent to MDO: <spec_name>
- <onSpec> – object property (defined for <StudentOnSpec> concept and has values of <Speciality>) semantically equivalent to MDO:<on_spec>
- <ExamSpeciality> – object property (defined for <Speciality> concept and has values of <Exams>)

<ExamSpeciality> property has not been defined in MDO before. It was agreed to add <exam_spec> object property (defined for MDO:<Speciality> concept and has values of MDO:<Exam>). Please note that by this time we have not analyzed IRO:<Exams> concept – so the addition of <exam_spec> property has not been properly finalized – was marked as the intended property addition.

- IRO:<StudentOnSpec> and MDO:<PersonOnSpec> – have the same meaning.
- MDO:<on_spec> object property addition has been finalized.
- The new properties provided by IRO:<StudentOnSpec> are <nGroup> and <examPasses> – added to MDO, <examPasses> has been marked as the intended property addition (object property defined on <StudentOnSpec> having values of <SessionExam>. <SessionExam> has not yet been analyzed).
- IRO:<Exams> and MDO:<Exam> – have the same meaning, as well as their property IRO:<examName> and MDO: <exam_title>.

Addition of <exam_spec> Object Property has been finalized.

Ontology engineers have also agreed that IRO:<SessionExam> is the new concept and should be added to MDO together with its datatype property IRO:<semesterNum>. IRO:<examType> datatype property has been moved to MDO:<Exam> concept and has gained broader meaning (not only a type of a session examination, but also an entrance exam and a school graduate certification exam). IRO: <grade> object property has been transformed to MDO: <grade> property. MDO: <grade> received its sub-properties:

- MDO:<session_grade> (an object property for MDO:<SessionExam> concept, the sub-class of MDO:<Exam>)
- MDO:<certification_grade> (an object property for MDO:<CertificationExam> concept, the sub-class of MDO:<Exam>)
- MDO: <entrant_grade> (an object property for MDO:<EntrantExam> concept, the sub-class of MDO:<Exam>)

Resulting IRO-MDO mappings are presented on Tables 2a and 2b.

Table 2a

**IRO – MDO Concept Mapping (example)**

| **Concept Mapping** | | |
|---|---|---|
| **MDO Concept** | **IRO Concept** | **Resource Name** |
| **1** | **2** | **3** |
| Discipline | Discipline | Entrant |
| Person | Profile | Entrant |
| Person | Student | Faculty |
| Exam | EntrantExam | Entrant |
| Exam | SertificationExam | Entrant |
| Exam | SessionExam | Faculty |
| Exam | Exams | Faculty |
| EntrantExam | EntrantExam | Entrant |

| 1 | 2 | 3 |
|---|---|---|
| SertificationExam | SertificationExam | Entrant |
| SessionExam | SessionExam | Faculty |
| Speciality | Speciality | Entrant |
| Speciality | Speciality | Faculty |
| PersonOnSpeciality | AboSpeciality | Entrant |
| PersonOnSpeciality | StudentOnSpec | Faculty |

