

Tempus MP JEP 23010-2003  
IT in University Management Network



**The Infrastructure for  
Electronic Data Interchange  
Reference Architecture  
Specification**

**ID:** D2.2.D.1  
**Version:** 1.0  
**State:** Working draft  
**Editor:** Vadim Ermolayev  
**Last update:** 28/02/2004  
**Document type:** Restricted access  
**Authors:** Andrew Bulat, Edwin Gray,  
Vadim Ermolayev, Natalya Keberle,  
Sergey Plaksin, Vladimir Shapar,  
Vladimir Vladimirov,  
Gregory Zholtkevich

---

**Abstract:** Electronic Data Interchange (EDI) is the computer-to-computer exchange of highly structured business data between one application and another within a trading community, this case, the universities and state organizations of Ukraine. This Specification drafts the reference architecture of IEDI. IEDI is the software infrastructure supporting EDI. More precisely, IEDI is the multi-layered distributed information system comprising the software servers, services, components and tools for providing intelligent ontology-driven information retrieval from distributed, heterogeneous, legally and physically autonomous Information Resources (IRs) in the organizational framework of the National Higher Education System.

IEDI reference architecture is built according to the following principles: the architecture is of mediator type using a centralized mediator; it exploits the hybrid approach to knowledge representation; it uses information resource registration to allow the resource become available for querying; it does not provide full automation for ontologies' mapping and alignment; IEDI components use rewriting techniques with mappings to produce, process, and perform queries.

---


**Status of this Document:** Draft recommendation

## Table of contents

Glossary of Terms and Abbreviations .....	1
1 Introduction .....	3
2 The Tasks for UNIT-NET IEDI.....	5
2.1 User Roles and Categories .....	5
2.2 Querying Distributed Autonomous Semantically Heterogeneous Information Resources .....	6
2.3 Registering Information Resources.....	7
2.4 Maintaining Coherent Semantic Descriptions.....	10
3 A Walkthrough Example .....	14
3.1 IR Registration .....	14
3.2 Querying IEDI .....	18
4 IEDI Reference Architecture .....	26
4.1 Architectural Layering .....	26
4.1.1 Architecture layers, Layer Components and User Roles .....	26
4.1.2 IEDI Clients and Servers .....	26
4.2 Ontologies.....	28
4.2.1 The Types of IEDI Ontologies.....	28
4.2.2 Ontologies Hierarchy .....	29
4.2.3 Ontologies in IEDI Processes .....	30
4.2.4 Ontology Representation Language.....	30
4.2.5 IEDI Ontology Repository.....	31
4.3 Perform Query Web Service .....	32
4.4 IR Wrappers.....	33
4.4.1 IEDI Generic Wrapper.....	33
4.4.2 IR Wrapper Bindings .....	34
4.5 Control Flows .....	34
4.6 Data Flows .....	37
5 Query Formulation.....	39
5.1 Requirements to the Query Formulation Tool Interfaces.....	39
5.2 Interaction with an AU.....	40
6 Query Decomposition.....	41
6.1 Extracting Sub-Queries for Different IR.....	41
6.1.1 Sub-queries extraction algorithm .....	41
6.2 Sub-Query Extraction by a Walkthrough Example.....	43
7 Query Performance.....	48
7.1 Query Translation .....	48
7.1.1 Query Translation Algorithm.....	49
7.1.2 Query Translation by a Walkthrough Example .....	49
8 Authentication and Security.....	53
8.1 Security Levels.....	53
8.2 Interactions of Authentication and Security Subsystem with IEDI Components.....	53
8.3 Acknowledgement of Public Key Certificates Validity .....	55
8.4 Authentication of the Sender of an Information Package .....	55
8.5 Verification of Information Package Integrity .....	55
8.6 Encryption and Decryption Information Package.....	55
8.7 Signature Information Package.....	57
References .....	58

Appendices .....	60
A. OWL code for the walkthrough example fragments of the University Entrant IRO, the University Faculty Students IRO and MDO.....	60




	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>Glossary of Terms and Abbreviations - 2.1 User Roles and Categories</b>	<b>28/02/2004</b>

## Glossary of Terms and Abbreviations


API	Application Programming Interface	Comments
<b>AU</b> <sup>1</sup>	Authorized User	
DAML+OIL	DARPA Agent Markup Language + Ontology Inference Layer:	
DBMS	DataBase Management System	
FaCT		
<b>IEDI</b>	The Infrastructure for Electronic Data Interchange	
I2R	Intelligent Information Retrieval	
I3	Intelligent Information Integration	
IR	Information Resource	
<b>IRDMO</b>	IR-Domain Mapping Ontology	
<b>IRO</b>	IR Ontology	
<b>IROE</b>	Information Resource Ontology Engineer	
<b>IRP</b>	Information Resource Provider	
<b>IRQL</b>	IR Query Language	
JDBC	Java DataBase Connectivity	
JVM	Java Virtual machine	
<b>MDO</b>	Mediator Domain Ontology	
Mediator	A general definition by G. Wiederhold: "A mediator is a software module ( <i>or a set of module components</i> <sup>2</sup> ) that exploits encoded knowledge about certain sets or subsets of data to create information for a higher layer of applications" Source: [WIE92]	
<b>MKB</b>	Mediator Knowledge Base	
<b>MOE</b>	Mediator Ontology Engineer	
MS	Microsoft	
MySQL		
ODBC	Open DataBase Connectivity	
Ontology	Ontologies are developed to provide a machine-processable semantics of information sources that can be communicated between different software and humans. A definition of an ontology by Fensel bases on the on the related definitions in [Gruber, 1993]: <i>An ontology is a formal, explicit specification of a shared conceptualisation.</i> <i>Conceptualisation</i> here stands for a simplified abstract model of some object or phenomenon in the world which identifies the relevant concepts of that object or phenomenon. <i>Explicit</i> means that the type of concepts used and the constraints on their use are defined explicitly – i.e., are specified by the means of the external formal language as the descriptions separate from the software or humans processing these descriptions. <i>Formal</i> refers to the fact that the ontology should be machine readable. Hereby different degrees of formality are possible. Adopted from: [FEN00]	
OWL		

<sup>1</sup> Abbreviations made **bold** are the abbreviations of the UNIT-NET IEDI Reference Architecture Specification.

<sup>2</sup> Extended (*italic*) by the editor of the Specification

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>Glossary of Terms and Abbreviations - 2.1 User Roles and Categories</b>	<b>28/02/2004</b>

OWL DL		
<b>QFL</b>	Query Formulation Language	
<b>QFT</b>	Query Formulation Tool	
RDF	Resource Description Framework	
RDFS	RDF Schema	
RDQL	<b>Query Language for RDF:</b> An RDF <a href="#">model</a> is graph, often expressed as a set of triples. An RDQL query consists of a graph pattern, expressed as a list of triple patterns. Each triple pattern is comprised of named variables and RDF values (URIs and literals). An RDQL query can additionally have a set of constraints on the values of those variables, and a list of the variables required in the answer set. Source: [RDQL04]	
SOAP	Simple Object Access Protocol	
SQL	Structured Query Language	
<b>UNIT-NET</b>	The network of Ukrainian Universities and State Bodies which participate in collaborative activities in establishing, developing and maintaining the IEDI by providing their IR-s in the way specified by this Reference Architecture Specification.	
<b>ULO</b>	Upper Level Ontology	
<b>UPRO</b>	User Profile Reference Ontology	
URI	Universal Remote Identifier	
URL	Unified Remote Locator	
<b>WKB</b>	Wrapper Knowledge Base	
Wrapper	<p>A software component which is combined with another software component to determine how that software component is executed. The wrapper acts as an interface between its caller and the wrapped component. This may be done for compatibility, e.g. if the wrapped component is in a different programming language or uses different calling conventions, or for security, e.g. to prevent the calling program from executing certain functions. The implication is that the wrapped component can only be accessed via the wrapper.</p> <p>Hence, wrapping a software system, like an IR Server is the process of defining and restricting access to a system through an abstract interface.</p> <p>A wrapper for an IR accepts queries in a given format, converts them into one or more commands or sub-queries understandable by the underlying IR and transforms the native results into a format understood by the application.</p>	
Web Service	A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP-messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards. Source: [WSA03]	
<b>WS</b>	Wrapper Web Service	
XML	eXtensible Markup Language	

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>1 Introduction - 2.1 User Roles and Categories</b>	<b>28/02/2004</b>

## 1 Introduction

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.

Consequently, in this 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 in 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. 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(-ues) 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


	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>1 Introduction - 2.1 User Roles and Categories</b>	<b>28/02/2004</b>

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 architecture 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 are not aimed to broaden the horizons of the current state of the art in I3 or, more specifically, in I2R. The task is 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 the national level.



	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>2 The Tasks for UNIT-NET IEDI - 2.1 User Roles and Categories</b>	<b>28/02/2004</b>

## 2 The Tasks for UNIT-NET IEDI

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:

- They might be opened or closed to external access
- They might be provided by different hardware and software using various notations and protocols
- They might be disparately structured or even have different data models behind them
- They are semantically heterogeneous

The purpose of UNIT-NET IEDI is to attempt to overcome some of these complications by providing a uniform framework for authorized and secure information retrieval from heterogeneous, distributed, autonomous information resources. One of the central points of IEDI is the uniform and coherent representation of the Domain and IR semantics by means of the family of Ontologies. Therefore, the processes of querying the resources of IEDI are ontology-driven and cannot be performed entirely automatically. Some of them require intensive ontology engineering by different human executives.

Another aspect to be mentioned with respect to IEDI functional processes is the state of the system distributedness. A query may demand to retrieve data from several physically 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 harmonization, supplied with appropriate methodologies and software tools. Human activities are performed by various user roles: authorized user, domain ontology engineer, IR ontology engineer, resource provider.

### 2.1 User Roles and Categories

IEDI user roles define the types of human users or actors which have different spheres of influence within the system and participate in different processes and underlying tasks. The roles are:


- Authorized user (AU)

An AU is one who has registered at IEDI and has received the digital certificate according to the procedure defined by the IEDI Authentication and Security Specification (see also Section 8 of this Specification and [IEDI-ASS04]). An AU's goal is to employ IEDI to retrieve the necessary data by formulating and posing queries to the system. The process this role is involved in, is Querying Information Resources (Section 2.2. of this Specification). An AU interacts through a web browser with IEDI Mediator Query Formulation component (Section 5. of this Specification). The queries are formulated in terms of the IEDI **Domain Ontology** (Section 4.2 of this Specification, [IEDI-DOS04]) by means of the IEDI **Query Formulation Language** (Section 5 of this Specification, [IEDI-QFLS04]). An AU also receives the results of the query marked up in terms of the IEDI Domain Ontology from the IEDI Mediator after the query is processed.

A software component may also be considered an AU in the context of this role specification. If an AU is a software component it SHOULD be capable of formulating queries (or have some library of pre-defined queries) in IEDI Query Formulation Language.

- IEDI Mediator Ontologies Engineer (MOE)

The main function of the MOE is to provide and maintain coherent semantic specifications of the knowledge and data integrated by IEDI. MOE develops and maintains the family of mediator ontologies comprising the **Upper Level Ontology** (Section 4.2 of this Specification, [IEDI-DOS04]), the **Domain Ontology** (Section 4.2 of this Specification, [IEDI-DOS04]) and the Mediator Reference Ontologies for User Profiling and IR Ontology Mapping. MOE maintains the Mediator Knowledge Base (MKB) (Section 4.2.5 of this Specification). IEDI provides the TOOL for this activity (Section 4.1 of this Specification). MOE has priority over the IR Ontology Engineer in taking decisions on IR-Domain Ontology merging and alignment.

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>2 The Tasks for UNIT-NET IEDI - 2.2 Querying Distributed Autonomous Semantically Heterogeneous Information Resources</b>	<b>28/02/2004</b>

MOE interacts with the respective IR Ontology Engineer in the processes of IR Registration (Section 2.3 of this Specification) and Maintaining Coherent Semantic Descriptions (Section 2.4 of this Specification) within IEDI life cycle. Both parties exploit the Ontology Discussion Tool for these negotiations.

- IR Ontology Engineer (IROE)

The function of an IROE is to provide and maintain semantic specifications of the knowledge and data of the IR. IROE is the representative of the IRP for the IEDI processes of IR Registration (Section 2.3 of this Specification) and Maintaining Coherent Semantic Descriptions (Section 2.4 of this Specification)

- IR provider (IRP)

IRP is normally a legal entity (an organization, like a University) which owns its Information Resource. IROE is normally employed by an IRP. IRP is responsible for the following tasks:

- Providing information for registering in an MKB and maintaining coherence between the semantics of the IEDE and of given IRP. IRP provides information using Resource Semantics Description Language. It means that each IRP should support requests for the current state of its metadata and send notification to the IEDE Mediator each time when the event connected with an IRP's semantic change occurs;
- Verification of the AU's rights to obtain requested information and maintenance of the information security rules (Section 8 of this Specification). IRP SHOULD NOT provide information about AU rights, but should notify the AU (in the response to the request result) about the lack of rights;
- Query processing and responding with the results of the request in the form of Query Results Representation Language.

## 2.2 Querying Distributed Autonomous Semantically Heterogeneous Information Resources


The overall process receives as its input an AU request to formulate and to perform a query. It generates as its output the set of partial query results received from IR wrappers, coherently marked up in terms of the IEDI Domain Ontology. AU then performs the analysis of the obtained results on his/her own or with the help of third party software. Results analyses and possible fusion are beyond the scope of IEDI. The process involves the following types of user roles:

- AU – takes part in the tasks and activities of Query Formulation, Query Assessment, Result Analysis
- MOE – takes part in the activities of Mediator Ontology Problem Analysis, Ontology Repair
- IROE – takes part in the activities of Resource Ontology Problem Analysis, Ontology Repair
- IRP – provides the authentication for AU and the access to IR by its Resource Wrapper

In more detail (see Fig. 1), the process is initiated by an AU contacting the IEDI Mediator through his/her web browser by visiting the corresponding URL. The Mediator Provides the AU with the Query Formulation Interface. The Query Formulation Task is performed in interaction between the AU and the Mediator. The terminology for the query formulation is provided by the Mediator Domain Ontology stored in the Mediator Knowledge Base (MKB). The mappings from the AU personal conceptualization of the Domain to the Mediator Domain Ontology are provided by the User Profile Reference Ontology which is incrementally collected at the MKB. More details on the Query Formulation Task are provided by Section 5 of this Specification.

After the query is formulated and written down in the notation of the IEDI Query Formulation Language it is assessed by the AU. After assessment the query is put through as input to the Sub-Query Extraction Task if in the opinion of the AU, the query is formulated correctly. Otherwise, it is returned back to re-formulation. An AU has the possibility to terminate the process at the formulation phase if he/she decides to do so. The context of the failure within the Query Formulation Task is sent to the MOE for further asynchronous analysis.

The purpose of the Sub-query Extraction Task is to generate partial queries to different IRs from the input query. The task is performed automatically by the Mediator Sub-Query Extraction Component (see Section 6 of this Specification for more details). Knowledge on which part of the input query may be transformed to the query to the particular IR is provided by the IR-Domain Mapping specified by the Mediator IR-D Reference Ontology. These Mappings are stored in the MKB as the results of the respective IR Registration Process (Section 2.3. of this Specification) and the process of the alignment of the IR and Domain Ontologies within the IR lifecycle. Possible failures of the Sub-Query Extraction may be caused by inconsistencies between the Mappings of various IR Ontologies to the Domain ontology – i.e. by the ontology problems. The Report Ontology Problem Task is if there is an extraction failure.

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>2 The Tasks for UNIT-NET IEDI - 2.3 Registering Information Resources</b>	<b>28/02/2004</b>

The goal of the Report Ontology Problem Task is firstly – to send the Problem Content to MOE for further Analysis and Ontology Repair Tasks and secondly – to attempt to detect if the Ontology Problem was critical or not and either terminate the process, or return it to the Query Formulation.

The input Sub-Query Queue is formed by the Extraction Subtask in the case of the extraction being successful. Perform Sub-Queries Task gets input Sub-Queries one by one from the queue and prepares their execution by:

- Generating SOAP content for the query execution web service of the respective IR Wrapper
- Invoking this web service

The Sub-Query encapsulated in the SOAP context is later used by the IR Wrapper Web Service to transform and to execute the query to its IR. Authentication and Security Policies for this task are provided by the IRP. Query results marked up in the terms of the IR Ontology are then returned to the Mediator. More details on the Sub-Query Translation Sub-Query Performance and Sub-Query Results Mark-Up Tasks are given in Sections 4–7 of this Specification.

Query Result Mark-Ups are then translated to the terms of the Domain Ontology in the frame of Map Query Result Mark-Up to Domain Ontology activity. IR-D Reference Ontology again provides the terminology mappings for this purpose. Marked-up sub-query results are then delivered to the AU browser and the process terminates.

## 2.3 Registering Information Resources

It is necessary that IR-s are registered to IEDI before it is possible to retrieve information. The registration is essentially the process of adding and aligning the semantic descriptions of the particular IR to the MDO. The process involves the following types of user roles:

- AU-s – express their interest in getting the information from the particular resource of the particular IRP
- MOE – takes part in the task of IR Registration to the MDO, performs MDO and IRDMO updates
- IROE – prepares the registration by performing the task of IRO Design and Deployment to the IR WKB
- IRP – takes the decision on IR registration, manages the activities of IRO design, IR Wrapper Deployment

In more detail (see Fig. 2), the process is initiated by the AUs' expressions of interest to use the particular IR for querying. These expressions of interest are submitted to the IRP by messages. IRP may consequently decide to make its IR available to IEDI and, therefore, initiates the task of the Preparation of the IR for Registration by sending the request message to its IROE. IROE then retrieves the IR Metadata and designs the IRO [IEDI-RWS04]. After the IR wrapper is deployed and the IRO is uploaded to its WKB the IR is ready and may be submitted to the registration.

The submission is done by the message containing the request to IR Registration and the URI of the IRO. In response the MOE uploads the IRO and performs its input check in conformance with the IRO Specification [IEDI-RWS04]. MOE requires the IRO to be re-designed and re-submitted if the conformance check fails. Otherwise MOE proceeds with the registration by acquiring the copies of MDO and IRDMO and initiates the activity on ontology merge and align negotiation. This negotiation is the manual collaborative process performed together by the MOE and the IROE and supported by the IEDI Ontology Negotiation Tool. The changes in MDO and IRDMO copies reflecting the appearance of the new IR to IEDI are done within the negotiation activity. The MOE would reject the IRO and request its re-design and re-submission. In case the of negotiation agreement not being struck. Otherwise, the MOE locks the MKB, uploads the copies of MDO and IRDMO to MKB and unlocks the MKB thus making the IR registration and respective ontology modifications complete. An example registration is discussed in Section 3.1.

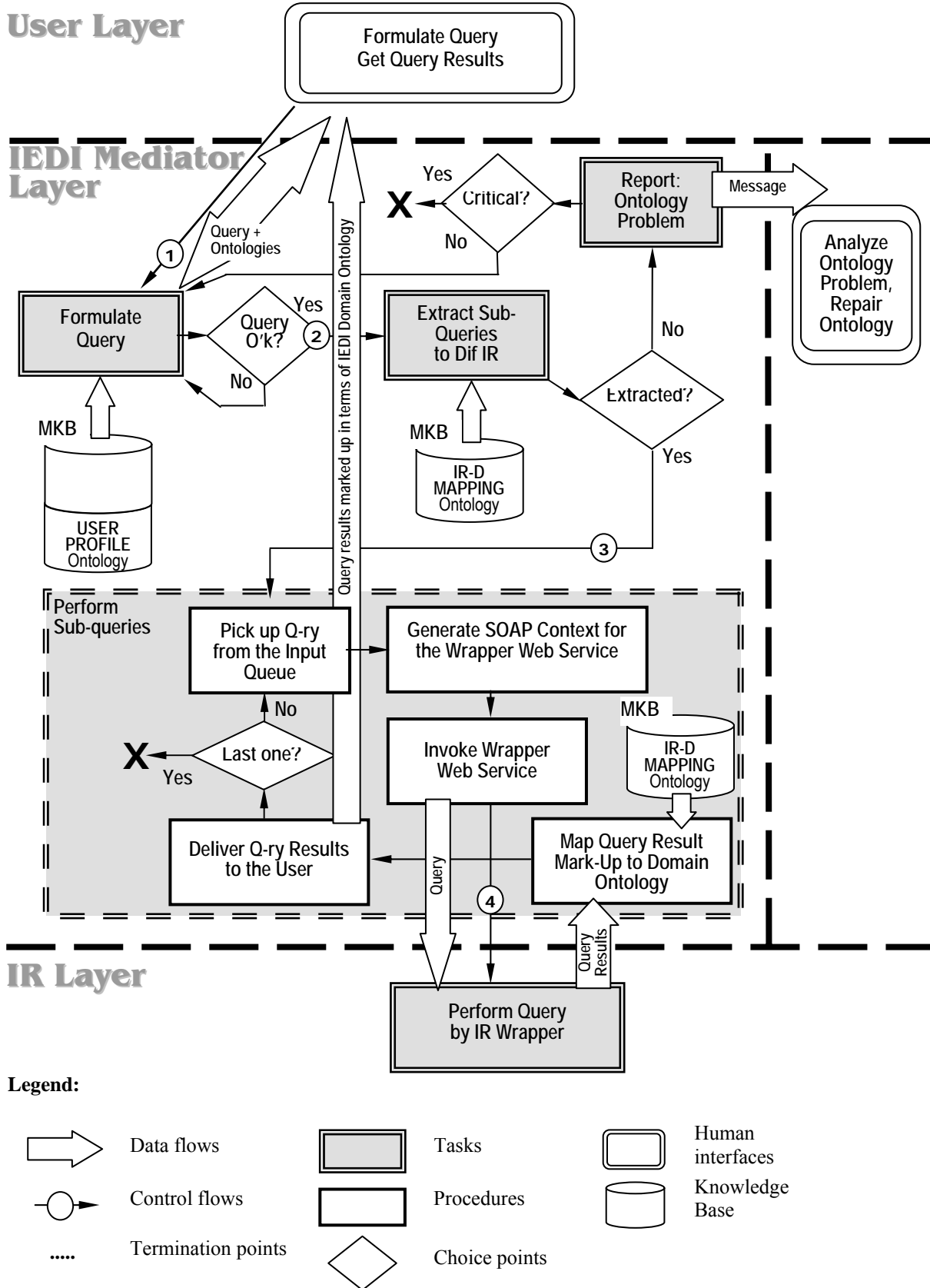


Fig. 1. IEDI Querying Process. The legend is valid for subsequent Fig. 2-4.