Table 2b

**IRO – MDO Slot Mapping (example)**

| IRO – MDO Slot Mapping | | | | |
|---|---|---|---|---|
| **MDO Concept** | **MDO Slot** | **IRO Concept** | **IRO Slot** | **Resource Name** |
| Person | city | Profile | city | Entrant |
| Person | gender | Profile | Sex | Entrant |
| Person | first_name | Profile | aboName | Entrant |
| Person | last_name | Profile | surname | Entrant |
| Person | second_name | Profile | secondName | Entrant |
| Person | first_name | Student | name | Faculty |
| Person | last_name | Student | surName | Faculty |
| Person | second_name | Student | secondName | Faculty |
| Exam | examOnDiscipline | EntrantExam | examOn Discipline | Entrant |
| Exam | examOnDiscipline | Sertification Exam | examOn Discipline | Entrant |
| Exam | exam_title | EntrantExam | EntrantExam Name | Entrant |
| Exam | exam_title | Sertification Exam | SertExamName | Entrant |
| Exam | exam_title | Exams | examName | Faculty |
| Exam | exam_type | SessionExam | examType | Faculty |
| Discipline | disciplineName | Discipline | disciplineName | Entrant |
| Discipline | includes | Discipline | includes | Entrant |
| EntrantExam | entrant_grade | EntrantExam | grade | Entrant |
| Sertification Exam | sertification_grade | Sertification Exam | grade | Entrant |
| SessionExam | session_grade | SessionExam | grade | Faculty |
| SessionExam | semester_num | SessionExam | semesterNum | Faculty |
| PersonOn Speciality | n_group | StudentOnSpec | nGroup | Faculty |
| PersonOn Speciality | exams_passes | ProfilesAbo | passes | Entrant |
| PersonOn Speciality | exams_passes | ProfilesAbo | hasMarks | Entrant |
| PersonOn Speciality | exams_passes | StudentOnSpec | examPasses | Faculty |
| PersonOn Speciality | on_spec | ProfilesAbo | AboSpec | Entrant |
| PersonOn Speciality | on_spec | StudentOnSpec | onSpec | Faculty |
| Speciality | spec_name | Speciality | SpecialityName | Entrant |
| Speciality | spec_name | Speciality | specialityName | Faculty |
| Speciality | exam_spec | Speciality | ExamSpeciality | Faculty |

**5.2. Querying IEDI.**

The process of posing a query to IEDI comprises several stages:

- Query formulation
- Sub-query extraction
- Sub-query performance

Query formulation stage is performed by an AU with the help of the IEDI Query Formulation Tool [DEKV05]. Visual graphical interface of this tool, called QFI – Query Formulation Interface, gives user a possibility to browse the ontologies, known to the mediator, namely – MDO, IRO-s, and to formulate a query by choosing the necessary ontology elements and by applying the necessary constraints to them. Details on QFI implementation and evaluation are discussed in [DEKV05]. The tool than generates the query in the notation of IEDI query formulation language. In frame of UnIT-NET IEDI we have chosen RDQL [RDQL04] query language, allowing automated construction of an RDF graph, having the structure defined in the query.

Back to our example, RDQL query for:

***Retrieve the list of the 1-st year students who had received maximal grade (5) in Mathematics at the entrance examinations and have failed to pass the 1-st Term examination in any basic course in Mathematics (got unsatisfactory grade – 2)***

Is generated as follows:

SELECT ?firstName, ?secondName, ?lastName, ?specialityName,
    ?sessionExTitle
WHERE
    (?*x*, stud:first_name, ?*firstName*),
    (?*x*, stud:second_name, ?*secondName*),
    (?*x*, stud:last_name, ?*lastName*),
    (?*x*, stud:exams_passes, ?y),
    (?*y*, stud:exam_title,?*entrantExTitle*),
    (?*y*, stud:exam_type, ?*examType1*),
    (?*y*, stud:entrant_grade, ?*entrantGrade*),
    (?x, stud:exams_passes, ?z),
    (?*z*, stud:exam_title,?*sessionExTitle*),
    (?*z*, stud:exam_type, ?*examType2*),
    (?*z*, stud:session_grade, ?*sessionGrade*),
    (?*y*, stud:semesterNum,?*semesterNum*),
    (?x, stud:on_spec, ?*a*),
    (?a, stud:specialityName, ?*specialityName*)
    (?y, stud:examOnDiscipline,?*r1*),
    (?*r1*,stud:disciplineName,?*entrDiscName*),
    (?*z*, stud:examOnDiscipline,?*r2*),
    (?*r2*,stud:disciplineName,?*sessionDiscName*),
    (?*r1*,stud:includes, ?*i1*),
    (?*i1*,stud:disciplineName,?*discName1*),
    (?*r2*,stud:includes, ?*i2*),
    (?*i2*,stud:disciplineName,?*discName2*)
AND (?*examType1* eq “Exam”), (?*examType2* eq “Exam”)
AND (?*entrDiscName* eq “**Mathematics**”),
    (?*sessionDiscName* eq “**Mathematics**”)
AND ((?entrantExTitle eq ? discName1)
    (?sessionExTitle eq ?discName2))
AND ((?*sessionExTitle* eq “**Linear Algebra**”) ||
    (?*sessionExTitle* eq “**Mathematical Analysis**”))
AND (?*entrantGrade* eq “**5**”), (?*sessionGrade* eq “**2**”)

AND (?*semesterNum* eq "**1**")
USING stud FOR <MDO-URL#>

After the query is approved by the AU it goes through the Sub-query extraction procedure. The task is to extract the sub-queries to different IR-s. The extraction is guided by the knowledge provided by the IRDMO. The details of qub-query extraction and query harmonization are described in [EKSV04b].

For our example the sub-queries are as follows (RDQL):

- For University Entrant IR (RDQL):

SELECT ?aboName,?secondName,?surName,?specialityName
WHERE
(?x, abo:aboName, ?aboName),
(?x, abo:secondName, ?secondName),
(?x, abo:surname, ?surName),
(?x, abo:passes,?q),
(?q, abo:EntrantExamName, ?entrantExamName),
(?q, abo:grade,?entrantGrade),
(?x, abo:AboSpec, ?a),
(?a, abo:SpecialityName, ?specialityName),
(?q, abo:examOnDiscipline,?r),
(?r, abo:disciplineName,?discName1),
(?r, abo:includes, ?i),
(?i, abo:disciplineName,?discName2)
AND (?discName1 eq "Mathematics")
AND (?discName2 eq ?entrantExamName)
AND (?entrantGrade eq "5"))
USING abo FOR <Univ. Entrant IRO URL#>
What means in the natural language:

***Retrieve the list of the university entrants who had received maximal grade (5) in Mathematics at the entrance examinations.***

- For Faculty Student IR (RDQL):

SELECT ?name,?secondName,?surName,?specialityName,
?examName
WHERE
(?x, stud:name, ?name),
(?x, stud:secondName, ?secondName),
(?x, stud:surName, ?surName),
(?x, stud:examPasses, ?y),
(?y, stud:examName, ?examName),
(?y, stud:grade, ?s),
(?y, stud:semesterNum,?q),
(?x, stud:onSpec, ?a),
(?a, stud:specialityName, ?specialityName)
AND ((?*examName* eq "**Linear Algebra**")
(?*examName* eq "**Mathematical Analysis**"))
AND (?*s* eq "**2**")
AND (?*q* eq "**1**")
USING stud FOR <Faculty Student IRO URL#>
What means in the natural language:

***Retrieve the list of the 1-st year students who have failed to pass the 1-st Term examination in any basic course in Mathematics (got unsatisfactory grade – 2).***

After the sub-queries are extracted they are put through to the respective resource wrappers for the execution. The wrappers are invoked via their semantically reinforced web services as described in [EKSV04a].

Wrapper web service invocation context is specified in SOAP [SOAP03]. For example, SOAP envelope for the Faculty Student IR query looks like:

```
POST /IRWrapperQuery HTTP/1.1
Host: <IR Wrapper Server URI>
Content-Type: text/xml; charset="utf-8"
<SOAP-ENV:Envelope
    xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
      SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
        <SOAP-ENV:Body>
            <m:IRWrapperQuery xmlns:m="http://<Faculty Student IR Wrapper Server URI>">
            <req xsi:type="SOAP-ENC:base64">
                        <base64 encoded query goes here as the array of characters>
            </req>
            </m:IRWrapperQuery>
        </SOAP-ENV:Body>
      </SOAP-ENV:Envelope>
```