## IEDI User Layer

### IEDI Mediator Layer

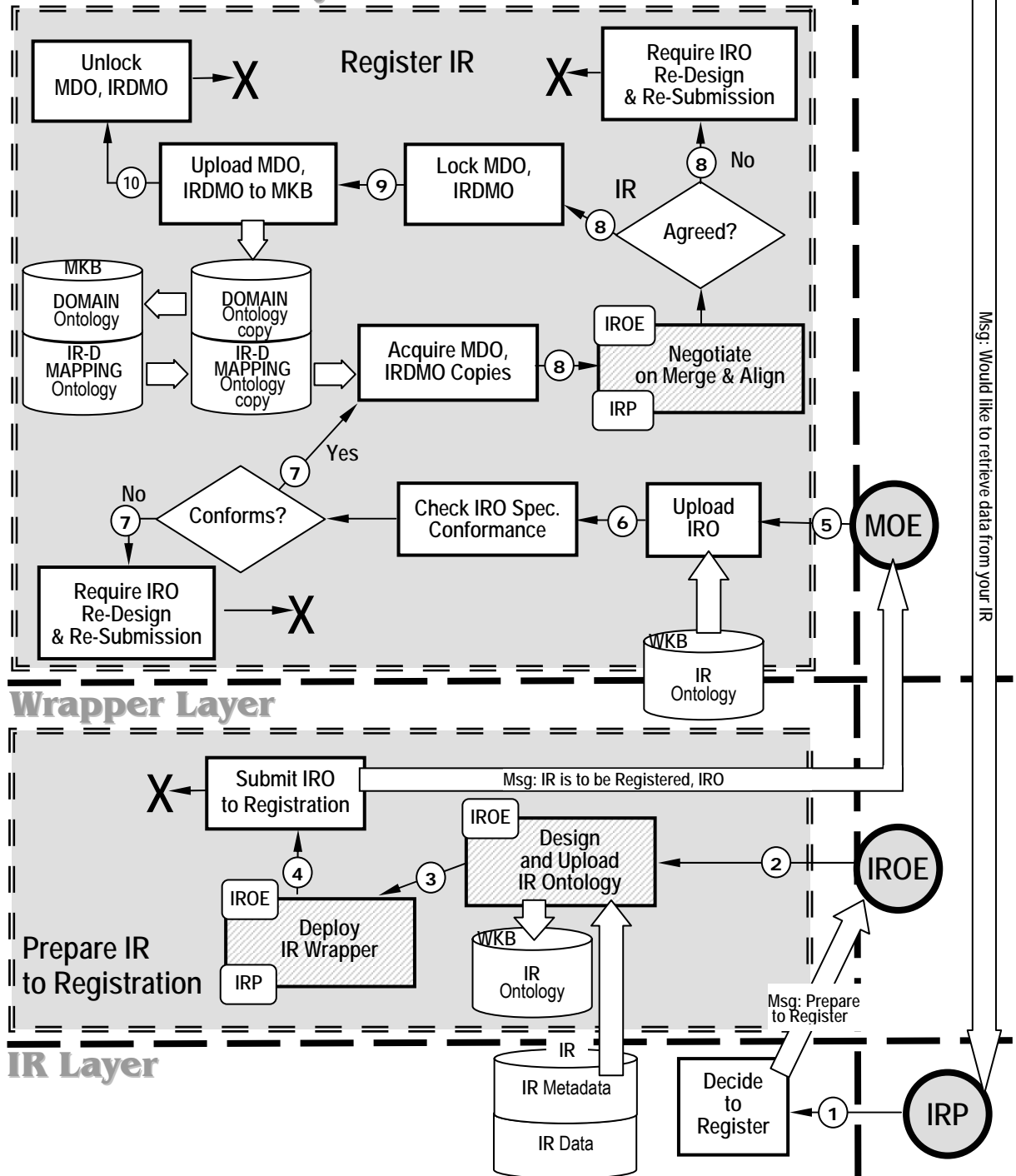



Fig. 2. The processes of IR registration.

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>2 The Tasks for UNIT-NET IEDI - 2.4 Maintaining Coherent Semantic Descriptions</b>	<b>28/02/2004</b>

## 2.4 Maintaining Coherent Semantic Descriptions

An important IEDI functionality is maintaining all its semantic descriptions in a coherent and agreed state. It involves the following types of user roles:

- MOE – takes part in the processes of IR Semantics Change Propagation, Mediator Domain Ontology revision
- IROE – takes part in the process of IR Semantics Change Propagation
- IRP – maintains the changes of the IR within its life cycle and initiates, when necessary, the process of IR Semantics Change Propagation

First, the need to maintain the coherence between the ontologies of IEDI arises when a new IR is registered to the Mediator or the IR semantics is changed:

- During an IR registration process the IEDI Domain ontology and the IR ontology are merged,
- Mediator Domain Ontology (MDO) may be extended to capture new concepts.

Detailed description of the activities comprising IR registration process is given in Section 2.3 of this Specification.

Second, a User Profile Reference Ontology (UPRO) is updated each time a user, while formulating a query, uses a new term, which was unknown to the Mediator before (not in the Domain Ontology).

Third, though IEDI Upper Level Ontology (ULO) is a kind of semantic grounding for IEDI and is considered “static”, some changes may still occur in it. However it is important to mention that the modifications to ULO should be first agreed by the group of people responsible for the maintenance of the UNIT-NET mediator. Frequent changes show that the wrong ULO was adopted or it was badly engineered by the UNIT-NET team.

Fourth, changes in IRO-s occur, but not under IEDI Mediator control as far as they are maintained autonomously by respective IROE-s. It is assumed that the updates in the resource semantics performed by autonomous IRP teams initiate the process of change propagation to the Mediator side.

The changes in MDO may be caused by incorporating new IR-s to IEDI. When a new IR is registered to IEDI it may appear that its IRO brings new domain knowledge extending the shared mediator ontology (MDO). Moreover, the semantics of a registered resource may be sporadically changed within the IR life cycle. These changes should be adequately reflected in MDO. However the process of change identification, adaptation and propagation is performed mostly manually in negotiations between the MOE and the respective IROE. IEDI at most provides some software tools, like the ones for ontology editing and ontology discussion. The appearance of new IR-s at the IEDI scene may also cause the necessity for the revisions and the harmonization of MDO. This process is performed solely by the MOE.


Both IRO and MDO updates affect IR-Domain Mapping Ontology (IRDMO). The updates of IRDMO are performed automatically as the activities within the IEDI processes of Maintaining Coherent Semantic Descriptions. The typical processes are:

- Propagate IR Semantics changes to MDO
- Revise MDO

The process of IR Semantics Change Propagation (see Fig.3) is initiated by an IRP, which invokes its IROE to revise the IRO according to the changes in the IR. IROE starts with the Task of Detecting Changes. It locks the IR and notifies MOE about it. MOE reacts by locking the IR at the Mediator, since the IR will not receive AU (sub-)queries until it is unlocked after the Ontology Change Propagation is accomplished. MOE also initiates Propagate Changes Task and waits for the IRO changes notification message from IROE.

IROE proceeds with analyzing IR changes by comparing old and new IR semantics. If it determines that the changes should be reflected in IRO it acquires and updates the IRO copy. Updates Log is generated by this activity and then forwarded to MOE, together with the message informing about IROE readiness for the Negotiation on IRO changes propagation. IROE also updates the IRO by uploading the IRO copy and unlocks the IR after the changes in the IRO copy are done.

After receiving the ready-to-negotiate message from the IROE the MOE prepares for collaborative manual MDO-IRO alignment by acquiring the copies of MDO and IRDMO. All the changes will be subsequently made to the copies. The copies are used: firstly – to ensure that IEDI may be used for processing queries at the time of the ontology updates, and, secondly – to apply only agreed and approved updates to IEDI ontologies. The next activity within the process is aimed at performing the negotiations and to collaboratively align MDO, IRO and IRDMO. The activity is performed manually by IROE and MOE. The main input forming the negotiation set is the Updates Log provided by the IROE. The procedure of ontology alignment is quite similar to that of the IR


	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>2 The Tasks for UNIT-NET IEDI - 2.4 Maintaining Coherent Semantic Descriptions</b>	<b>28/02/2004</b>

Registration Process (refer to Section 2.3 of this Specification) and is performed until all the IRO changes are discussed and adequately mapped to the MDO. The difference between the IR Registration process is that the alignment procedure here should consider the deletions of the IRO and reflect these deletions properly to the IEDI ontologies. The significant point here is that the ontology element marked as deleted in the Updates Log may be deleted from the MDO and IRDMO only in the case where there are no mappings of this very element to another IRO-s of another IR-s. Another point is that, even if the ontology element is no longer associated with any IRO and may be deleted from MDO and IRMO, the MOE may still decide to keep the element in MDO for various reasons (e.g., the element is in the MDO Core [IEDI-DOS04], MOE expects that the element will be associated with the respective IR-s, ontology structure, etc.)

MOE locks the MDO and the IRDMO after the negotiation on the ontology changes propagation is accomplished and the changes to the ontology copies are done. The copies are then uploaded to the MKB, the ontologies are unlocked and the process terminates.

To summarize, the process of IR Semantics Change Propagation is actually the combination of the two concurrent tasks for IR Changes Detection and IRO Changes Propagation. These activities are led by IROE and MOE respectively. IR and IEDI ontology locks are used for synchronizing these activities.

The process of MDO Revision is evidently very similar to the IRO Changes Propagation task of the former process. The only difference is that it doesn't require the participation of any IROE-s and is performed by MOE on his/her own. Refer to Fig. 4 for more details.

	MP JEP 23010-2003 "IT in University Management NETWORK"		IEDI-Ref-Arch-DR-10.doc
	Document	IEDI Reference Architecture Specification. Draft Recommendation	Version 1.0
	Topic	2 The Tasks for UNIT-NET IEDI - 2.4 Maintaining Coherent Semantic Descriptions	28/02/2004

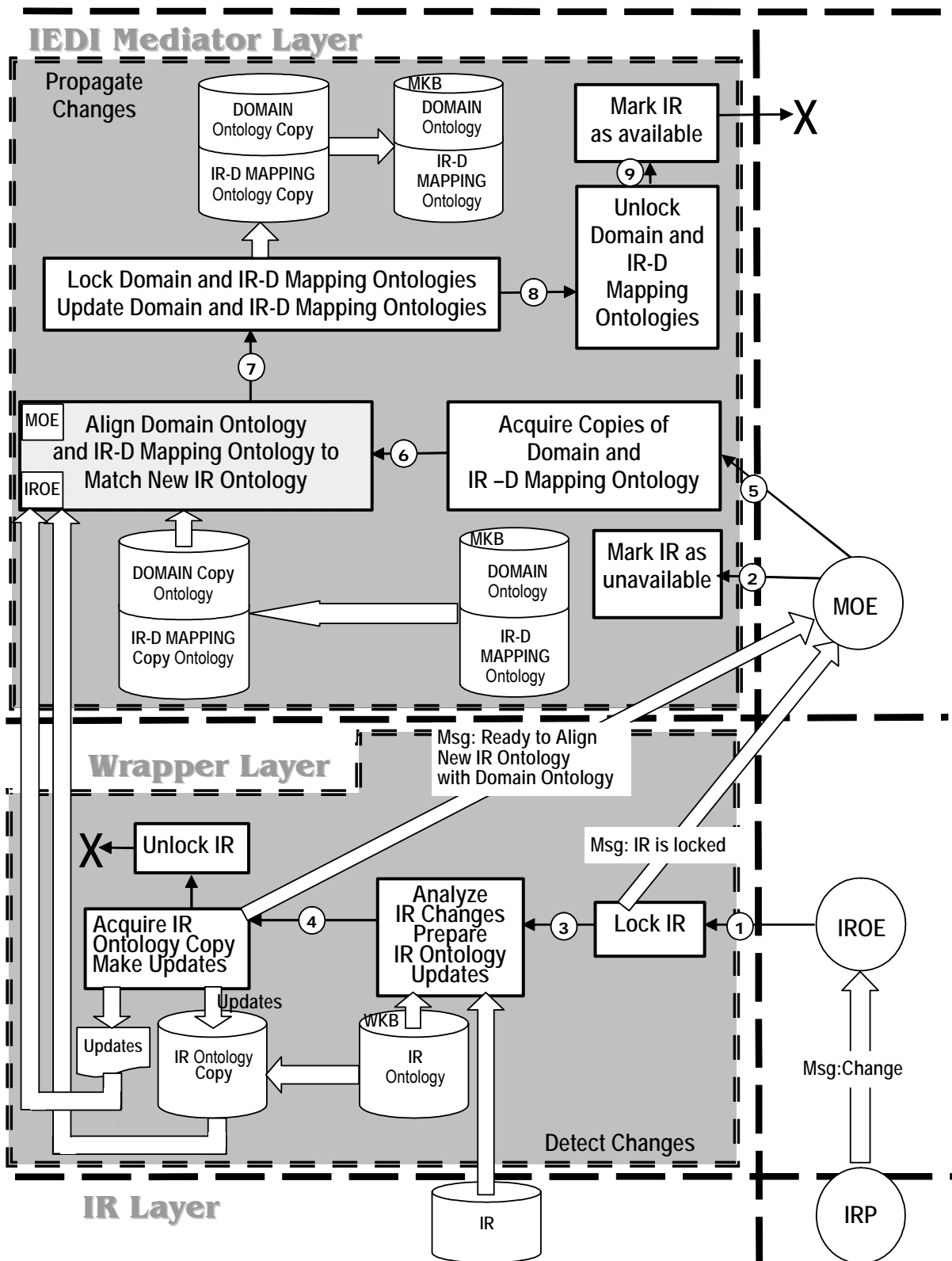



Fig. 3. The processes of Detecting IR Semantics Changes and Propagating IRO Changes.



	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>2 The Tasks for UNIT-NET IEDI - 2.4 Maintaining Coherent Semantic Descriptions</b>	<b>28/02/2004</b>

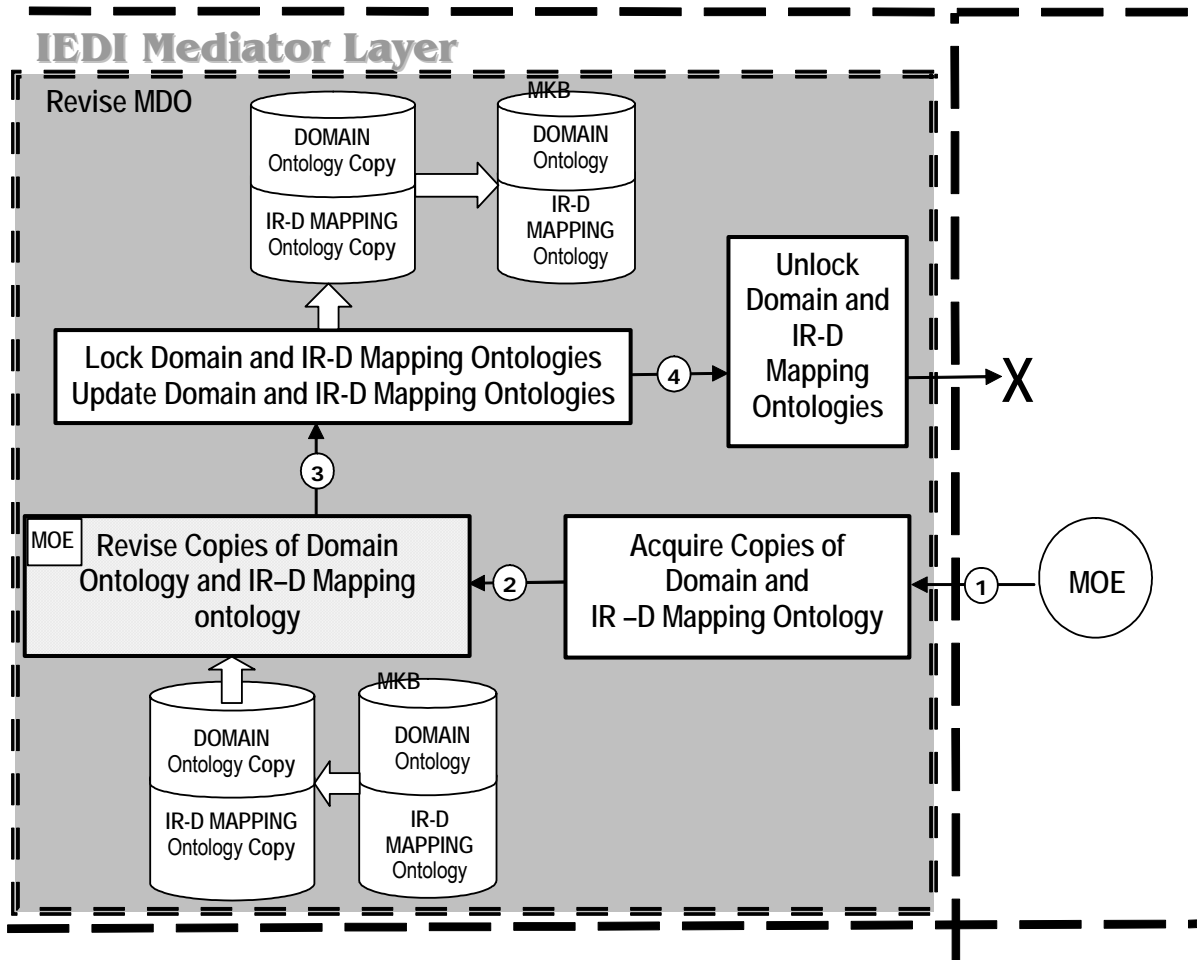



Fig. 4. "Propagate changes" task when changes affect only Domain ontology.

	MP JEP 23010-2003 "IT in University Management NETWORK"		IEDI-Ref-Arch-DR-10.doc
	Document	IEDI Reference Architecture Specification. Draft Recommendation	Version 1.0
	Topic	3 A Walkthrough Example - 3.1 IR Registration	28/02/2004

### 3 A Walkthrough Example

This section demonstrates how IEDI functionality scenarios may be utilized in a practical application in the University Management Domain. The usage is demonstrated by the example of a typical query, the results of which might be useful, for example, in the assessment of the quality of secondary education in a region, or at a National level:

**Retrieve the list of the 1-st year students who have received maximum 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 an MS Access application. Example query fragments of the DataBase Schemas for these IR-s are given in Fig. 5 and Fig. 6.

#### 3.1 IR Registration

It is assumed in IEDI that an IR should first be registered (Section 2.3) before becoming available to AU queries. The registration process comprises the design and deployment of the IRO and subsequent merge with the IRO to the IEDI MDO. The graphical representation for the example IRO fragments is shown in Fig. 7. The ontologies are formalized in OWL [OWL03] by means of the Protégé 2000 Ontology Editor Ver. 1.9. [NSD01]. The fragments of the OWL code for the University Entrant IRO and the University Faculty Students IRO are given in Appendix A.