An IR wrapper obtains SOAP message with RDQL query already in terms of that IRO and performs query translation to the query language that IR supports, e.g. to SQL, or Xquery. The algorithm of query translation is given in details in [EKSV04a]. For our example, the result of the translation is as follows:

| Query in the terms of IRO | MS SQL query in the terms of IR Schema |
|---|---|
| SELECT ?*name*, ?*secondName*, ?*surName*, ?*specialityName*, ?*examName*<br>WHERE<br>(?*x*, stud:name, ?*name*),<br>(?*x*, stud:secondName, ?*secondName*),<br>(?*x*, stud:surName, ?*surName*),<br>(?*x*, stud:examPasses, ?*y*),<br>(?*y*, stud:examName, ?*examName*),<br>(?*y*, stud:grade, ?*s*),<br>(?*y*, stud:semesterNum,?*q*),<br>(?*x*, stud:onSpec, ?*a*),<br>(?*a*, stud:specialityName, ?*specialityName*)<br>AND ((?*examName* eq "**Linear Algebra**")<br>\|\| (?*examName* eq "**Mathematical Analysis**"))<br>AND (?*s* eq "**2**")<br>AND (?*q* eq "**1**")<br>USING stud<br>FOR <Faculty Student IRO URL#> | SELECT<br>Student.name, Student.secondName, Student.surName, Speciality.specialityName, Exams.examName<br>FROM<br>StudentOnSpec, ExamType,<br>Exams, SessionExam,<br>Student, Speciality<br>WHERE<br>SessionExam.grade='2' AND<br>SessionExam.semesterNum=1 AND<br>ExamType.ExamType='Exam' AND<br>(<br>Exams.examName='Linear Algebra' OR<br>Exams.examName='Mathematical Analyses' ) AND<br>Exams.code = SessionExam.examcode AND<br>Student.code = StudentOnSpec.Studcode AND<br>ExamType.code = SessionExam.extypecode AND<br>StudentOnSpec.code =<br>SessionExam.StudSpecCode AND StudentOnSpec.speccode = Speciality.code; |

The query is than executed by the IR Server. The results of the query execution are delivered as plain tabulated text:

| **surname** | **name** | **second_name** | **speciality** | **examName** |
|---|---|---|---|---|
| ОВЕРКО | НАТАЛІЯ | ОЛЕКСАНДРІВНА | Прикладна математика | Mathematical Analyses |
| КОНОНОВА | ІРИНА | ВІКТОРІВНА | Математика | Linear Algebra |
| УРСУЛОВА | НІНА | ВІТАЛІЇВНА | Фінанси | Linear Algebra |
| … | … | … | … | … |
| ГАВРИЛЮК | ЮЛІЯ | СЕРГІЇВНА | Фінанси | Linear Algebra |

This result is than marked-up in the terms of the IRO:

```
<row>
        <surname>ОВЕРКО</surname>
        <name>НАТАЛІЯ</name>
        <second_name>ОЛЕКСАНДРІВНА</second_name>
        <speciality>Прикладна математика</speciality>
        <exam_spec>Mathematical Analyses</exam_spec>
</row>…
<row>
        <surname>ГАВРИЛЮК</surname>
        <name>ЮЛІЯ</name>
        <second_name>СЕРГІЇВНА</second_name>
        <speciality>Фінанси</speciality>
        <examName>Linear Algebra</examName>
</row>
```

and returned back to the mediator as the result provided by the web service. Web service response is enveloped in SOAP:

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset="utf-8"
Content-Length: <the length of the response>

<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
    <SOAP-ENV:Body>
      <m:IRWrapperQueryResponse xmlns:m="http://<IR Wrapper Server URI>">
        <res xsi:type="SOAP-ENC:base64">
------ Query result goes here ----------------------
PHJvdz4KCTxzdXJOYW1lPs7CxdDKzjwvc3V
…
L1NwZWNpYWxpdHlOYW1lPgoJPGV4YW1OYW1lPsL78fjg/yDs4PLl7ODy6OrgPC9leGFtTmFt
ZT4KPC9yb3c+Cg==
------ Query result goes here ----------------------
</res>
      </m:IRWrapperQueryResponse>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

In turn, the mediator component translates the result mark-up to the terms of the MDO:

```
<row>
        <last_name>ОВЕРКО</last_name>
        <first_name>НАТАЛІЯ</first_name>
        <second_name>ОЛЕКСАНДРІВНА</second_name>
        <spec_name>Прикладна математика</spec_name>
```

```
        <exam_spec>Mathematical Analyses</exam_spec>
</row>…
<row>
        <last_name>ГАВРИЛЮК</last_name>
        <first_name>ЮЛІЯ</first_name>
        <second_name>СЕРГІЇВНА</second_name>
        <spec_name>Фінанси</spec_name>
        <exam_spec>Mathematical Analyses</exam_spec>
</row>
```

This result is conveyed to the AU as one of the parts of the results for the query.

Another result of the sub-query for our example is the one received from the University Entrant IR Wrapper.

In the form of the plain tabulated text it looks like:

| **surname** | **name** | **second_name** | **speciality** |
|---|---|---|---|
| ОВЕРКО | НАТАЛІЯ | ОЛЕКСАНДРІВНА | Прикладна математика |
| КОНОНОВА | ІРИНА | ВІКТОРІВНА | Математика |
| УРСУЛОВА | НІНА | ВІТАЛІЇВНА | Фінанси |
| … | … | … | … |
| ГАВРИЛЮК | ЮЛІЯ | СЕРГІЇВНА | Фінанси |

After mark-up translation it looks like:

```
<row>
        <last_name>ОВЕРКО</last_name>
        <first_name>НАТАЛІЯ</first_name>
        <second_name>ОЛЕКСАНДРІВНА</second_name>
        <spec_name>Прикладна математика</spec_name>
</row>
…
<row>
        <last_name>ГАВРИЛЮК</last_name>
        <first_name>ЮЛІЯ</first_name>
        <second_name>СЕРГІЇВНА</second_name>
        <spec_name>Фінанси</spec_name>
</row>
```

**4. Concluding Remarks.**

The paper presented the reference architecture of IEDI. IEDI is the multi-layered distributed intelligent software system comprising software servers, services, components and tools providing intelligent ontology-driven information retrieval from distributed, heterogeneous, legally and physically autonomous IRs for the organizational network of the Ukrainian National Higher Education System. This architecture is built using the following principles:

- It is a mediator-wrapper architecture with the centralized mediator.
- It exploits the hybrid approach for knowledge representation.
- It uses information resource registration to allow the resource to become available for querying.
- IEDI combines processes performed both automatically (ontology driven distributed query processing) and manually (ontology discussion, merging, mapping, and alignment during IR Registration and Ontology Coherence Maintenance).
- Its components use rewriting techniques with mappings to produce, process, and perform queries.

From the semantic point of view IEDI exploits the hierarchy of ontologies which are replenished incrementally when new IRs are registered to the mediator. These ontologies are domain theories and partial mapping specifications to be used as knowledge specifications in software components implementing the functionality of the mediator: to assist in query formulation, to decompose the query into the set of sub-queries (one per relevant IR), to convey the sub-queries

to the respective IR wrappers, to translate the sub-query at the IR wrapper level, to mark-up query results.