As described in Section 2.3., the registration of the IR to IEDI comprises merging the respective IRO with the MDO. This merge is performed collaboratively by the MOE and the IROE and affects MDO and IRDMO. The changes to 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 in the IRDMO in the form of mappings. These mappings are later used in transforming AU queries. In 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

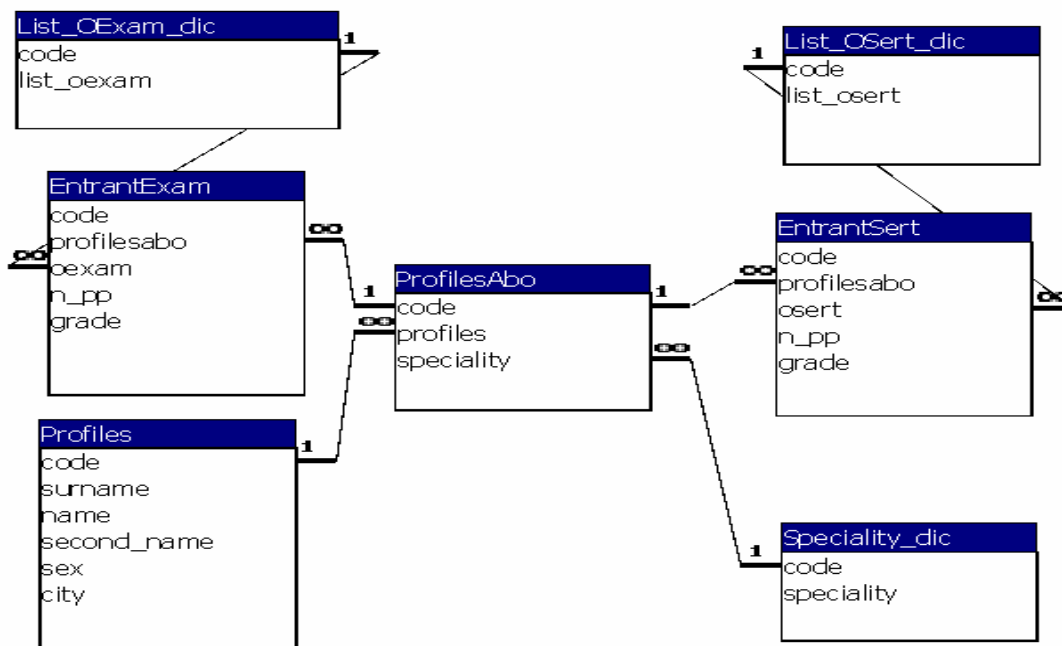



Fig. 5. The fragment of the DB Schema for the University Entrant IR.

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>3 A Walkthrough Example - 3.1 IR Registration</b>	<b>28/02/2004</b>

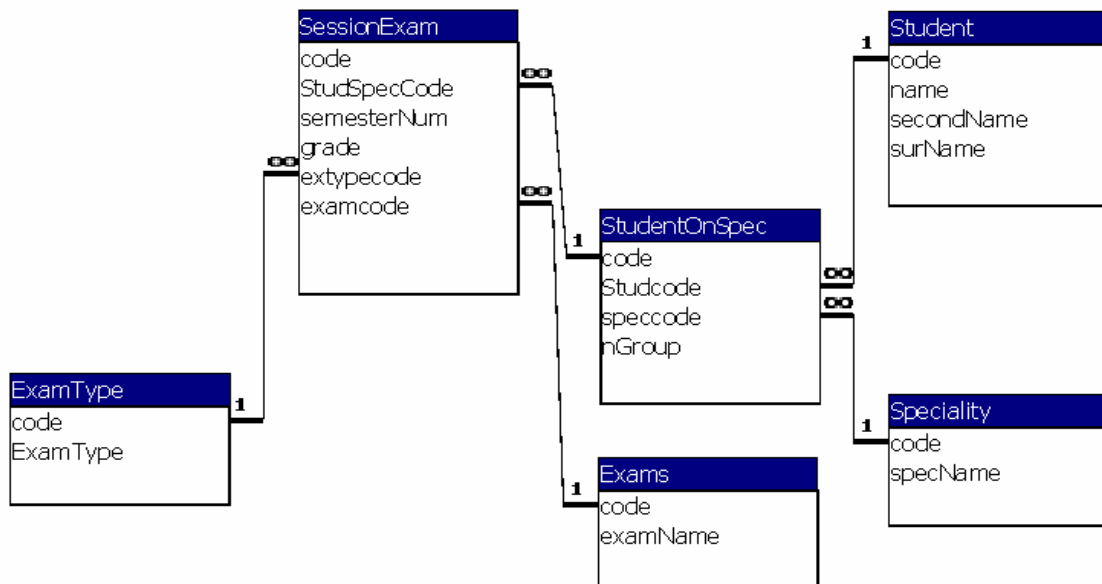


Fig. 6. The fragment of the DB Schema for the University Faculty Students IR.

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 the <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 since the concept does not induce any relationships to another concepts.

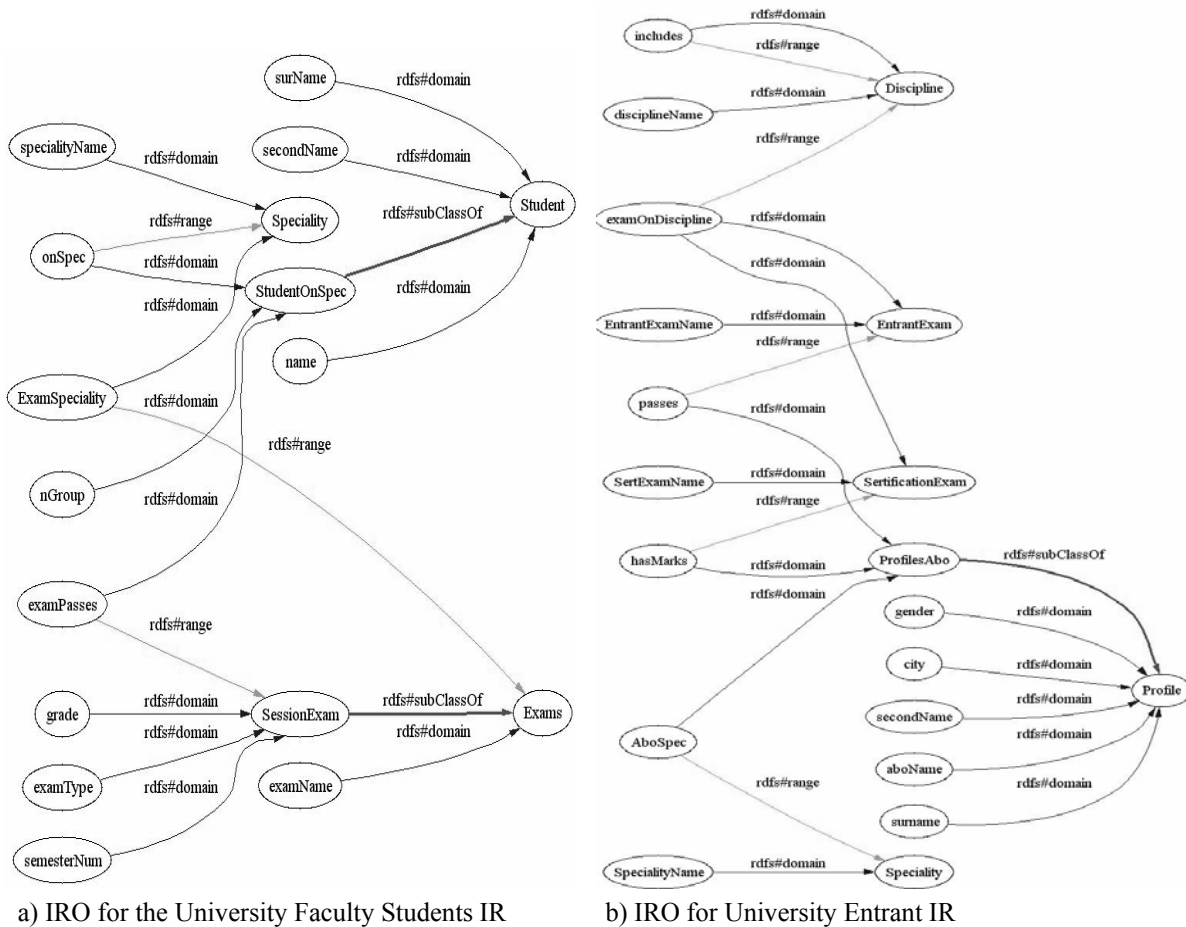
- IRO: <Speciality> and MDO: <Speciality>

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

	MP JEP 23010-2003 "IT in University Management NETWORK"		IEDI-Ref-Arch-DR-10.doc
	Document	IEDI Reference Architecture Specification. Draft Recommendation	Version 1.0
	Topic	3 A Walkthrough Example - 3.1 IR Registration	28/02/2004



a) IRO for the University Faculty Students IR

b) IRO for University Entrant IR


Fig. 7. Graphical representation of the fragments of the IRO-s for the example IR-s.

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> - bijective together with their property IRO: <examName> ↔ 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>)

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>3 A Walkthrough Example - 3.1 IR Registration</b>	<b>28/02/2004</b>

- MDO: <entrant\_grade> (an Object property for MDO: <EntrantExam> concept, the sub-class of MDO: <Exam>)

Resulting IRO-MDO mappings are presented on Tables 1a and 1b. The graphical representation for the resulting MDO is given on Fig. 9.


Table 1a IRO – MDO Concept Mapping (example)

<b>Concept Mapping</b>		
<b>MDO Concept</b>	<b>IRO Concept</b>	<b>Resource Name</b>
Discipline	Discipline	Entrant
Person	Profile	Entrant
Person	Student	Faculty
Exam	EntrantExam	Entrant
Exam	SertificationExam	Entrant
Exam	SessionExam	Faculty
Exam	Exams	Faculty
EntrantExam	EntrantExam	Entrant
SertificationExam	SertificationExam	Entrant
SessionExam	SessionExam	Faculty
Speciality	Speciality	Entrant
Speciality	Speciality	Faculty
PersonOnSpeciality	AboSpeciality	Entrant
PersonOnSpeciality	StudentOnSpec	Faculty

Table 1b

IRO – MDO Slot Mapping (example)

<b>IRO – MDO Slot Mapping</b>				
<b>MDO Concept</b>	<b>MDO Slot</b>	<b>IRO Concept</b>	<b>IRO Slot</b>	<b>Resource Name</b>
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	examOnDiscipline	Entrant
Exam	examOnDiscipline	SertificationExam	examOnDiscipline	Entrant
Exam	exam_title	EntrantExam	EntrantExamName	Entrant
Exam	exam_title	SertificationExam	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

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>3 A Walkthrough Example - 3.2 Querying IEDI</b>	<b>28/02/2004</b>

IRO – MDO Slot Mapping				
MDO Concept	MDO Slot	IRO Concept	IRO Slot	Resource Name
SertificationExam	sertification_grade	SertificationExam	grade	Entrant
SessionExam	session_grade	SessionExam	grade	Faculty
SessionExam	semester_num	SessionExam	semesterNum	Faculty
PersonOnSpeciality	n_group	StudentOnSpec	nGroup	Faculty
PersonOnSpeciality	exams_passes	ProfilesAbo	passes	Entrant
PersonOnSpeciality	exams_passes	ProfilesAbo	hasMarks	Entrant
PersonOnSpeciality	exams_passes	StudentOnSpec	examPasses	Faculty
PersonOnSpeciality	on_spec	ProfilesAbo	AboSpec	Entrant
PersonOnSpeciality	on_spec	StudentOnSpec	onSpec	Faculty
Speciality	spec_name	Speciality	SpecialityName	Entrant
Speciality	spec_name	Speciality	specialityName	Faculty
Speciality	exam_spec	Speciality	ExamSpeciality	Faculty

### 3.2 Querying IEDI

As described in Section 2.2 the process of posing a query to IEDI comprises several stages:

- Query formulation
- Sub-query Extraction
- Sub-query performance

The query formulation stage is performed by an AU with the help of the IEDI Query Formulation Tool. The query is formulated in the terms of the MDO by choosing the necessary ontology elements and by applying the necessary constraints to them. The tool then generates the query in the notation of IEDI QFL. For the example presentation purposes we have chosen RDQL [RDQL04] query language. RDQL query for:


***Retrieve the list of the 1-st year students who have received maximum 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 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),

```

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>3 A Walkthrough Example - 3.2 Querying IEDI</b>	<b>28/02/2004</b>

```

(?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")
AND (?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. 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"), (?discName2 eq ?entrantExamName)
AND (?entrantGrade eq "5")
USING abo FOR <Univ. Entrant IRO URL#>

```

What this 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),

```

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>3 A Walkthrough Example - 3.2 Querying IEDI</b>	<b>28/02/2004</b>

(?x, stud:onSpec, ?a),  
 (?a, stud:specialityName, ?specialityName)



	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>3 A Walkthrough Example - 3.2 Querying IEDI</b>	<b>28/02/2004</b>

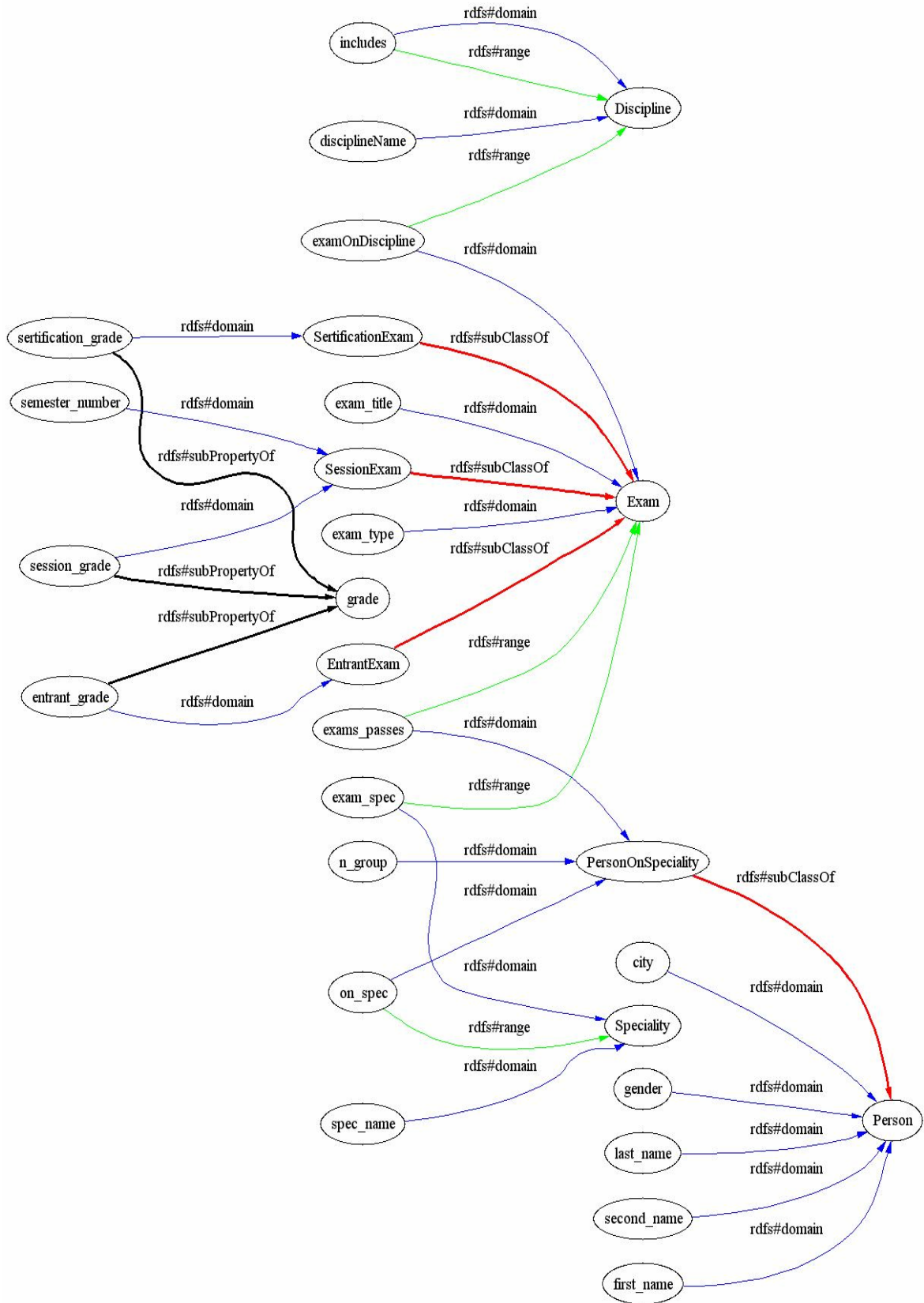



Fig. 9. Graphical representation of the MDO fragment for the example.

	MP JEP 23010-2003 "IT in University Management NETWORK"		IEDI-Ref-Arch-DR-10.doc
	Document	IEDI Reference Architecture Specification. Draft Recommendation	Version 1.0
	Topic	3 A Walkthrough Example - 3.2 Querying IEDI	28/02/2004

```

AND ((?examName eq "Linear Algebra")
|| (?examName eq "Mathematical Analysis"))
AND (?s eq "2")
AND (?g eq "1")
USING stud FOR <Faculty Student IRO URL#>

```

What means:

- in 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).***
- And, more formally:

From stud

Retrieve the instances of the following properties of the classes:

IRO:<Student>: <name>, <secondName>, <surName>,

IRO:<Speciality>: <specialityName>

For which it is true that (conjunction):

These instances have the following properties: <name>, <secondName>, <surName>, <nGroup>

These instances have the <onSpec> relationship with the instances of the class <Speciality>

These instances have the <examPasses> relationship (they pass the examination) with the instances of the class <Exam>

The instances of the <Exam> class have the properties: <examName>, <grade>, <semesterNum>

We are interested only in the instances of <Exam> for which the property:

<examName> = "Linear Algebra"

OR <examName> = "Mathematical Analysis"

AND

We are interested only in the instances of <StudentOnSpec> (subclass of <Student>) which are in relationship with the instances of <SessionExam> for which the properties:

<grade> = 2

AND <semesterNum> = 1

After the sub-queries are extracted they are put through to the respective resource wrappers for the execution. The wrappers are invoked via their web services.


Wrapper web service invocation context is specified in SOAP [SOAP03]. For example, the 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">

```

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>3 A Walkthrough Example - 3.2 Querying IEDI</b>	<b>28/02/2004</b>

```

<base64 encoded query goes here as the array of characters>
</req>
</m:IRWrapperQuery>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>


```

The query is then translated by the wrapper as described in the Section 7.1.

Query in the terms of IRO	MS SQL query in the terms of IR Schema
<pre> 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 &lt;Faculty Student IRO URL#&gt; </pre>	<pre> SELECT   Student.name, Student.secondName,   Student.surName,   Speciality.specialityName,   Exams.examName FROM   StudentOnSpec, ExamType,   Exams, SessionExam,   Student, Speciality WHERE   SessionExam.grade='2' AND   SessionExam.semesterNum=1 AND   ExamType.ExamType='Exam' AND   (     Exams.examName='Linear Algebra'   OR     Exams.examName='Mathematical Analyses'   ) AND   Exams.code = SessionExam.examcode AND   Student.code =     StudentOnSpec.Studcode AND   ExamType.code =     SessionExam.extypecode AND   StudentOnSpec.code =     SessionExam.StudSpecCode AND   StudentOnSpec.speccode =     Speciality.code; </pre>

The query is then executed by the IR Server (refer to Section 4.4. for more details). The results of the query execution are delivered as the 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
ТОВПІНЕЦЬ	ОЛЕКСАНДР	ОЛЕКСАНДРОВИЧ	Менеджмент орг.	Mathematical Analyses
ГЛАДИР	АЛЬОНА	ІГОРІВНА	Фінанси	Linear Algebra
ГРИГОР'ЄВА	ОЛЬГА	МИКОЛАЇВНА	Інформатика	Mathematical Analyses
ГАВРИЛЮК	ЮЛІЯ	СЕРГІЇВНА	Фінанси	Linear Algebra

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>3 A Walkthrough Example - 3.2 Querying IEDI</b>	<b>28/02/2004</b>

This result is then 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
      ...
      LlNwZWNPYXpdl0YW1lPgoJPGV4YW10YW1lPsL78fjg/yDs4PLl70Dy6OrgPC9leGFtTmFt
      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>
...
```

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>3 A Walkthrough Example - 3.2 Querying IEDI</b>	<b>28/02/2004</b>

```

<row>
  <last_name>ГАВРИЛЮК</last_name>
  <first_name>ЮЛИЯ</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:


<b>surname</b>	<b>name</b>	<b>second_name</b>	<b>speciality</b>
ОВЕРКО	НАТАЛІЯ	ОЛЕКСАНДРІВНА	Прикладна математика
КОНОНОВА	ІРИНА	ВІКТОРІВНА	Математика
ПАТРАШКОВА	ІРИНА	ІЛЛІВНА	Облік та аудит
ГУРЖІЙ	ВІТАЛІЙ	АНАТОЛІЙОВИЧ	Менеджмент організацій
УРСУЛОВА	НІНА	ВІТАЛІЇВНА	Фінанси
КЛЯГІНА	ОЛЬГА	ВІКТОРІВНА	Фінанси
ТОВПНЕЦЬ	ОЛЕКСАНДР	ОЛЕКСАНДРОВИЧ	Менеджмент організацій
НЕДОЛУГА	АНАСТАСІЯ	ВІКТОРІВНА	Екологія
ГЛАДИР	АЛЬОНА	ІГОРІВНА	Фінанси
ЧАВИЧАЛОВ	ОЛЕГ	ВОЛОДИМИРОВИЧ	Облік та аудит
ПАВЛЕНКО	НАТАЛІЯ	ПЕТРІВНА	Менеджмент організацій
КЛЮЧЕНКО	ЯНА	ОЛЕГІВНА	Менеджмент організацій
СОСНІНА	ОЛЕНА	ОЛЕКСАНДРІВНА	Менеджмент організацій
РИМАР	ЮЛІЯ	ПАВЛІВНА	Фінанси
ВЕЛИКИХ	ОКСАНА	ПЕТРІВНА	Економічна кібернетика
ГРИГОРСВА	ОЛЬГА	МИКОЛАЇВНА	Інформатика
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>

```

	MP JEP 23010-2003 "IT in University Management NETWORK"		IEDI-Ref-Arch-DR-10.doc
	Document	IEDI Reference Architecture Specification. Draft Recommendation	Version 1.0
	Topic	4 IEDI Reference Architecture - 4.1 Architectural Layering	28/02/2004

## 4 IEDI Reference Architecture

This section provides the descriptions of the overall organization and the critical constituents of the IEDI architecture:

- Architecture layers, components, clients and servers
- Ontologies
- Web services
- Wrappers
- Control and data flows

### 4.1 Architectural Layering

IEDI Architectural layering is defined according to the analysis of the IEDI processes and tasks and reflects the mediator type of IEDI architecture (refer to Fig. 10). 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?

#### 4.1.1 Architecture layers, Layer Components and User Roles

IEDI User Layer is the environment for AU-s and AU Clients (Section 4.1.2). IEDI IR Wrapper and IR Layers represent autonomous, heterogeneous, and distributed IR holders. 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. 10 and 11.

#### 4.1.2 IEDI Clients and Servers

The software components of IEDI may be 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 (Fig. 10) 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 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. 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
- Ontology Editor

MOE and IROE Clients interoperate with the following IEDI components: IEDI Mediator Access Server

IEDI Servers are the active components of IEDI Architecture which provide services to IEDI Clients. IEDI Servers are shown on Fig. 10 and 11. Their functions and components are described in detail further on in the Specification.

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>4 IEDI Reference Architecture - 4.1 Architectural Layering</b>	<b>28/02/2004</b>

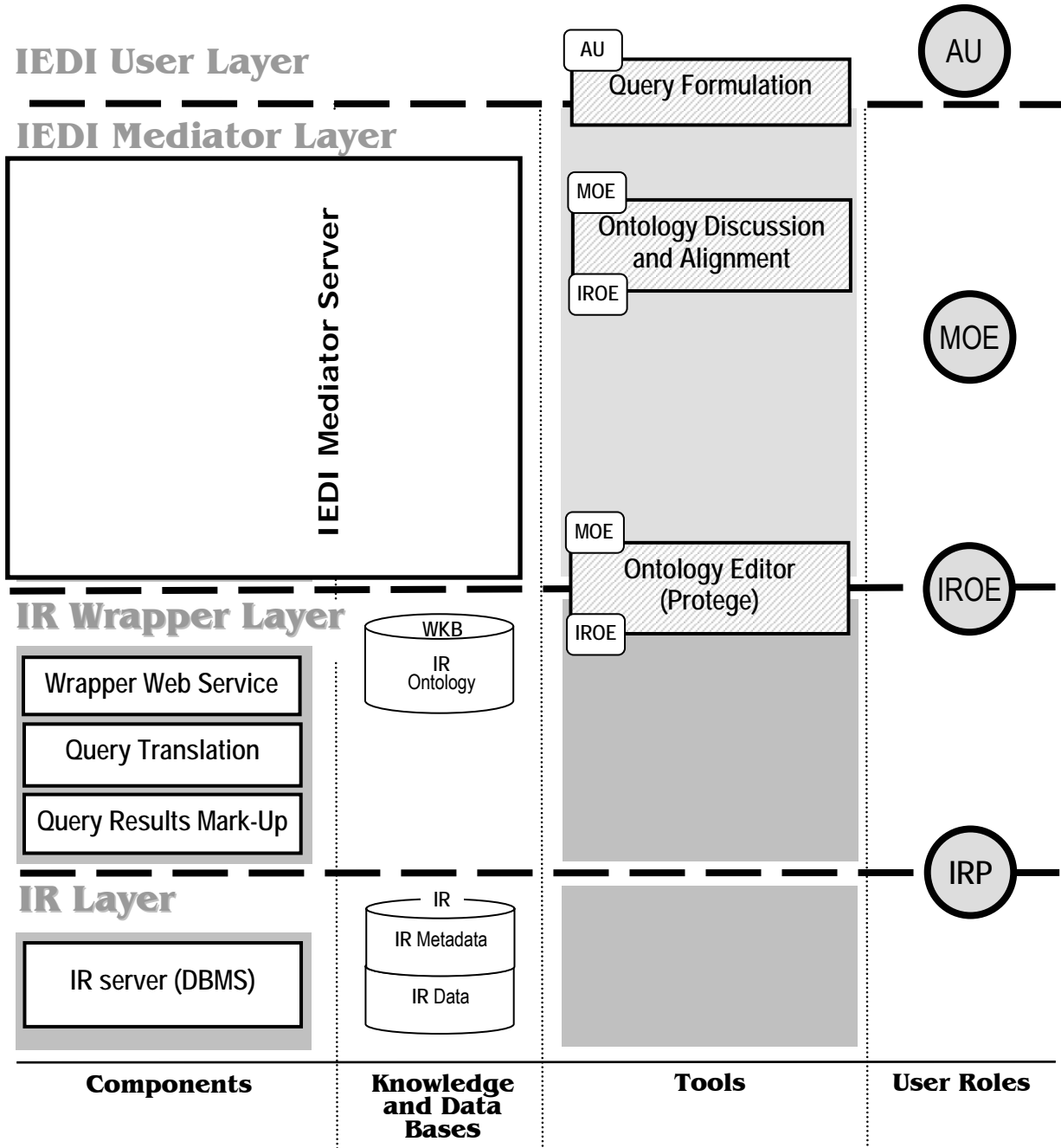



Fig. 10. IEDI architecture layers, layer components and user roles

	MP JEP 23010-2003 "IT in University Management NETWORK"		IEDI-Ref-Arch-DR-10.doc
	Document	IEDI Reference Architecture Specification. Draft Recommendation	Version 1.0
	Topic	4 IEDI Reference Architecture - 4.2 Ontologies	28/02/2004

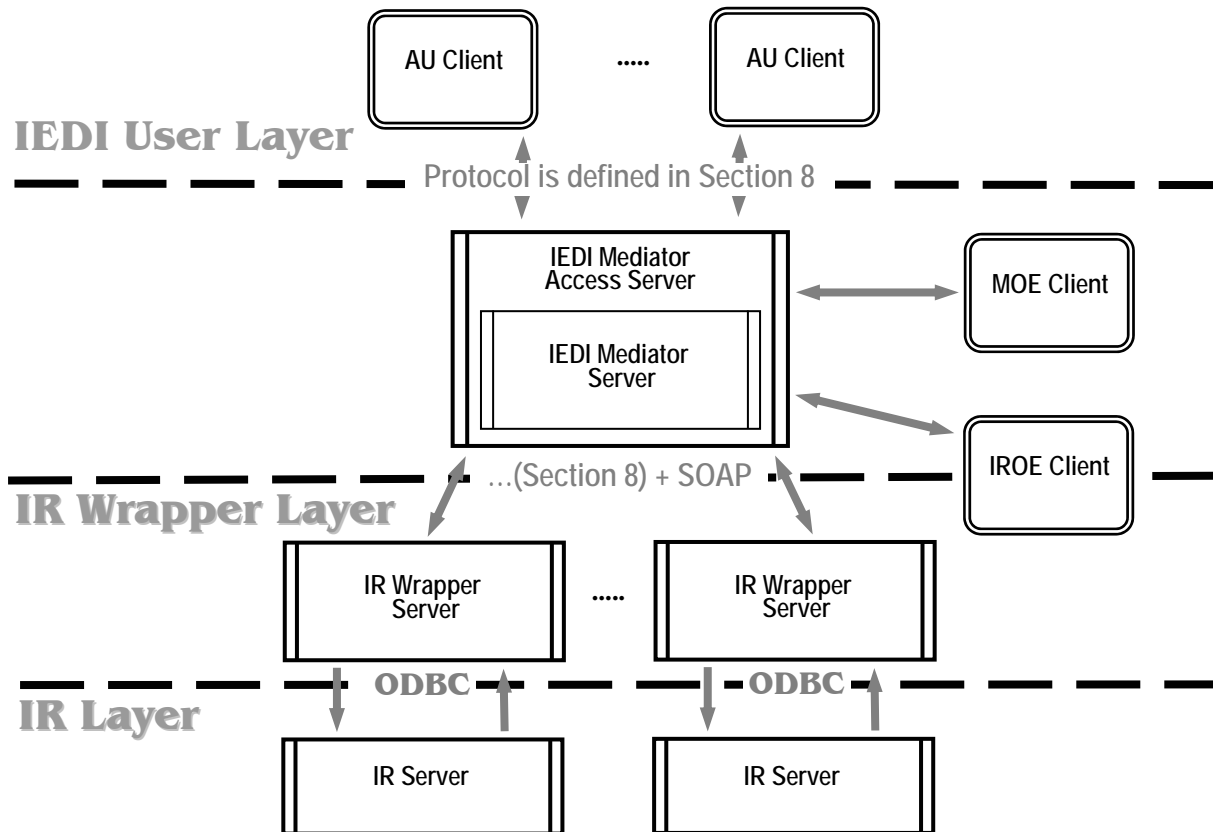


Fig. 11. IEDI Clients and Servers Layering

## 4.2 Ontologies

IEDI by its role is the distributed mediator system providing some kind of 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 in the form of ontologies at different layers of the architecture.


### 4.2.1 The Types of IEDI Ontologies

UNIT-NET 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** defines basic top-level elements. These elements according to their definitions are being 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 sense between mediator-side knowledge engineers and those from the resource-side. Top-level ontology allows any two ontologies within UNIT-NET to be comparable.

**Domain ontology** represents particular domain knowledge. There are several reasons to explore domain ontology in UNIT-NET Mediator. First one is that domain ontology provides UNIT-NET authorized users 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 user profile. Another reason is that domain ontology presents a vision of the community on the domain, and therefore plays an educational role.



	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>4 IEDI Reference Architecture - 4.2 Ontologies</b>	<b>28/02/2004</b>

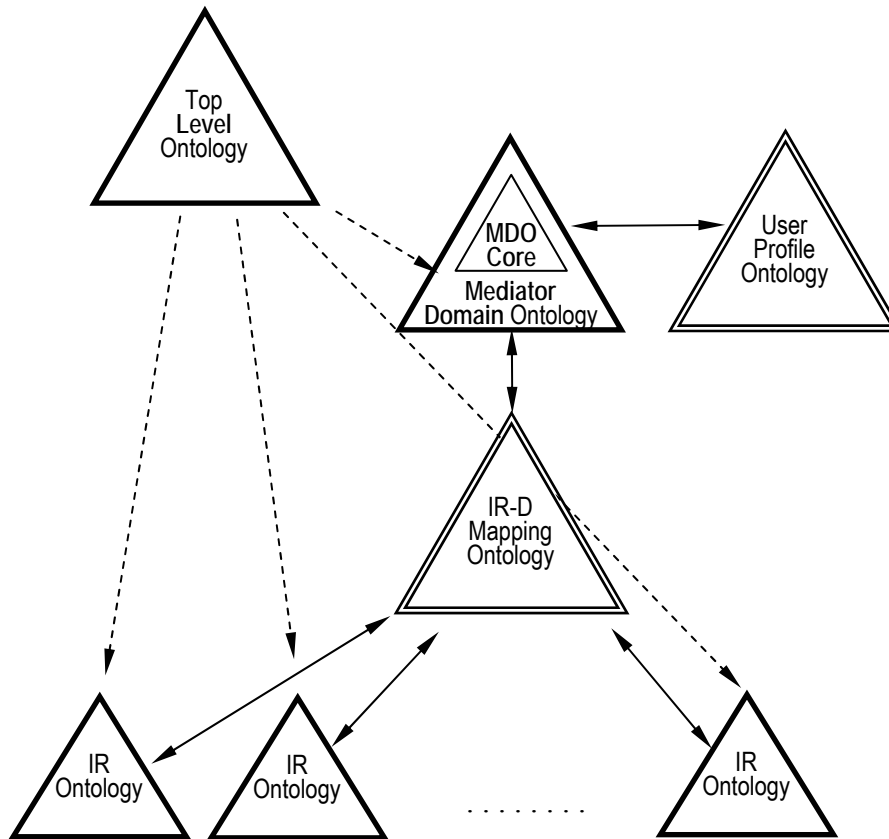


Fig. 12 IEDI Ontologies Hierarchy

**Resource ontology** 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 ontology** is an ontology, which stores knowledge on correspondences between concepts in two or more ontologies. Usually reference ontology keeps axioms on equivalence/subsuming between concepts/slots.

#### 4.2.2 Ontologies Hierarchy

IEDI ontologies are organized into the hierarchy (refer to Fig. 12).

The root of the ontologies' hierarchy is the top-level ontology (or upper-level ontology, ULO). According to the common understanding of how it should look like (see, e.g., DOLCE [DOL02]), the top-level ontology provides for concepts like object, time, time period, action and other ontological notions. It is very important to make definitions of these concepts be shared across UNIT-NET IEDI as far as the easy way to make any pair of ontologies comparable is to ensure both ontologies to share common basic concepts. Top-level ontology is not related to a particular domain; on the contrary, it should be cross-domain-oriented. It is designed or adopted at the mediator development stage.

The main ontology within the UNIT-NET mediator is the mediator domain ontology (MDO). Mediator domain ontology has the necessary part – MDO Core ontology. The content of MDO Core is the set of basic elements of the domain, which is agreed upon with a National Ministry as with the information user (e.g. Ukrainian Ministry of Education and Sciences). MDO Core ontology is defined as the minimal knowledge base, which has to be presented in any information resource registered at the UNIT-NET mediator. Mediator domain ontology extends MDO Core with elements imported from information resources' ontologies, thus it stipulates answering extended queries. Domain ontology is used in the processes of query answering, resource registration and changes adaptation. It is constructed by MOE.

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>4 IEDI Reference Architecture - 4.2 Ontologies</b>	<b>28/02/2004</b>

Basic sources for the MDO within the UNIT-NET IEDI mediator are the ontologies of information resources (IROs). Each IRO is constructed at the resource side by IROE independently from other resources and from mediator. The only requirements are (i) to provide elements for the whole MDO Core ontology and (ii) to relate elements of IRO to UNIT-NET top-level ontology. These requirements give a way to compare two independent IROs and guarantee answers to queries within the bounds of the MDO Core.

IRO construction process inspects resource data structure and additional information on data constraints, related to the features of particular resource. This process is almost manual. Only resource, possessing IRO, may register to mediator.

Within the ontology hierarchy there are two ontologies, which are the intermediates between autonomous resources, authorized users and MDO. These are IR-Domain Mapping ontology and User Profile ontology.

User profile reference ontology (UPRO) is a reference ontology, which collects the user personal knowledge on meanings of key words/phrases. This personal knowledge is represented as a semantic relationship between the given key word/phrase and the concept of the MDO. UPRO allows storing more than one different meanings of key word/phrase for particular users. UPO may be extended when authorized user poses a query, containing new (for this user UPRO part) key words/phrases, or changed when user decided to keep new meaning of his key word/phrase. More detailed description of communications between UNIT-NET IEDI mediator and AU see in Section 5.2 of this Specification.

User profile ontology MAY be used in the process of query formulation, when an input AU query is transformed to the query in the terms of MDO (see Section 5 of this Specification).

Besides, UPRO contains AU-specific authentication information pointing to the subset of MDO concepts which are available for the usage of this specific AU in his/her queries.

IR-Domain Mapping ontology (IRDMO) is a reference ontology, which keeps links between MDO and particular autonomous IROs. The reason to create and maintain such type of ontology is that each concept from the domain ontology may have several correspondent concepts in IROs. This information allows to extract sub-queries over particular resources from incoming user query, in terms of MDO and to execute these sub-queries independently at the resource side. IRDMO must contain mappings of all the MDO Core concepts and slots to all IROs. However, IRDMO may not have mappings of particular concepts from MDO for some information resource.

#### 4.2.3 Ontologies in IEDI Processes

Ontologies are used in processes of querying distributed autonomous semantically heterogeneous information resources (see Section 2.2, this Spec.), in registering new information resource (see Section 2.3, this Spec.), and in maintaining coherent semantic descriptions (see Section 2.4, this Spec.).


Table 2 provides the list of UNIT-NET IEDI processes and ontologies involved.

Processes	Ontologies	ULO	MDO Core	MDO	IRDMO	IRO	UPRO
		Querying distributed autonomous semantically heterogeneous information resources	--	R	R	R	R
Register new information resource		R	R	R/U	R/U	R	--
Maintaining coherent semantic descriptions		R	R/U	R/U	R/U	R/U	R/U

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

#### 4.2.4 Ontology Representation Language

Semantic Web oriented languages are chosen to represent all the ontologies within UNIT-NET IEDI mediator. Semantic Web is the paradigm for data and knowledge presentation in the World Wide Web, which explores the semantics of data as the markup for hypertext document. Known languages for the Semantic Web have rich set of elements for semantics description. They include RDF, RDFS, DAML+OIL, and finally – Web Ontology

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>4 IEDI Reference Architecture - 4.2 Ontologies</b>	<b>28/02/2004</b>

Language OWL. All the ontologies within UNIT-NET IEDI mediator and at the resources' side are presented in machine-readable form by means of ontology description language OWL DL. This choice is grounded with the following:

- OWL is developed specially for distributed ontologies, when parts of ontology are spread over the Web.
- OWL today is the most expressive language with bounded decidability procedure. It takes all the expressive means from DAML+OIL and RDF/RDFS;
- OWL may use known inference engines such as FaCT, RACER to check ontology correctness.

#### 4.2.5 IEDI Ontology Repository

IEDI Ontology Repository is distributed over the Infrastructure and, from the functional point of view, reflects the Ontologies Hierarchy presented in Section 4.2.2. Ontologies at Mediator Layer as well as the ontologies at the side of each IRP are stored in Knowledge Bases (KB). These KB-s are planned to be implemented with the help of Jena 2.0 [JENA04] API.

Jena is a Java-based framework for developing Semantic Web applications. It features:

- RDF API
- Statement centric methods for manipulating an RDF model as a set of RDF triples
- Resource centric methods for manipulating an RDF model as a set of resources with properties
- Cascading method calls for more convenient programming
- Built in support for RDF containers (bag, alt, and seq)
- Enhanced resources (resource properties may be extended by an application)
- Integrated parsers and writers for RDF/XML (ARP), N3 and N-TRIPLES
- Support for typed literals

Jena 2.0 functional components are as follows:

**ARP - Jena's RDF/XML Parser.** ARP aims to be fully compliant with the latest decisions of the RDF Core Working Group. Jena 2.0 version is compliant with the Editor's Working Drafts at time of release. ARP is typically invoked using Jena's read operations, but can also be used as a standalone component.

**Persistence Subsystem.** Jena 2.0 persistence subsystem implements an extension to the Jena Model class that provides persistence for models through the use of a back-end database engine. The persistence subsystem supports a Fastpath capability for RDQL queries that dynamically generates SQL queries to perform as much of the RDQL query as possible within an SQL database engine. Currently, Jena 2.0 can use three SQL database engines, MySQL, Oracle and PostgreSQL. These are supported on Linux and WindowsXP. Persistence subsystem is also designed to be portable to other SQL database engines.

**Reasoning Subsystem.** Jena 2.0 reasoning subsystem comprises a generic rule based inference engine together with configured rule sets for RDFS and for the OWL/Lite subset of OWL Full. The subsystem is designed to be extensible so that it should be possible to plug a range of external reasoners into Jena in future releases.

**Ontology Subsystem.** Jena 2.0 ontology API is intended to support programmers who are working with ontology data based on RDF. Specifically, this means support for OWL, DAML+OIL and RDFS. A set of Java abstractions extend the generic RDF Resource and Property classes to model more directly the class and property expressions found in ontologies using the above languages, and the relationships between these classes and properties. The ontology API works closely with the reasoning subsystem derive additional information that can be inferred from a particular ontology source. Jena 2.0 ontology subsystem also includes a document manager that assists with process of managing imported ontology documents.

**RDQL query language.** RDQL is a query language for RDF data. The implementation in Jena 2.0 is coupled to relational database storage so that optimized query is performed over data held in a Jena relational persistence store.

Given IEDI ontologies are stored and manipulated by the means of Jena 2.0 framework, the following convention on the ontology representation is approved for IEDI Ontology Repository. Ontology Knowledge bases will be represented and used in following formats:

- Text files in OWL format for exchange purposes, which will be parsed and saved by Jena ARP
- Jena API RDF model for performing queries, and for manipulating and managing ontologies' instances and properties

	MP JEP 23010-2003 "IT in University Management NETWORK"		IEDI-Ref-Arch-DR-10.doc
	Document	IEDI Reference Architecture Specification. Draft Recommendation	Version 1.0
	Topic	4 IEDI Reference Architecture -	28/02/2004

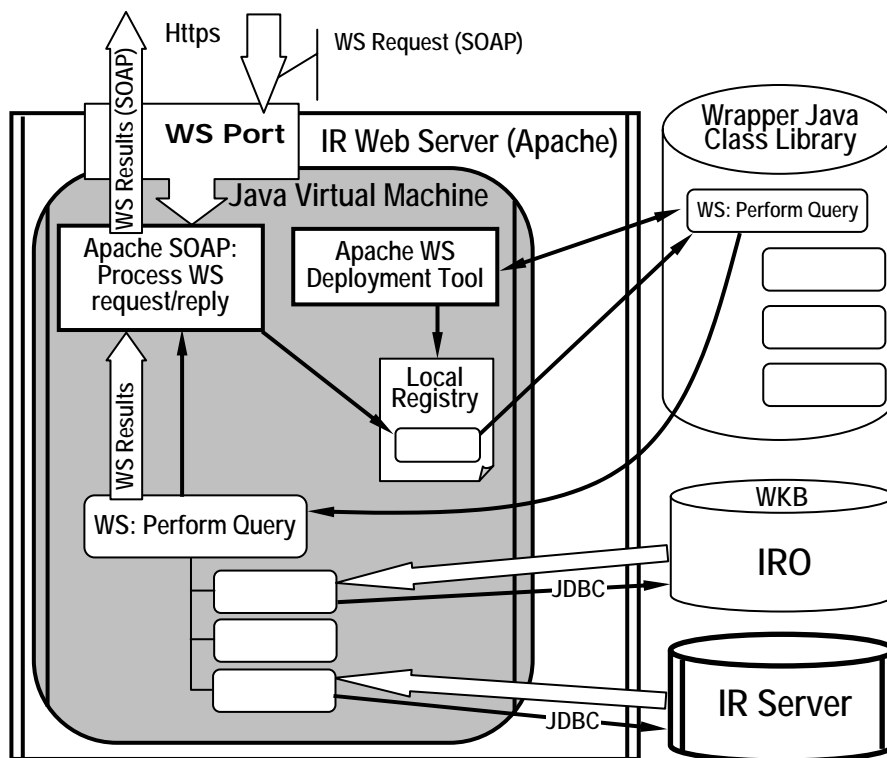


Fig. 13. IEDI Generic Wrapper Server Architecture.