## *BIBLIOGRAPHY*


1. [AKS96] Arens, Y., Knoblock, C.A., Shen, W.: Query Reformulation for Dynamic Information Integration. Journal of Intelligent Information Systems, 1996.
2. [BAY97] 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. [BCD98] Bergamaschi, S., Castano, S., De Capitani di Vimercati, S., Montanari, S. Vincini, M.: An Intelligent Approach to Information Integration. In: Proc. Of Formal Ontology in Information Systems (FOIS-98), June, 1998.
4. [BGE04] Bulat, A., Ermolayev, V., Gray, E., Keberle, N., Plaksin, S., Shapar, V., Vladimirov, V., Zholtkevich, G.: 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/
5. [CJO01] Cui, Z., Jones, D., O'Brien, P.: Issues in Ontology-based Information Integration. 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, pp.141-146.
6. [CJO02] Cui, Z., Jones, D., O'Brien, P.: Semantic B2B Integration: Issues in Ontology-based Applications. SIGMOD Record, Vol.31, No.1, March 2002. Pp.43-48
7. [DEF99] Decker, S., Erdmann, M., Fensel, D., Studer, R.: 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.
8. [EKP03] Ermolayev, V., Keberle, N., Plaksin, S., Vladimirov, V. (2003) 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, pp. 9-20, LNI Series, ISBN 3-88579-359-8
9. [GAM95] 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.
10. [GRA97] Gray. P., Preece A., Fiddian N., Gray W., Bench-Capon T., Shave M., Azarmi N., Wiegand M.: KRAFT: Knowledge Fusion From Distributed Databases and Knowledge Bases. In: Proc. 8th Intl. Workshop on Database and Expert System Applications (DEXA-97), IEEE Press, pp. 682-691.
11. [KS00] Kashyap V., Sheth A.: Information Brokering across Heterogeneous Digital Data: A Metadata-based Approach. Kluwer Academic Publishers, 2000
12. [LR00] Lattes V., Rousset M.-C.: The Use of CARIN Language and Algorithms for Information Integration: The PICSEL System. International Journal of Cooperative Information Systems, Vol.9, No.4, 2000, pp.383-401.
13. [MBG02] Masolo, C., Borgo, S., Gangemi A., Guarino, N., Oltramari, A., Schneider, L.: WonderWeb Project Deliverable D17. The WonderWeb Library of Foundational Ontologies and the DOLCE ontology. URL: http://www.compscipreprints.com/comp/Preprint/stborgo/20021127/2/WondeWebD17v2.0.pdf
14. [NSD01] F.-Noy, N., Sintek, M., Decker, S., Crubezy, M., Fergerson, R.W., Musen, M.A.: Creating Semantic Web Contents with Protege-2000. IEEE Intelligent Systems 16(2):60-71, 2001.
15. [OWL03] OWL Web Ontology Language Reference. W3C Recommendation 10 February 2004. URL: http://www.w3.org/TR/owl-ref/
16. [STU00] Stuckenschmidt H., Wache H., Voegele T., Visser U.: 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, pp. 35-46.
17. [WAC01] Wache, H., Vogele, T., Visser, U., Stuckenschmidt, H., Schuster, G., Neumann, H., Hubner, S.: 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, pp.108-118, URL: http://www.cs.vu.nl/~heiner/public/ois-2001.pdf

18. [WIE92] Wiederhold, G.: Mediators in the Architecture of Future Information Systems. IEEE Computer, 25, 3 (March), 1992, 38–49.
19. [EKSV04a] Ermolayev, V.A., Keberle N.G., Shapar V.V., Vladimirov V.N. Semantically Reinforced Web Services for Wrapping Autonomous Information Resources.: *Herald of Kharkiv Karazin National University, series "Mathematical Modelling. Information Technologies. Automated Management Systems"*, 2004, No.629, pp.56-69
20. [DEKV05] Dzhurinsky E.N., Ermolayev, V.A., Keberle, N.G., Vladimirov, V.N., Visual Semantic Query Formulation and Execution in UnIT-NET IEDI, In: *Herald of Kharkiv Karazin National University, series "Mathematical Modelling. Information Technologies. Automated Management Systems"*, 2005, No.703, pp.95-108
21. [EKSV04b] Ermolayev, V.A., Keberle, N.G., Shapar, V.V., Vladimirov, V.N. Ontology-Driven Sub-Query Extraction for Distributed Autonomous Information Resources in UnIT-NET IEDI, In: *S.W. Liddle, T.Halpin, H.C. Mayr, A. Doroshenko (Eds.):Proceedings of the 3rd International Conference on Information Systems Technologies and its Applications,* GI-LNI P-48, pp.137-150, July 14-16, Salt Lake City, Utah, USA, 2004.