- Persistent storage in Jena relational database (MySQL). This format is used by Jena API and is not directly accessed by other parts of system.

### 4.3 Perform Query Web Service

The task of the Wrapper Web Service is to perform the queries received from the IEDI Mediator. IEDI Wrapper web service, together with its sub-ordinate functional components, are implemented as Java classes compiled to byte code and are executed by JVM of the Wrapper Web Server. Tomcat Web Server is chosen for the prototype implementation. The architecture of the Generic IR Web Server and the process of WS execution is presented on Fig. 13.

Before a Wrapper Web Service (WS) could be executed by JVM it should be deployed by Apache SOAP WS Deployment Tool. The deployment results in placing the code of the web service class(es) to the Wrapper Java Class Library classes compiled to byte code and the registering of the WS at the Local WS Registry.

The requests to perform a WS are conveyed by means of the secured protocol (please refer to Section 8 for more details). When a SOAP request to perform a WS comes to the WS port of the Wrapper Web Server it is processed by the Apache SOAP processing service. The SOAP processing service extracts WS information from the SOAP envelope, checks if the requested service is available at the local registry and invokes the execution of the WS at the JVM. The components of the Perform Query Web Service are described in the Section 4.4. The components interact with the Wrapper Knowledge Base (WKB) and with the IR by the means of JDBC.

IEDI Mediator needs the information about the available IR Wrapper Web Services to properly address the queries to the corresponding IR-s. This information contains the URI of the WS, is provided when an IR is registered to the IEDI Mediator, and is stored in the IRDMO.

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>4 IEDI Reference Architecture - 4.4 IR Wrappers</b>	<b>28/02/2004</b>

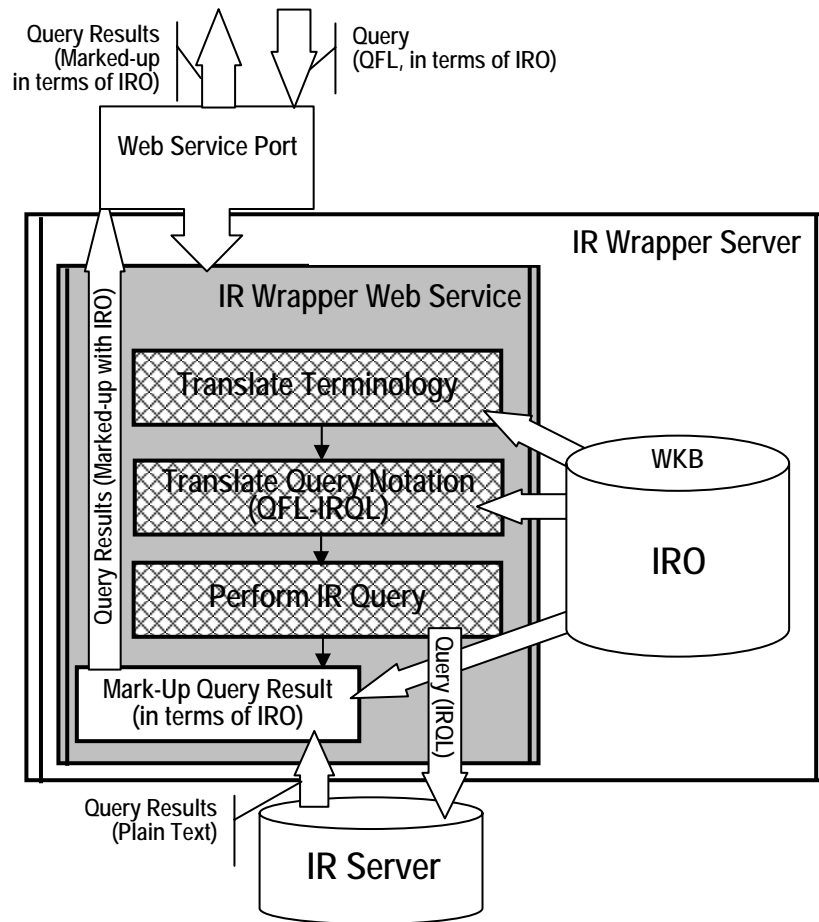


Fig. 14. IEDI Generic Wrapper Architecture.

## 4.4 IR Wrappers

IEDI IR Wrappers provide uniform access to registered IR-s. The wrappers for the specific IR-s are designed and deployed according to IEDI Wrapper Specification [IEDI-RWS04]. IEDI Generic Wrapper provides the software pattern for a specific IR Wrapper design.

### 4.4.1 IEDI Generic Wrapper

IEDI Generic Wrapper is the architectural abstraction and the software pattern for constructing and deploying IR Wrappers in the process of the Preparation to the IR Registration (refer to Section 2.3). A wrapper main function is to provide the framework for the uniform access to the respective IR. The implementation of this function comprises:

- Provide the web service as the interface to query the IR by IEDI Mediator
- Perform the translation of the query in the terms of the IR Ontology to the query in the terms of the IR schema (if there is the schema: for example, the IR is the Relational Data Base)
- Perform the translation of the query in IEDI Query Formulation Language (QFL) [IEDI-QFLS04] to the query in the notation of the IR Query Language (IRQL)
- Query the IR
- Mark-up the result of the query in the terms of the IRO

IEDI Generic Wrapper architecture is shown on Fig. 14. It comprises both IR invariant and IR Specific components. The latter are later referred to as IR Wrapper Bindings. IEDI Generic Wrapper implementation provides the IR invariant components and the skeletons for IR Wrapper Bindings.

	MP JEP 23010-2003 "IT in University Management NETWORK"		IEDI-Ref-Arch-DR-10.doc
	Document	IEDI Reference Architecture Specification. Draft Recommendation	Version 1.0
	Topic	4 IEDI Reference Architecture - 4.5 Control Flows	28/02/2004

#### 4.4.2 IR Wrapper Bindings

IR Generic Wrapper implementation MUST provide the software pattern of the architectural components shown in Fig. 14. Specific wrappers implemented for specific IR-s SHOULD inherit the Generic Wrapper components. A specific IR wrapper is designed by choosing (or designing) the appropriate instantiation (binding) of IR specific wrapper components:

- Translate Terminology
- Translate Query Notation
- Perform IR Query

For example, if an IR is the Relational Data Base managed by MS SQL Server the following component bindings should be chosen:

- Translate Terminology (IRO to IR DB Metadata Elements)
- Translate Query Notation (QFL to MS SQL)
- Perform IR Query (ODBC, MS SQL Server)

It is supposed that these resource specific components are developed in the conformance with this Specification, the IEDI IR Wrapper Specification [IEDI-RWS04], and are collected in the IEDI IR Wrapper Bindings Library in the frame of the Open Source Licensing principles.

#### 4.5 Control Flows

This section provides the Control Flow Diagrams for the tasks of:

- IR registration to IEDI Mediator (Fig. 15)
- Maintenance of the coherence in MDO and IRO (Fig. 16)
- IEDI query processing (Fig. 17)

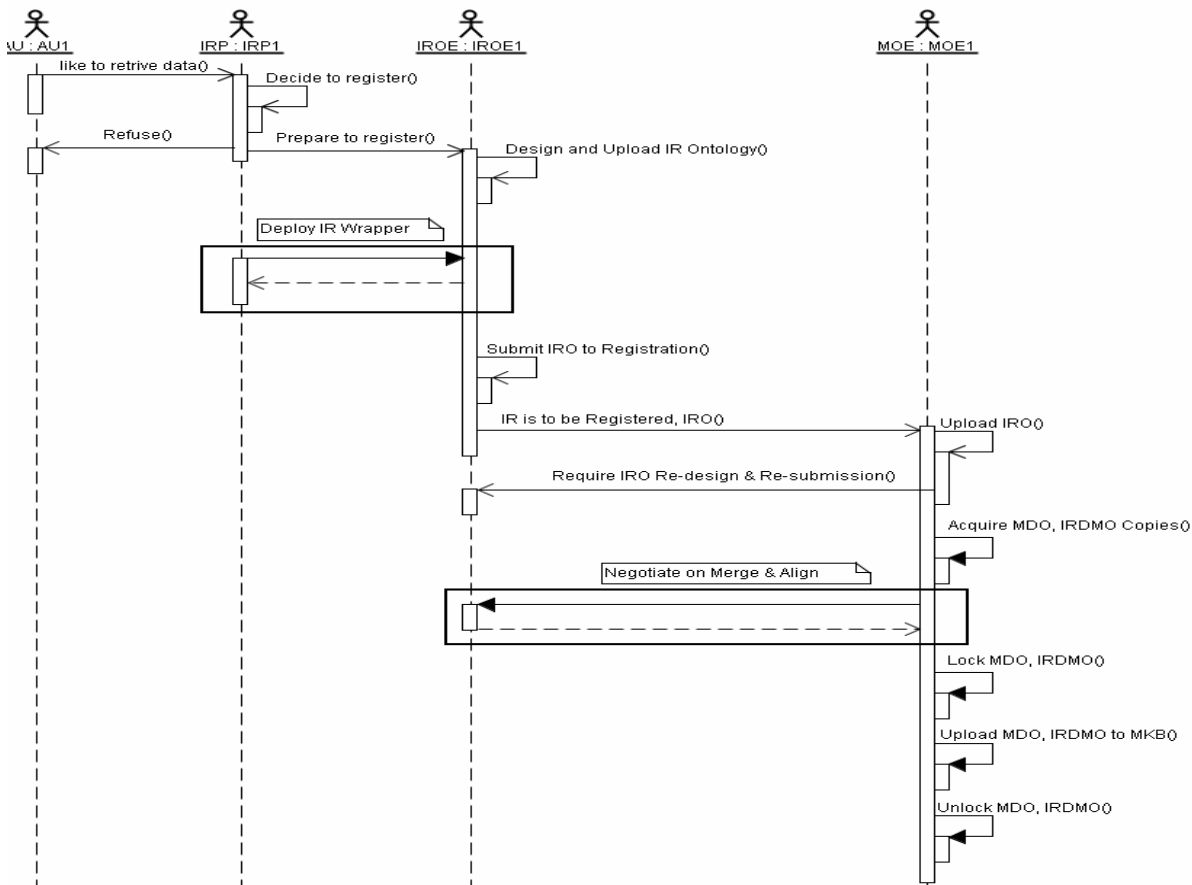



Fig. 15. Control flow for IR registration.

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>4 IEDI Reference Architecture - 4.5 Control Flows</b>	<b>28/02/2004</b>

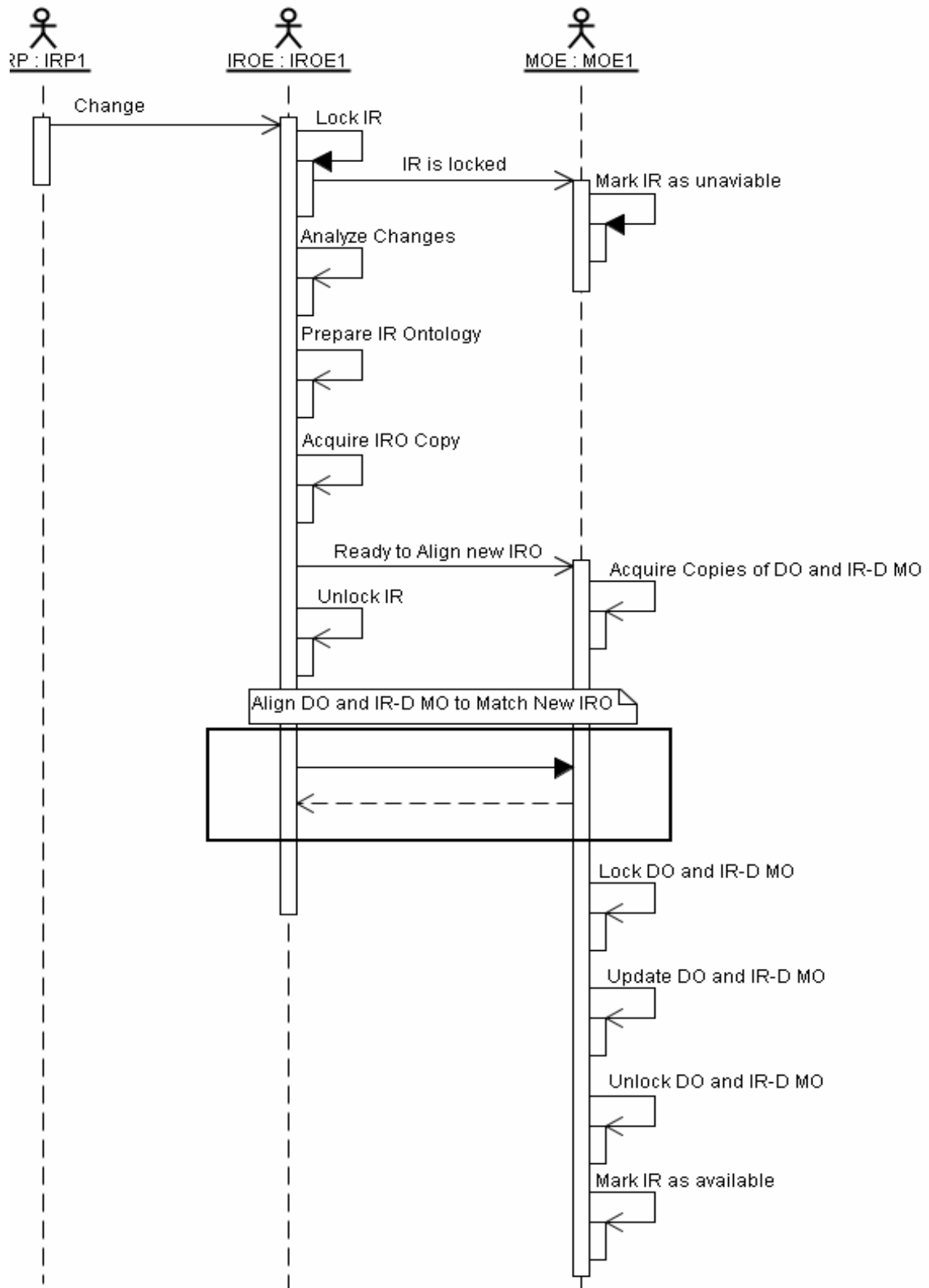



Fig. 16. Control flow for ontology coherence maintenance.

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>4 IEDI Reference Architecture - 4.5 Control Flows</b>	<b>28/02/2004</b>

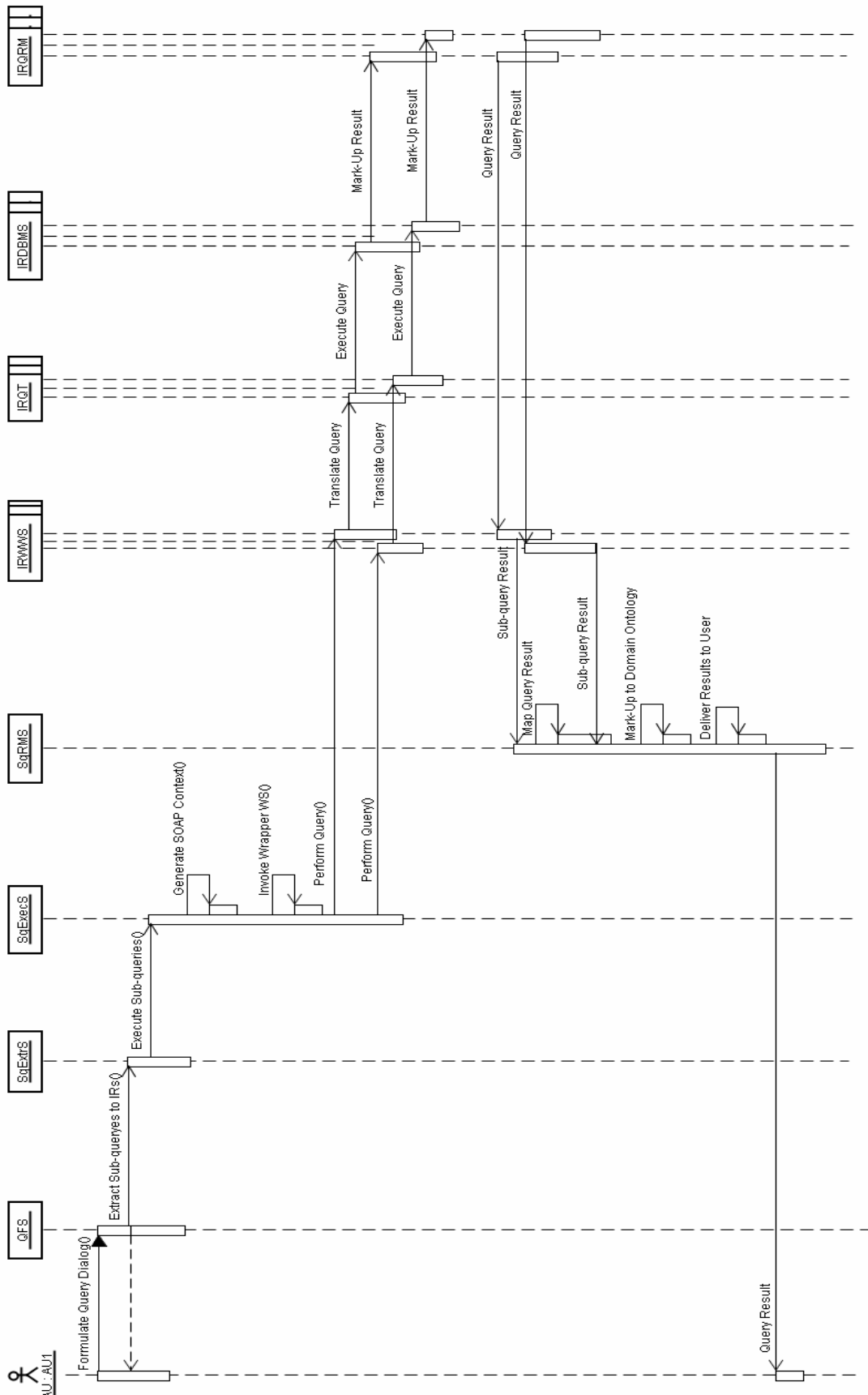


Fig. 17. Control flow for IEDI query processing.



	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>4 IEDI Reference Architecture - 4.6 Data Flows</b>	<b>28/02/2004</b>

## 4.6 Data Flows

This section provides the Data Flow Diagrams for the tasks of:

- IR registration to IEDI Mediator (Fig. 18)
- Maintenance of the coherence in MDO and IRO (Fig. 19)
- IEDI query processing (Fig. 20)

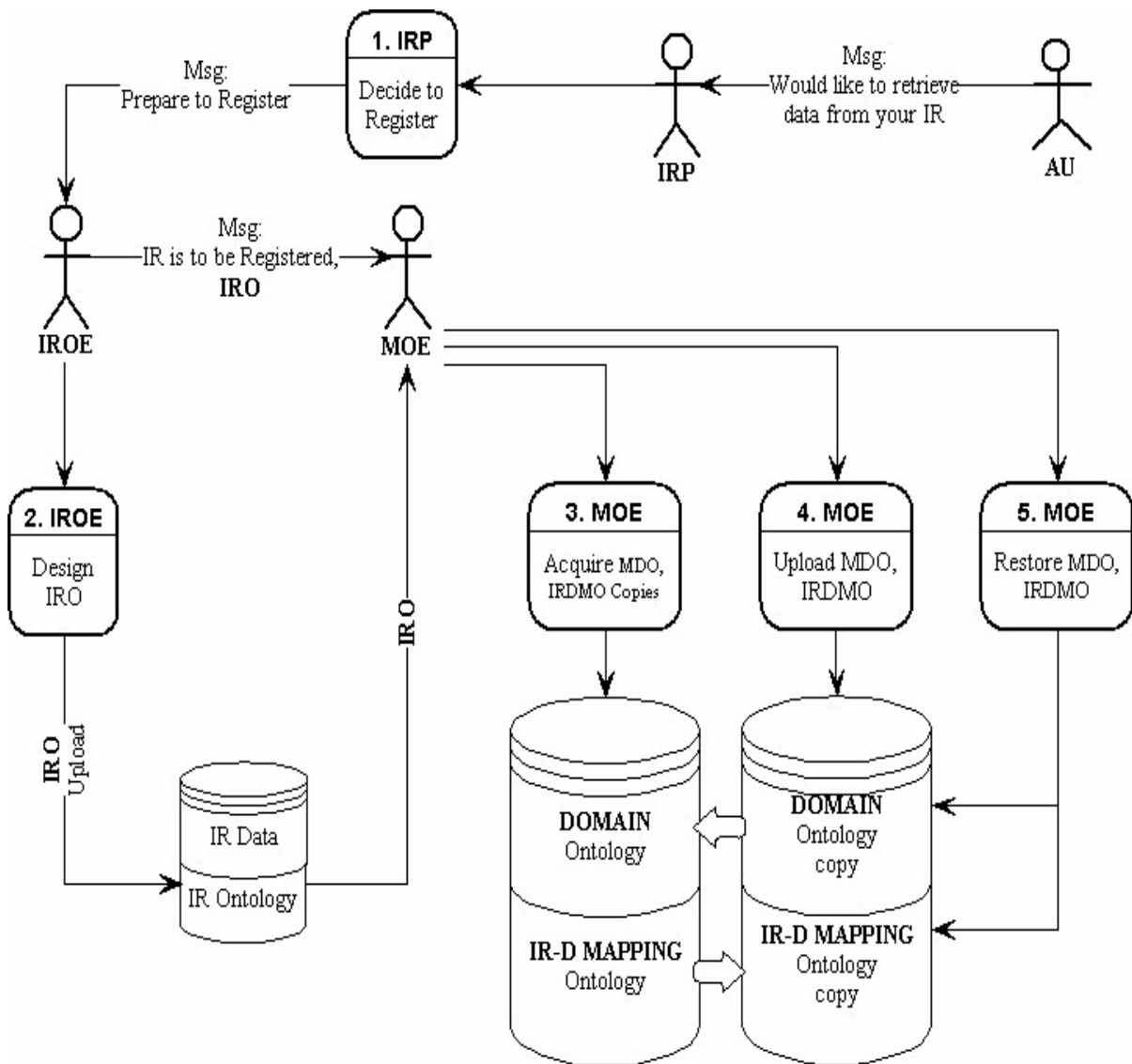


Fig. 18. Data flow for IR registration.

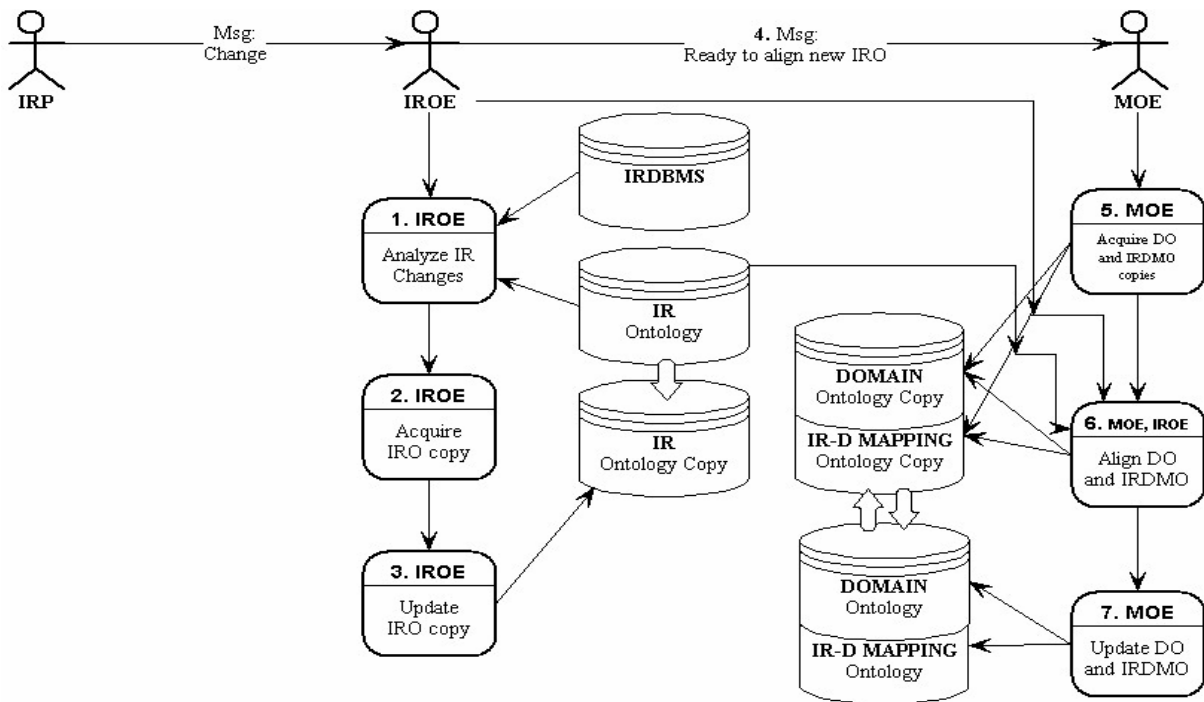


Fig. 19. Data flow for ontology coherence maintenance.

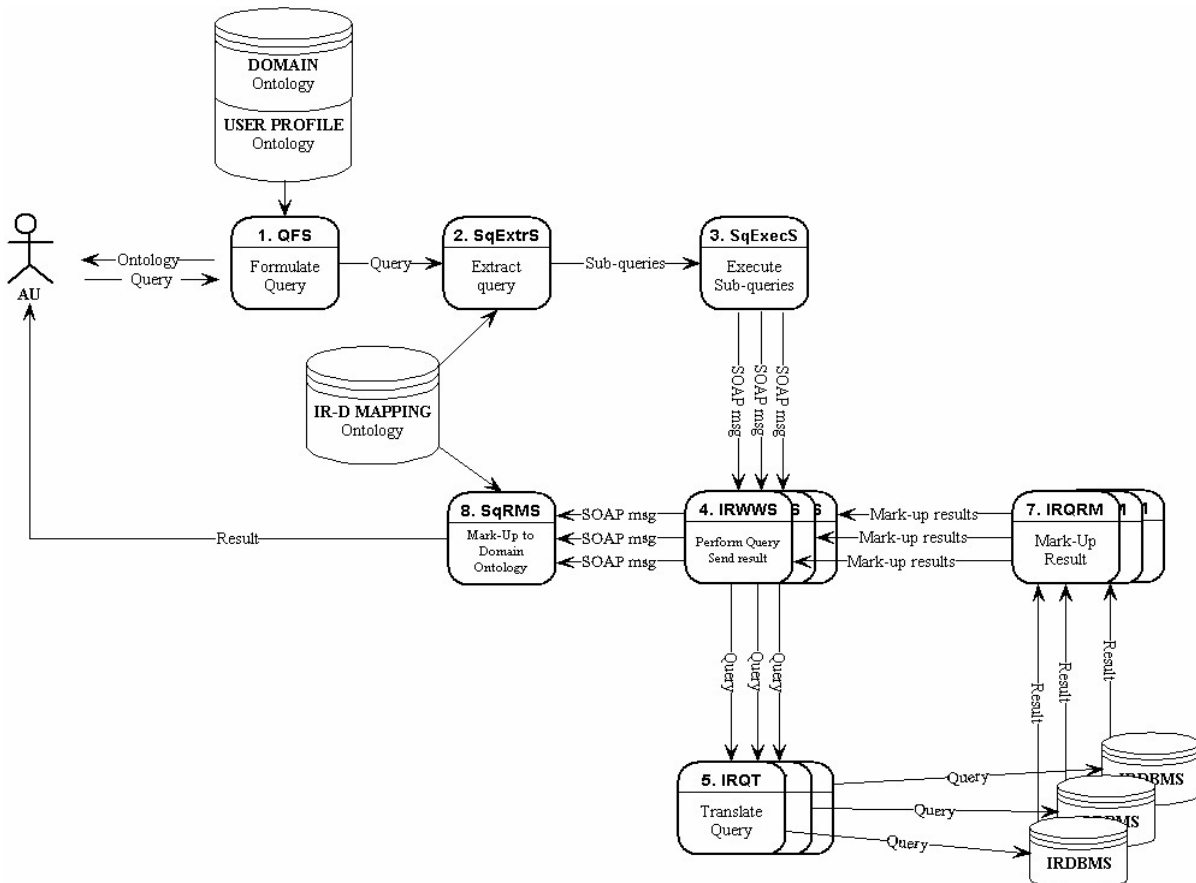


Fig. 20. Data flow for IR registration.

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>5 Query Formulation - 5.1 Requirements to the Query Formulation Tool Interfaces</b>	<b>28/02/2004</b>

## 5 Query Formulation

AU queries are formulated in the terms of MDO and eventually in the notation of IEDI QFL. RDQL is so far chosen as the QFL for the research prototype implementation and for the evaluation purposes. In IEDI it is not required that an AU is able to code his or her queries in QFL notation. Instead, the tool for ontology-driven Query Formulation Tool (QFT) is provided by the IEDI mediator. It is also required that an AU approves the formulated query before it is actually processed (refer to Section 2.2).

### 5.1 Requirements to the Query Formulation Tool Interfaces

The general requirement to QFT is that an AU SHOULD have a visual ontology-based interface which allows him to interactively choose the terms of the MDO, to apply constraints to these chosen terms and, thus, to formulate the query. The query in QFL SHOULD be generated automatically from the sequence of the AU ontology term choices and the constraints applied by the AU to the selected ontology terms.


The blueprint of the QFT interface layout is given in Table 3. The walkthrough example is used to illustrate the possible contents:

***Retrieve the list of the 1-st year students who have received maximum 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).***

Table 3

Query Formulation Interface

<p><b>MDO browse area</b></p> <p><b>Person</b></p> <p><i>Parents</i></p> <p><i>Properties</i></p> <p>last_name</p> <p>second_name</p> <p>first_name</p> <p><b>PersonOnSpeciality</b></p> <p><i>Properties</i></p> <p>n_group</p> <p>exam_passes type <b>Exam</b></p>	<p><b>Selected ontology terms</b></p> <p>[view] [remove] Person.first_name [report [ ]</p> <p>[view] [remove] Person.second_name [report [ ]</p> <p>[view] [remove] Person.last_name [report [ ]</p> <p>[view] [remove] Speciality.specialityName [report [ ]</p> <p>[view] [remove] Exam.exam_title [report [ ]</p> <p>[view] [remove] PersonOnSpeciality.exams_passes</p> <p>[view] [remove] Exam.exam_title</p> <p>[view] [remove] Exam.exam_type</p> <p>[view] [remove] EntrantExam.entrant_grade</p> <p>[view] [remove] SessionExam.session_grade</p> <p>[view] [remove] SessionExam.semester_num</p> <p>[view] [remove] PersonOnSpeciality.on_spec</p> <p>[view] [remove] Speciality.spec_name</p> <p>[view] [remove] Exam.examOnDiscipline</p> <p>[view] [remove] Discipline.disciplineName</p>
<p><b>Constraints</b></p> <p>Selected item's DataType related constraint interface</p> <p><b>Item:</b> Exam.exam_type</p> <p><b>DataType:</b> <a href="http://www.w3.org/2001/XMLSchema#string">http://www.w3.org/2001/XMLSchema#string</a></p> <p><b>Constraint Name:</b> examType1</p>	<p><b>RDQL Query Code</b></p> <pre>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),</pre>

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>5 Query Formulation - 5.2 Interaction with an AU</b>	<b>28/02/2004</b>

<b>Constraint Type:</b> eq (equal)	<pre>(?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") AND (?sessionGrade eq "2") AND (?semesterNum eq "1")  USING stud FOR &lt;MDO-URL#&gt;</pre>
<b>Constraint Value:</b> "Exam"	

## 5.2 Interaction with an AU

An AU browses the MDO in **MDO browse area** and is able to select ontology elements to be added to the query. The selected element appears in **Selected elements list area**. MDO structure is displayed as a tree. The root element of the displayed ontology tree is current element, for example **Person** (refer to Table 3). An ontology element may have subitems of several types associated with it. If **A** is the current ontology element its' associated items are:

- **Parents List** – **A** is in rdfs#subClassOf relationship with the elements of the Parents List
- **Properties list** – elements which are in rdfs#domain relationship with **A**
- **Children list** – elements which are in rdfs#subClassOf relationship with **A**.

Special types (Parents, Properties, types) are highlighted with italic.

If element **A** is in rdfs#range relationship with some other element **B** then **A** will be accompanied by *type* keyword and the highlighted name of **B**. Elements highlighted bold can be selected by clicking. The selected element becomes the root of the ontology fragment visible in the MDO browse area.

In **Selected elements list area** an AU accumulates the MDO elements which will be used in his or her query. The following buttons indicate respective actions which may be performed on these ontology elements:

- [view] – show the chosen element as the root of the MDO fragment in MDO browse area. The constraints for the selected ontology element will be also shown in the **Element Constrains area**. An AU may then set or update the constraints on the properties of the chosen ontology element.
- [remove] – remove the chosen ontology element from the **Selected elements list area** and therefore from the query

The report checkbox, if checked, indicates that the resulting query should return the instances of the selected ontology element in the query results

**Element Constraints area** is the placeholder for the dialogue elements which allow to set constraints to ontology elements. These dialogs differ from each other according to the DataType of the ontology element and form the library of Constraint Dialogues.

**RDQL preview area** shows the preview of the current RDQL query.

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>6 Query Decomposition - 6.1 Extracting Sub-Queries for Different IR</b>	<b>28/02/2004</b>

## 6 Query Decomposition

User query is formulated in terms of MDO by means of Query Formulation Language (Section 9). To obtain results from resources this query should be decomposed into a set of sub-queries. A sub-query is denoted as a correct query in the terms of the particular IRO. Hence, each sub-query **MUST** be forwarded to the corresponding IR Wrapper and then processed by the IR Wrapper.

### 6.1 Extracting Sub-Queries for Different IR

The process of sub-queries extraction is performed in accordance with the following principles:

- *All the concepts enumerated in the user query and their requested properties **MUST** be presented in the query result.* This principle stands for the *stability* of MKB and WKB Bases at the period from the moment when an AU approves the query till the moment when the Mediator delivers the query results to the AU.
- *For each requested concept/property in the AU query in the terms of MDO there must be found corresponding concepts/properties in at least one of registered IRO-s.* This principle denotes the *completeness* of the decomposition procedure.

The process uses the knowledge stored in the IRDMO (see Section 4.2). IRDMO consists of two parts: Concept Mapping and Slot Mapping. Concept Mapping captures **equivalence relationships** between concepts from the MDO and concept(-s) in IRO-s only. Each concept may have several corresponding concepts in different IRO-s, as well as any concept may match several different concepts even within one IRO. Slot Mapping stores **equivalences** between slots of each concept in MDO and slots of concept (-s) in IRO-s.

Initial query and the sub-queries over particular IRO-s are presented in the same language – QFL (RDQL so far). QFL query **MUST** consist of the following clauses: SELECT, FROM, WHERE, AND, USING.

- **SELECT** clause determines variable names for required instances. **SELECT** clause identifies the variables to be returned in query result(-s).
- **WHERE** clause determines conditions, which should be satisfied. **WHERE** clause specifies the graph of an ontology fragment equivalent to the semantics of the query as a list of triple patterns. This set of triples is further on used as the query pattern.
- **AND** clause specifies the constraints – boolean expressions to be fulfilled in the resulting graph.
- **USING** clause is used for abbreviation to shorten the length of URIs.

Consider the following RDQL query example and the comments to the parts of this query (Table 4).

Table 4.

RDQL Query Parts

Query	Comments
<b>SELECT</b> ?fn, ?sn, ?ln, ?entrantExTitle <b>WHERE</b> (?x, stud:first_name, ?fn), (?x, stud:second_name, ?sn), (?x, stud:last_name, ?ln), (?x, stud:exams_passes, ?y), (?y, stud:exam_title,?entrantExTitle), (?y, stud:semesterNum,?semesterNum), <b>AND</b> (?semesterNum == "1") <b>USING</b> stud <b>FOR</b> <the URI of the IEDI MDO>	<i>?fn, ?sn, ?ln, ?semesterNum, ?entrantExTitle</i> – variable names, chosen arbitrarily for internal use  <i>first_name, second_name, last_name</i> – the names of slots in MDO  <i>stud</i> – is a prefix to shorten the full path to the MDO.

#### 6.1.1 Sub-queries extraction algorithm


Input: correct RDQL Query (here and below – MDO Query), satisfying current MDO state.

Output: a set of  $m$  correct RDQL Queries (here and below – IRO[ $m$ ] Queries), satisfying current IRO[ $m$ ] states.

Pre-conditions: ---

Function: Sub-queries Extraction.

Post-effects: ---

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>6 Query Decomposition - 6.1 Extracting Sub-Queries for Different IR</b>	<b>28/02/2004</b>

The following algorithm performs the terminological mapping from the terms of MDO to the terms of the IRO[ $i$ ], where  $1 \leq i \leq m$  stands for  $i$ -th IRO, possessing the terms from the input query. Value of  $m \leq n$ , where  $n$  – the no of IR-s, currently registered to the Mediator.

Denote a triple as  $\langle\langle \text{subj} \rangle, \langle \text{predicate} \rangle, \langle \text{obj} \rangle\rangle$ .

### 1. Preliminary grouping

Collect all the triples from the WHERE section of the MDO Query with the same  $\langle \text{subj} \rangle$ -part into the groups.

### 2. Find determining concepts

For each triples group taken from (1) get from the  $\langle \text{predicate} \rangle$ -part corresponding slot names and find in MDO determining concept, to which all these slots are attached by means of the OWL “domain” property.

Comments on existence and unicity of the determining concept see below.

### 3. Concept mapping

For each triples group from (1) use IRDMO for mapping of each determining concept from MDO Query into the equivalent concept from the IRO[ $i$ ]. Group concepts belonging to the same IRO[ $i$ ], where  $i = 1, \dots, n$ . Exclude from consideration empty groups, i.e. groups, which correspond to the IRO[ $i$ ] and do not possess any determining concept requested in the MDO Query.

### 4. Slot mapping

Make  $m$  copies of the MDO Query, where  $m \leq n$ .

For  $i = 1, \dots, m$

For each triples group taken from (1)

Replace for each triple in the group slot name in the  $\langle \text{predicate} \rangle$ -part of each triple with unique equivalent slot name, belonging to the IRO[ $i$ ] and attached to the concept, which is the mapping of the determining concept for this group of triples.

Remove triples groups, which do not have corresponding mapping for the determining concept of the group.

### 5. IRO[ $m$ ] Query clarification – making the query connected RDQL graph.

For  $i = 1, \dots, m$

For each triples group taken from (1)

Remove from the group triples, tho  $\langle \text{obj} \rangle$ -part is referencing to the triple from non-existing group

### 6. IRO[ $m$ ] Query clarification – forming SELECT-section

For  $i = 1, \dots, m$

For each variable from the SELECT-section

Check if the variable belongs to the one of the remaining groups of triples.

**If** there are zero groups at which this variable appears

(Neither in the  $\langle \text{subj} \rangle$ -part, nor in the  $\langle \text{obj} \rangle$ -part)

**Then** exclude this variable name from the SELECT section

### 7. IRO[ $m$ ] Query clarification – forming AND-section

For  $i = 1, \dots, m$


For each variable from the AND-section

Check if the variable belongs to the one of the remaining groups of triples.

**If** there are zero groups at which this variable appears

(Neither in the  $\langle \text{subj} \rangle$ -part, nor in the  $\langle \text{obj} \rangle$ -part)

**Then** exclude this variable name from the AND section

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>6 Query Decomposition - 6.2 Sub-Query Extraction by a Walkthrough Example</b>	<b>28/02/2004</b>

At the second step of the algorithm it is assumed that for each triples group exists unique determining concept. The reasons to make the assumption are as follows. First – the determining concept exists for each triples group. Otherwise in MDO Query will be slot, which is absent in the MDO. Second – the determining concept is unique for each triples group. Otherwise if in the MDO exist 2 or more concepts possessing the same slot sets (this is possible for subclasses and super classes). In this case algorithm should take the highest concept in the hierarchy, which possesses all the slots from the set. Sub-query extraction algorithm is illustrated by Fig. 21.

## 6.2 Sub-Query Extraction by a Walkthrough Example

Now consider the algorithm on the walkthrough example. The query in the natural language is as follows:

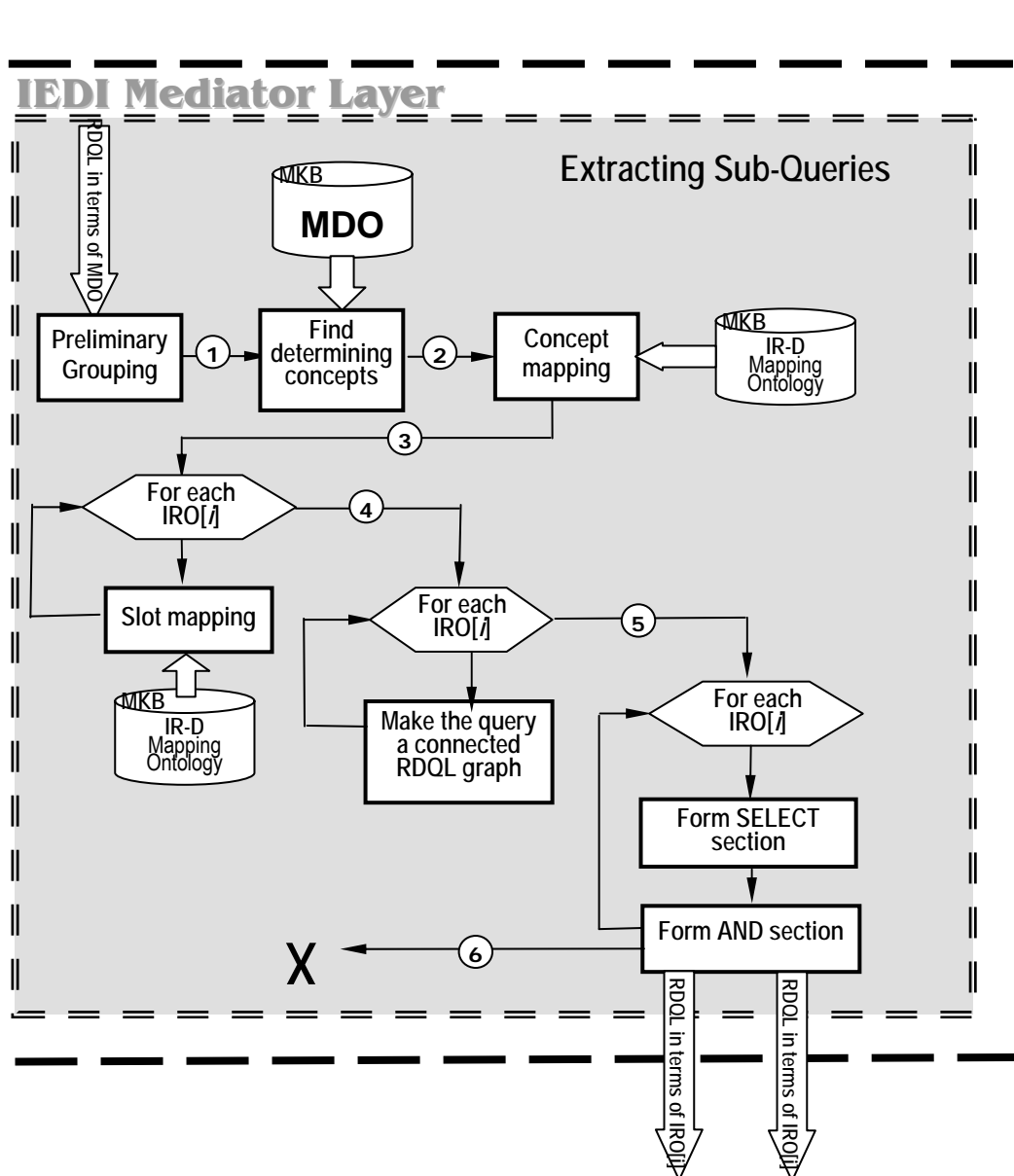



Fig. 21. Graphical illustration for the Sub-query extraction algorithm

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>6 Query Decomposition - 6.2 Sub-Query Extraction by a Walkthrough Example</b>	<b>28/02/2004</b>

**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)  
Display Student's Name, Second Name, Surname, his Speciality and the Name of the Course.**

Corresponding MDO Query is 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),
  (?x, stud:exams_passes, ?z), (?x, stud:on_spec, ?a),
  (?y, stud:exam_title, ?entrantExTitle), (?y, stud:exam_type, ?examType1),
  (?y, stud:entrant_grade, ?entrantGrade), (?y, stud:examOnDiscipline, ?r1),
  (?z, stud:exam_title, ?sessionExTitle), (?z, stud:exam_type, ?examType2),
  (?z, stud:session_grade, ?sessionGrade), (?z, stud:semesterNum, ?semesterNum),
  (?z, stud:examOnDiscipline, ?r2),
  (?a, stud:specialityName, ?specialityName)
  (?r1, stud:disciplineName, ?entrDiscName), (?r1, stud:includes, ?i1),
  (?r2, stud:disciplineName, ?sessionDiscName), (?r2, stud:includes, ?i2),
  (?i1, stud:disciplineName, ?discName1), (?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")
AND (?sessionGrade eq "2")
AND (?semesterNum eq "1")
USING stud FOR <MDO-URL#>
```

After the first step (**Preliminary grouping**) and second step (**Find determining concept**) the result will be as in the Table 5.

Table 5 Preliminary grouping and determining concepts detection.

Determining concepts	MDO Query
PersonOnSpeciality	<pre>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),   (?x, stud:exams_passes, ?z),   (?x, stud:on_spec, ?a),   (?y, stud:exam_title, ?entrantExTitle),   (?y, stud:exam_type, ?examType1),   (?y, stud:entrant_grade, ?entrantGrade),   (?y, stud:examOnDiscipline, ?r1),   (?z, stud:exam_title, ?sessionExTitle),   (?z, stud:exam_type, ?examType2),   (?z, stud:session_grade, ?sessionGrade),</pre>
EntrantExam	



	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>6 Query Decomposition - 6.2 Sub-Query Extraction by a Walkthrough Example</b>	<b>28/02/2004</b>

SessionExam	(?z, stud:examOnDiscipline, ?r2), (?a, stud:specialityName, ?specialityName) (?r1, stud:disciplineName, ?entrDiscName), (?r1, stud:includes, ?i1), (?r2, stud:includes, ?i2), (?r2, stud:disciplineName, ?sessionDiscName), (?i1, stud:disciplineName, ?discName1), (?i2, stud:disciplineName, ?discName2)
Speciality	
Discipline	AND (?examType1 eq "Exam"), (?examType2 eq "Exam")
Discipline	AND (?entrDiscName eq "Mathematics"), (?sessionDiscName eq "Mathematics")
Discipline	AND ((?entrantExTitle eq ?discName1)    (?sessionExTitle eq ?discName2))
Discipline	AND ((?sessionExTitle eq "Linear Algebra")    (?sessionExTitle eq "Mathematical Analysis"))
	AND (?entrantGrade eq "5")
	AND (?sessionGrade eq "2")
	AND (?semesterNum eq "1")
	USING stud FOR <MDO-URL#>

After the third step (**Concept mapping**) the result will be as follows in the Table 6

Table 6. Concept mapping.

<b>MDO Query</b> <b>Determining concepts</b>	<b>IRO[1] – Entrant</b>	<b>IRO[2] – Faculty</b>
PersonOnSpeciality	ProfilesAbo	StudentOnSpec
EntrantExam	EntrantExam	-
SessionExam	-	SessionExam
Speciality	Speciality	Speciality
Discipline	Discipline	-

After the forth step (**Slot mapping**) the result will be as follows in the Table 7.

Table 7 Slot mapping.

<b>IRO[1] – Entrant</b> <b>Intermediate query</b>	<b>IRO[2] – Faculty</b> <b>Intermediate query</b>
SELECT ?firstName, ?secondName, ?lastName, ?specialityName, ?sessionExTitle  WHERE (?x, abo:aboName, ?firstName), (?x, abo:secondName, ?secondName), (?x, abo:surname, ?lastName), (?x, abo:passes, ?y), * (?x, abo: passes, ?z), (?x, abo:AboSpec, ?a), (?y, abo:EntrantExamName, ?entrantExTitle), (?y, abo:examType, ?examType1), (?y, abo:grade, ?entrantGrade),	SELECT ?firstName, ?secondName, ?lastName, ?specialityName, ?sessionExTitle  WHERE (?x, stud:name, ?firstName), (?x, stud:secondName, ?secondName), (?x, stud:surName, ?lastName), * (?x, stud:examPasses, ?y), (?x, stud:examPasses, ?z), (?x, stud:onSpec, ?a), (?z, stud:examName, ?sessionExTitle),

	MP JEP 23010-2003 "IT in University Management NETWORK"		IEDI-Ref-Arch-DR-10.doc
	Document	IEDI Reference Architecture Specification. Draft Recommendation	Version 1.0
	Topic	6 Query Decomposition - 6.2 Sub-Query Extraction by a Walkthrough Example	28/02/2004

<pre>(?y, abo:examOnDiscipline,?r1), (?a, abo:specialityName, ?specialityName) (?r1,abo:disciplineName,?entrDiscName), (?r1,abo:includes, ?i1), *(?r2,abo:includes, ?i2), *(?r2,abo:disciplineName,?sessionDiscName), (?i1,abo:disciplineName,?discName1), *(?i2,abo: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") AND (?sessionGrade eq "2") AND (?semesterNum eq "1")  USING abo FOR &lt;MDO-URL#&gt;</pre>	<pre>(?z, stud:examType, ?examType2), (?z, stud:grade, ?sessionGrade), (?z, stud:semesterNum,?semesterNum), *(?z, stud:examOnDiscipline,?r2), (?a, stud:specialityName, ?specialityName) 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") AND (?sessionGrade eq "2") AND (?semesterNum eq "1")  USING stud FOR &lt;MDO-URL#&gt;</pre>
--	--


In the WHERE sections of both queries symbol “\*” marks triples with the “dead” link

Next steps clarify intermediate IRO[m] Queries.

After the fifth step (Making the query connected RDQL graph) triples with dead <obj>-part will be removed (see Table 8).

Table 8. Producing connected RDQL graph.

IRO[1] – Entrant Intermediate query	IRO[2] – Faculty Intermediate query
<pre>SELECT ?firstName, ?secondName, ?lastName, ?specialityName, ?sessionExTitle  WHERE  (?x, abo:aboName, ?firstName), (?x, abo:secondName, ?secondName), (?x, abo:surname, ?lastName), (?x, abo:passes, ?y), (?x, abo:AboSpec, ?a), (?y, abo:EntrantExamName,?entrantExTitle), (?y, abo:examType, ?examType1), (?y, abo:grade, ?entrantGrade), (?y, abo:examOnDiscipline,?r1), (?a, abo:specialityName, ?specialityName) (?r1,abo:disciplineName,?entrDiscName), (?r1,abo:includes, ?i1), (?i1,abo:disciplineName,?discName1),  AND (?examType1 eq "Exam"), (?examType2 eq "Exam")</pre>	<pre>SELECT ?firstName, ?secondName, ?lastName, ?specialityName, ?sessionExTitle  WHERE  (?x, stud:name, ?firstName), (?x, stud:secondName, ?secondName), (?x, stud:surName, ?lastName), (?x, stud:examPasses, ?z), (?x, stud:onSpec, ?a), (?z, stud:examName,?sessionExTitle), (?z, stud:examType, ?examType2), (?z, stud:grade, ?sessionGrade), (?z, stud:semesterNum,?semesterNum), (?a, stud:specialityName, ?specialityName)  AND (?examType1 eq "Exam"), (?examType2 eq "Exam")</pre>


	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>6 Query Decomposition - 6.2 Sub-Query Extraction by a Walkthrough Example</b>	<b>28/02/2004</b>

<pre> 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") AND (?sessionGrade eq "2") AND (?semesterNum eq "1")  USING abo FOR &lt;MDO-URL#&gt; </pre>	<pre> 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") AND (?sessionGrade eq "2") AND (?semesterNum eq "1")  USING stud FOR &lt;MDO-URL#&gt; </pre>
--	---

Results of the sixth step (**forming the SELECT-section**) and seventh step (**forming the AND-section**) are presented in the Table 9.

Table 9. Forming SELECT and AND sections.

<b>IRO[1] – Entrant Intermediate query</b>	<b>IRO[2] – Faculty Intermediate query</b>
<pre> SELECT ?firstName, ?secondName, ?lastName, ?specialityName  WHERE   (?x, abo:aboName, ?firstName),   (?x, abo:secondName, ?secondName),   (?x, abo:surname, ?lastName),   (?x, abo:passes, ?y),   (?x, abo:AboSpec, ?a),   (?y, abo:EntrantExamName,?entrantExTitle),   (?y, abo:examType, ?examType1),   (?y, abo:grade, ?entrantGrade),   (?y, abo:examOnDiscipline,?r1),   (?a, abo:specialityName, ?specialityName)   (?r1,abo:disciplineName,?entrDiscName),   (?r1,abo:includes, ?i1),   (?i1,abo:disciplineName,?discName1),  AND (?examType1 eq "Exam") AND (?entrDiscName eq "Mathematics") AND ((?entrantExTitle eq ? discName1) AND (?entrantGrade eq "5")  USING abo FOR &lt;IRO Entrant-URL#&gt; </pre>	<pre> SELECT ?firstName, ?secondName, ?lastName, ?specialityName, ?sessionExTitle  WHERE   (?x, stud:name, ?firstName),   (?x, stud:secondName, ?secondName),   (?x, stud:surName, ?lastName),   (?x, stud:examPasses, ?z),   (?x, stud:onSpec, ?a),   (?z, stud:examName,?sessionExTitle),   (?z, stud:examType, ?examType2),   (?z, stud:grade, ?sessionGrade),   (?z, stud:semesterNum,?semesterNum),   (?a, stud:specialityName, ?specialityName)  AND (?examType2 eq "Exam") AND ((?sessionExTitle eq "Linear Algebra")        (?sessionExTitle eq "Mathematical Analysis")) AND (?sessionGrade eq "2") AND (?semesterNum eq "1")  USING stud FOR &lt;IRO-Faculty URL#&gt; </pre>

	MP JEP 23010-2003 "IT in University Management NETWORK"		IEDI-Ref-Arch-DR-10.doc
	Document	IEDI Reference Architecture Specification. Draft Recommendation	Version 1.0
	Topic	7 Query Performance - 7.1 Query Translation	28/02/2004

## 7 Query Performance

AU queries are processed distributedly, as partial queries at different IR wrappers. As it was outlined in Section 3.4 the task for a query performance comprises (refer to Fig. 22):

- Query translation
- Query execution
- Query results mark-up

Query execution is performed on the IR Server Side. Query Execution component of an IR Wrapper communicates the query and receives the query results from the IR Server through JDBC.

Query results mark-up is performed according to the mapping of IR metadata (attribute names) to IRO concepts and/or slots. This mapping is the part of the IRO.

Query translation is the most complex activity and is performed as described in the following subsection 7.1.

### 7.1 Query Translation

The purpose of the query translation is twofold:

- Translate the incoming query specified in the notation of the IEDI QFL (RDQL) to the notation of the IR Query Language
- Translate the terminology used in the incoming query and specified in the terms of the IRO to the terminology bounded with the IR (the metadata)

The procedure of the terminology translation is invariant to the IR Query Language, but depends on the formalism IR description and the type of the specific IR. The procedure of the language notation translation depends only on the specific query language applicable to the IR Server. Therefore the component of the Query translation is implemented not for a generic IR Wrapper, but for a IR Wrapper Binding for a particular IR.

The following Query Translation Algorithm is specified for an IR in the form of a relational database and SQL query language. This type of an IR is the most common for the IEDI.

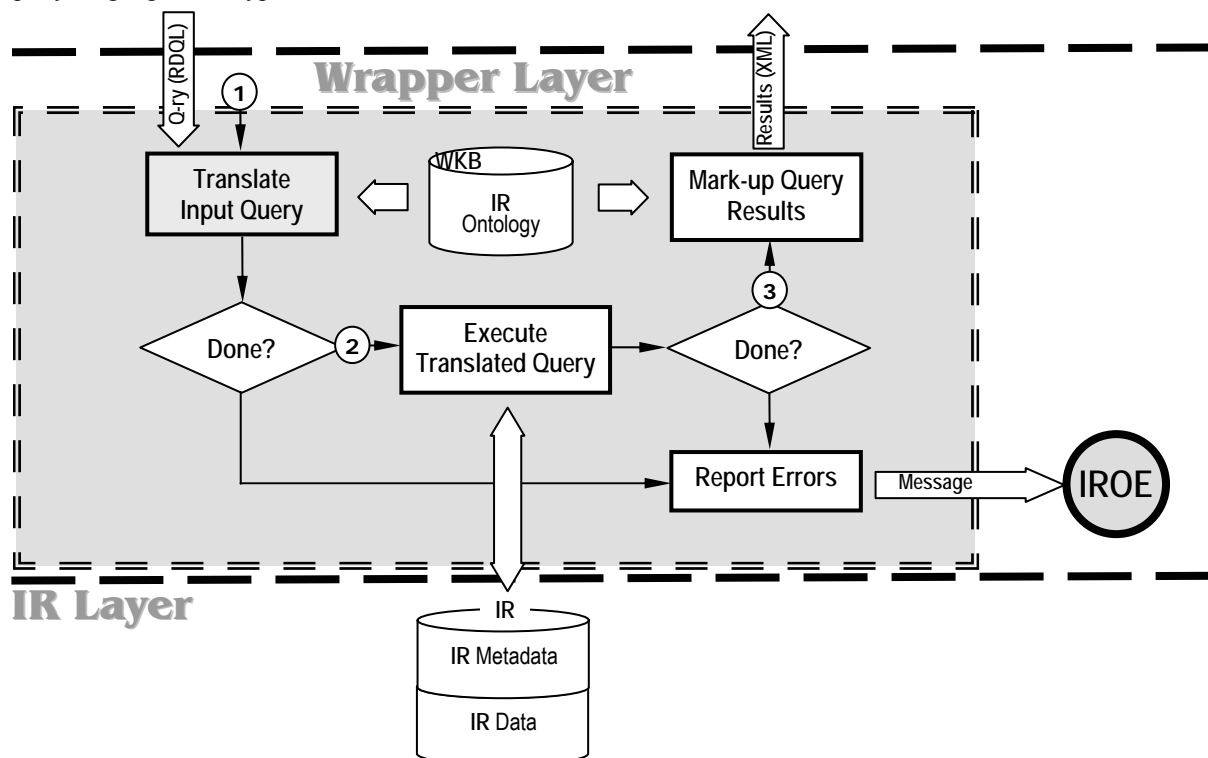


Fig. 22. Query processing by IR Wrapper.

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>7 Query Performance - 7.1 Query Translation</b>	<b>28/02/2004</b>

### 7.1.1 Query Translation Algorithm

Input: RDQL Query

Output: SQL Query

Pre-conditions: ---

Function: Query Translation

Post-effects: ---

The following algorithm performs both the translation of the language notation and terminological mapping from the terms of IRO to the terms of the IR Schema

1. **Form the SELECT clause of the SQL query:**  
For each variable name in the SELECT clause of the RDQL query replace variable names by corresponding slot names taken from WHERE clause of the RDQL query.
2. For each slot name in the SELECT clause and in the WHERE clause replace the slot name by <Table name>.<Field name>. The mapping is taken from *IRO Resource Mapping* table.
3. **Form the FROM clause of the SQL query:**  
Add all table names used in QUERY (SELECT and WHERE clauses) to FROM clause.
4. **Form the WHERE clause of the SQL query:**  
**For** each triple in the triple list section of RDQL WHERE clause  
(<?x, TableName.FieldName, ?y>)  
**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 SQL 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  
**End For**
5. **For** each pair of table names from the SQL SELECT clause  
**If** the pair appears as the **PrimaryTable – ForeignTable** pair in the *IRO Relationships* Table  
Add corresponding **Expression** from *IRO Relationships* Table to the SQL WHERE clause (connect with AND logical connective)  
**End If**  
**End For**
6. Remove USING clause of RDQL query.


### 7.1.2 Query Translation by a Walkthrough Example

This section illustrates the procedure of the Input Query Translation by the application of the algorithm described in Section 7.1.1 to the partial RDQL query to the Faculty Student IR taken from the walkthrough example of Section 2. The query in the natural language is as follows:

***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)  
Display Student's Name, Second Name, Surname, his Speciality and the Name of the Course.***

In RDQL and in the Faculty Student IRO terminology the same query looks like as follows:

```
SELECT ?name, ?secondName, ?surName, ?specialityName, ?examName
```

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>7 Query Performance - 7.1 Query Translation</b>	<b>28/02/2004</b>

```

WHERE
  (?x, stud:name, ?name),
  (?x, stud:secondName, ?secondName),
  (?x, stud:surName, ?surName),
  (?x, stud:examPasses, ?y),
  (?y, stud:examName, ?examName),
  (?y, stud:grade, ?sessionGrade),
  (?y, stud:semesterNum, ?semesterNum),
  (?x, stud:onSpec, ?a),
  (?a, stud:specialityName, ?specialityName)
AND ((?examName eq "Linear Algebra") || (?examName eq "Mathematical Analysis"))
  AND (?sessionGrade eq 2)
  AND (?semesterNum eq 1)
USING stud FOR <Faculty Student IRO URL#>

```

Applying the 1-st step of the query translation algorithm results in the following:

```

SELECT stud:name, stud:secondName, stud:surName, stud:specialityName,
  stud: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, ?sessionGrade),
  (?y, stud:semesterNum, ?semesterNum),
  (?x, stud:onSpec, ?a),
  (?a, stud:specialityName, ?specialityName)
AND ((?examName eq "Linear Algebra") || (?examName eq "Mathematical Analysis"))
  AND (?sessionGrade eq 2)
  AND (?semesterNum eq 1)
USING stud FOR <Faculty Student IRO URL#>

```


Applying the 2-nd step of the query translation algorithm results in the following:

```

SELECT Student.name, Student.secondName, Student.surName,
  Speciality.specName, Exams.examName
WHERE
  (?x, Student.name, ?name),
  (?x, Student.secondName, ?secondName),
  (?x, Student.surName, ?surName),
  (?x, SessionExam.code, ?y),
  (?y, Exams.examName, ?examName),
  (?y, SessionExam.grade, ?sessionGrade),
  (?y, SessionExam.semesterNum, ?semesterNum),
  (?x, StudentOnSpec.code, ?a),
  (?a, Speciality.specName, ?specialityName)
AND ((?examName eq "Linear Algebra") || (?examName eq "Mathematical Analysis"))
  AND (?sessionGrade eq 2)
  AND (?semesterNum eq 1)
USING stud FOR <Faculty Student IRO URL#>

```

Applying the 3-d and 4-th step of the query translation algorithm results in the following:

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>7 Query Performance - 7.1 Query Translation</b>	<b>28/02/2004</b>

```

SELECT Student.name, Student.secondName, Student.surName,
       Speciality.specName, Exams.examName
FROM Student, Speciality, SessionExam, Exams, StudentOnSpec
WHERE
  (?x, Student.name, ?name),
  (?x, Student.secondName, ?secondName),
  (?x, Student.surName, ?surName),
  (?x, SessionExam.code, ?y),
  (?y, Exams.examName, ?examName),
  (?y, SessionExam.grade, ?sessionGrade),
  (?y, SessionExam.semesterNum, ?semesterNum),
  (?x, StudentOnSpec.code, ?a),
  (?a, Speciality.specName, ?specialityName)
AND ((?examName eq "Linear Algebra") || (?examName eq "Mathematical Analysis"))
AND (?sessionGrade eq 2)
AND (?semesterNum eq 1)
USING stud FOR <Faculty Student IRO URL#>

```

And, finally, applying the 5-th and 6-th step of the query translation algorithm results in the following:

```

SELECT Student.name, Student.secondName, Student.surName,
       Speciality.specName, Exams.examName
FROM Student, Speciality, SessionExam, Exams, StudentOnSpec
WHERE
  SessionExam.grade = 2 AND
  SessionExam.semesterNum = 1 AND
  (
    Exams.examName = "Linear Algebra" OR
    Exams.examName = "Mathematical Analyses"
  ) AND
  Exams.code = SessionExam.examcode AND
  Student.code = StudentOnSpec.Studcode AND
  StudentOnSpec.code = SessionExam.StudSpecCode AND
  StudentOnSpec.speccode = Speciality.code;

```


Faculty Student IRO mappings used in the query translation are given in the following Tables 10 and 11.

Table 10. IRO Relationship Mapping for Student

PrimaryTable	ForeignTable	expression
ExamType	SessionExam	SessionExam.extypecode=ExamType.code
Student	StudentOnSpec	StudentOnSpec.syudcode=Student.code
Speciality	StudentOnSpec	StudentOnSpec.speccode=Speciality.code
Exams	SessionExam	SessionExam.examcode=Exams.code
StudentOnSpec	SessionExam	StudentOnSpec.code=SessionExam.StudSpecCode


Table 11. IRO Resource Mapping for Student

concept	slot	table	field
Student	Name	Student	name
Student	surName	Student	surName
Student	secondName	Student	secondName
Speciality	SpecialityName	Speciality	specname

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>7 Query Performance - 7.1 Query Translation</b>	<b>28/02/2004</b>

<b>concept</b>	<b>slot</b>	<b>table</b>	<b>field</b>
StudentOnSpec	StudentSpeciality	StudentOnSpec	studcode
SessionExam	examType	ExamType	examtype
SessionExam	semesterNum	SessionExam	semesterNum
SessionExam	grade	SessionExam	grade
StudentOnSpec	examPasses	SessionExam	code
Exams	examName	Exams	examName
StudentOnSpec	nGroup	StudentOnSpec	nGroup
StudentOnSpec	onSpec	StudentOnSpec	Code



	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>8 Authentication and Security - 8.1 Security Levels</b>	<b>28/02/2004</b>

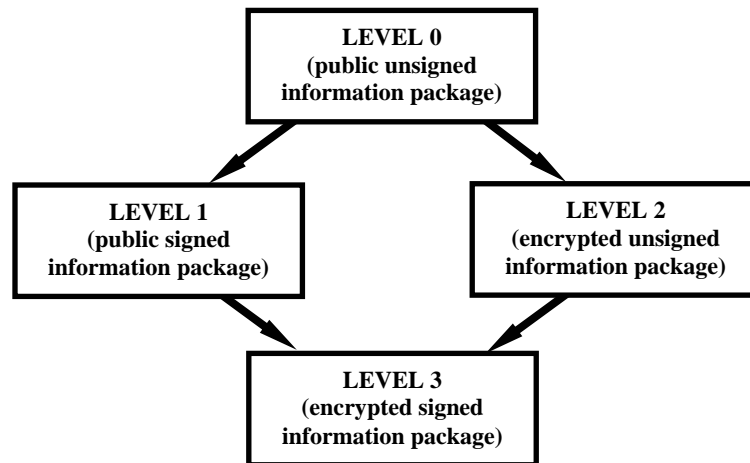


Fig. 23. Security Levels Graph

## 8 Authentication and Security

This Section uses the term Information Package to denote any block of information which is passed within UNIT-NET IEDI. Main tasks which have to be solved by the Authentication and Security Subsystem (ASS) are:

- Authentication of Information Package sender
- Cryptographic protection of Information Package
- Signing of Information Package.

### 8.1 Security Levels

In accordance with the main tasks of ASS the following types of information packages could be distinguished:

- Public unsigned information packages (level 0)
- Public signed information packages (level 1)
- Encrypted unsigned information packages (level 2)
- Encrypted signed information packages (level 3).

Logical interactions between levels are shown on Fig. 23 (the direction of an arrow shows the increase of information package security)

In order to form the heirarchy of protocols the topological sorting of the Security Levels Graph is chosen (Fig. 24).

Thus, the Four-Level Authentication and Security Protocol is defined. Its structure is shown on Fig 24.

On the level of information packages the basic data transmission protocol provided by this specification is used (SOAP for example). Next levels add their header with special information, which is necessary for sender authentication and digital electronic signature joining as well as checking the authenticity of information package, its encrypting and decrypting. The names of wrapped packages on correspondent protocol levels are:


- Encrypted package
- Signed package
- Authenticated package

### 8.2 Interactions of Authentication and Security Subsystem with IEDI Components

The interaction between Authentication and Security Subsystem and other IEDI components is shown on Fig...

The main components of Authentication and Security Subsystem are:

- U-ASS – User Authentication and Security Subsystem;

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>8 Authentication and Security - 8.2 Interactions of Authentication and Security Subsystem with IEDI Components</b>	<b>28/02/2004</b>

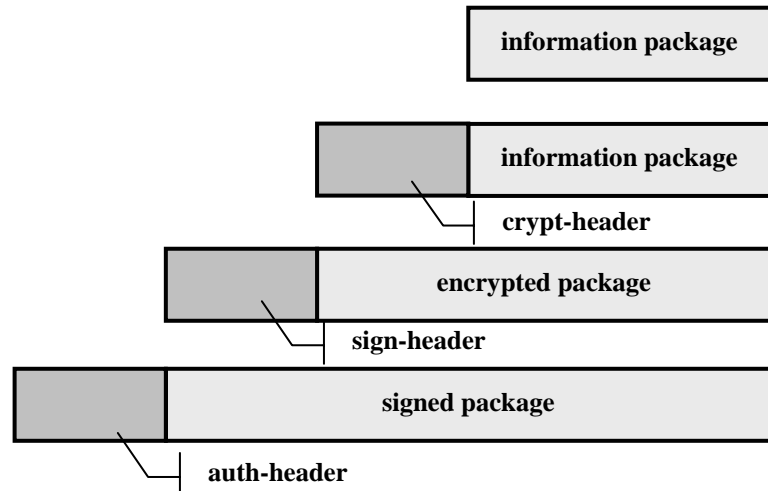
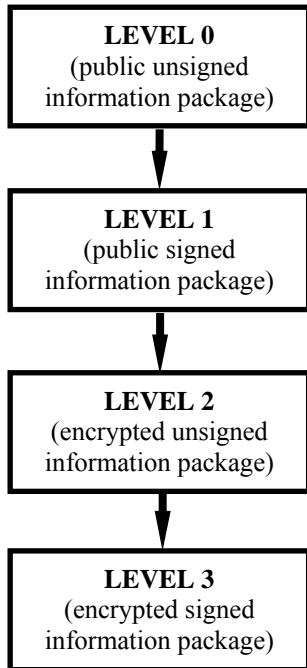


Fig. 24. Topological Sorting of Security Levels Graph

Fig. 25. Four-Level Authentication and Security Protocol

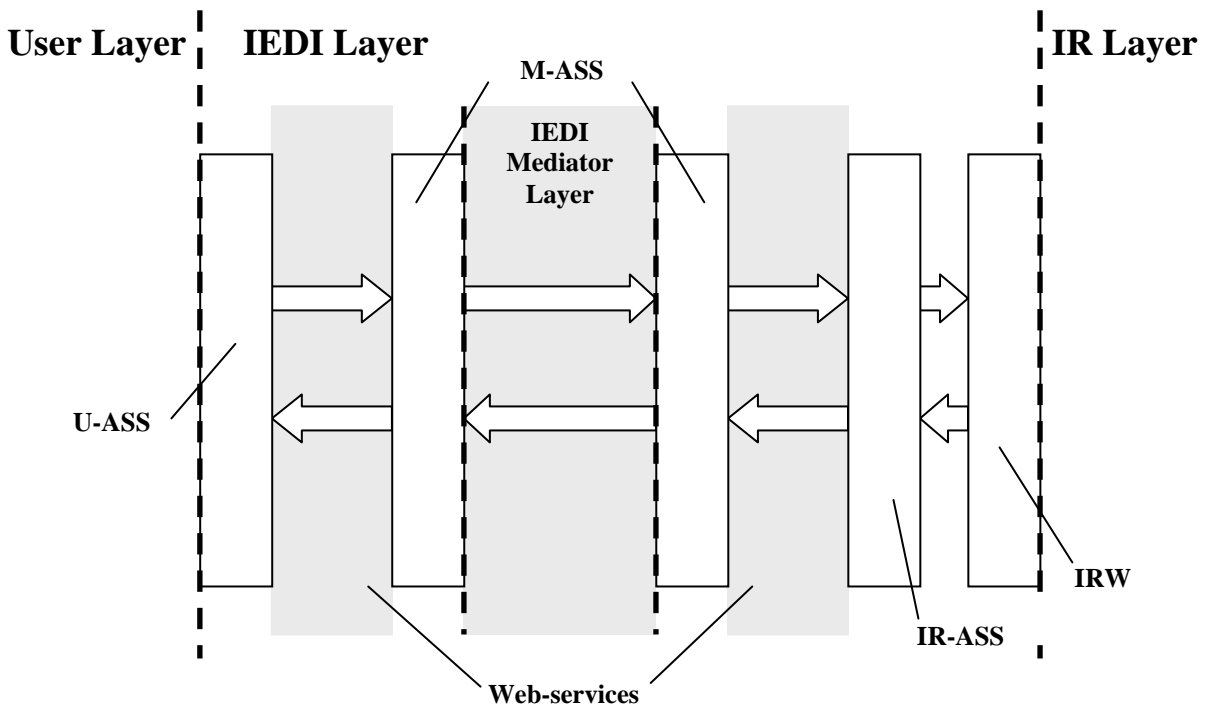



Fig. 26. Interaction Authentication and Security Subsystem with IEDI Components

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>8 Authentication and Security - 8.3 Acknowledgement of Public Key Certificates Validity</b>	<b>28/02/2004</b>

- M-ASS – Mediator Authentication and Security Subsystem;
- IR-ASS – Information Resource Authentication and Security Subsystem.

Since they are functionally symmetric further on we will identify the correspondent components of information package sender and receiver which interact through the Web-services media.

### 8.3 Acknowledgement of Public Key Certificates Validity

Acknowledgement of public key certificates validity for participants of UNIT-NET is done in accordance with the requirements of the Ukrainian Law “On electronic digital signature” [VRU03] and other normative acts provided by this law.

### 8.4 Authentication of the Sender of an Information Package

The procedure of package sender authentication consists in comparing of IP-address included in authenticated package with actual IP-address of the sender which can be found out by the analysis of the package on the transportation level of TCP protocol. If they do not coincide the notification is passed to the component designed for ASS event handling.

### 8.5 Verification of Information Package Integrity

The verification of information package integrity is done using the digital electronic signature mechanism. If the information package is not signed it is considered that its integrity is uncrippled.

### 8.6 Encryption and Decryption Information Package

The protecting of information package from unapproved access is done by its encrypting with symmetric infinite key obtained by the generator of pseudo random numbers. The generator parameters are passed in the package header and encrypted with the receiver public key.

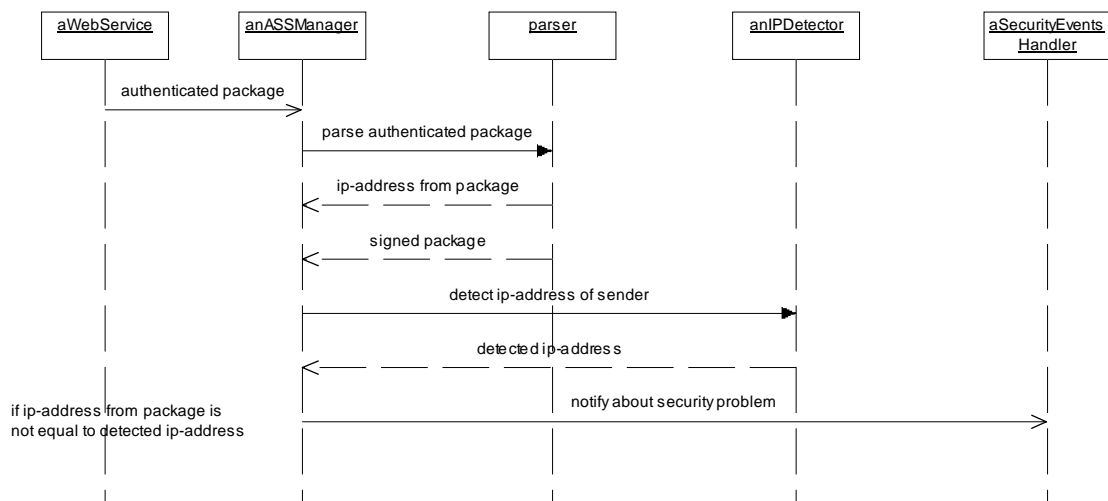



Fig. 27. Control Flow Diagram for Authentication of Information Package Sender

	MP JEP 23010-2003 "IT in University Management NETWORK"		IEDI-Ref-Arch-DR-10.doc
	Document	IEDI Reference Architecture Specification. Draft Recommendation	Version 1.0
	Topic	8 Authentication and Security - 8.6 Encryption and Decryption Information Package	28/02/2004

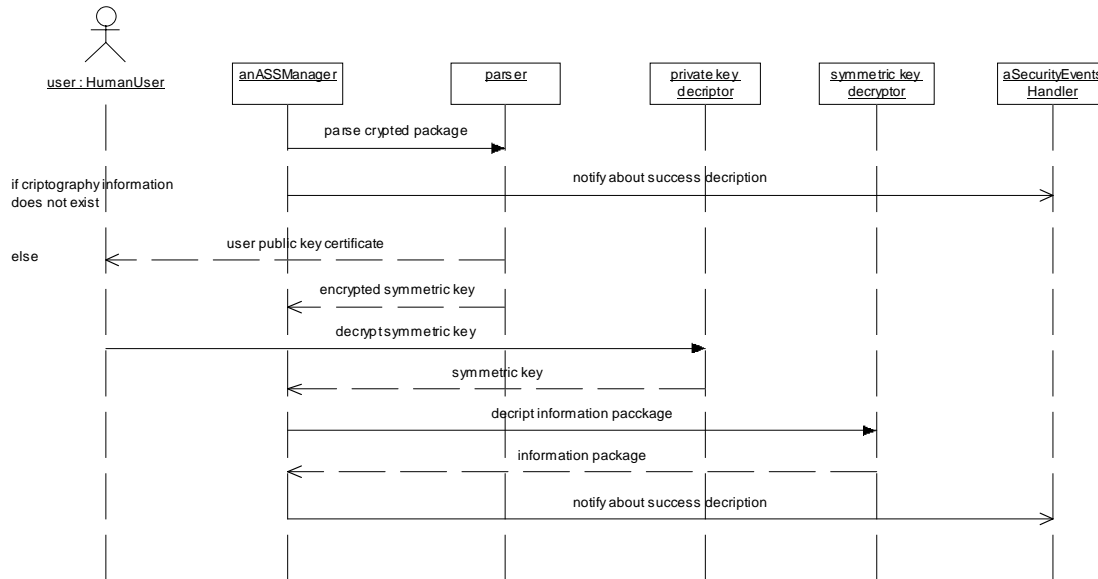


Fig. 28. Decryption Control Flow

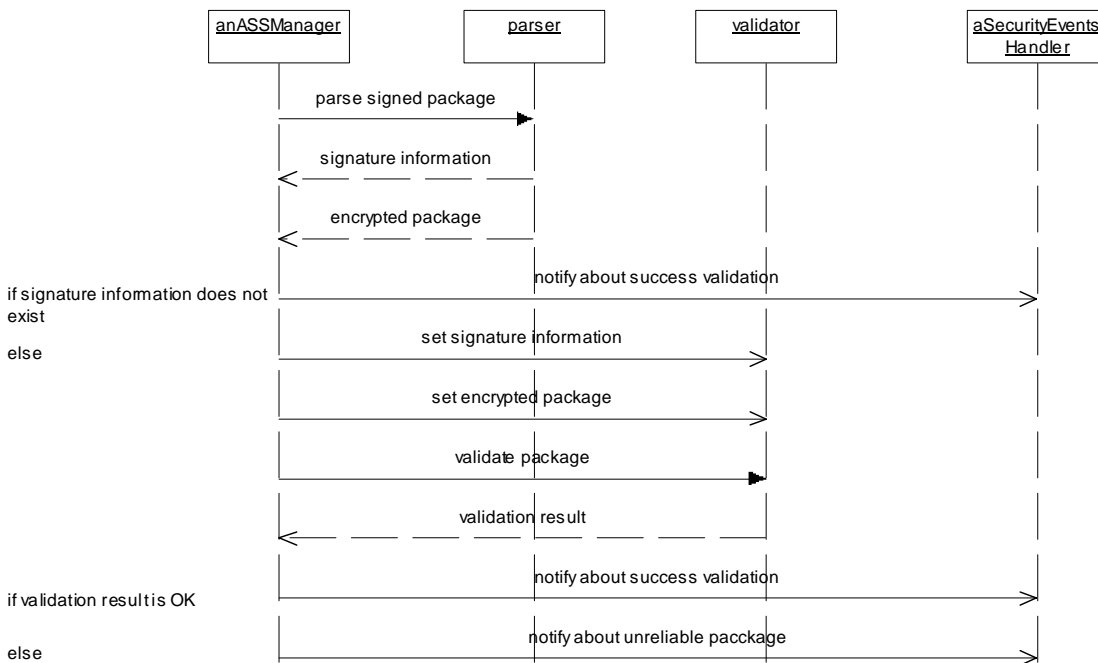


Fig. 29. Control Flow Diagram for Verification of Information Package Integrity

	MP JEP 23010-2003 "IT in University Management NETWORK"		IEDI-Ref-Arch-DR-10.doc
	Document	IEDI Reference Architecture Specification. Draft Recommendation	Version 1.0
	Topic	8 Authentication and Security - 8.7 Signature Information Package	28/02/2004

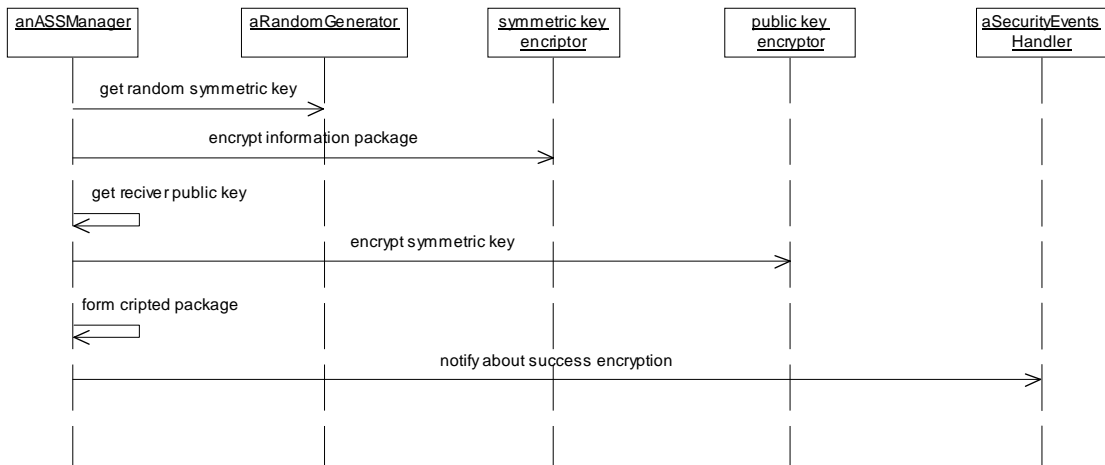


Fig. 30. Encryption Control Flow

### 8.7 Signature Information Package

The digital electronic signature is joined to the initial or encrypted package by the algorithm of public key encrypting. The choice of one or another algorithm is in the competence of the appropriate state structures and depends on the current legislation. The RSA algorithm could be used as a software prototype.

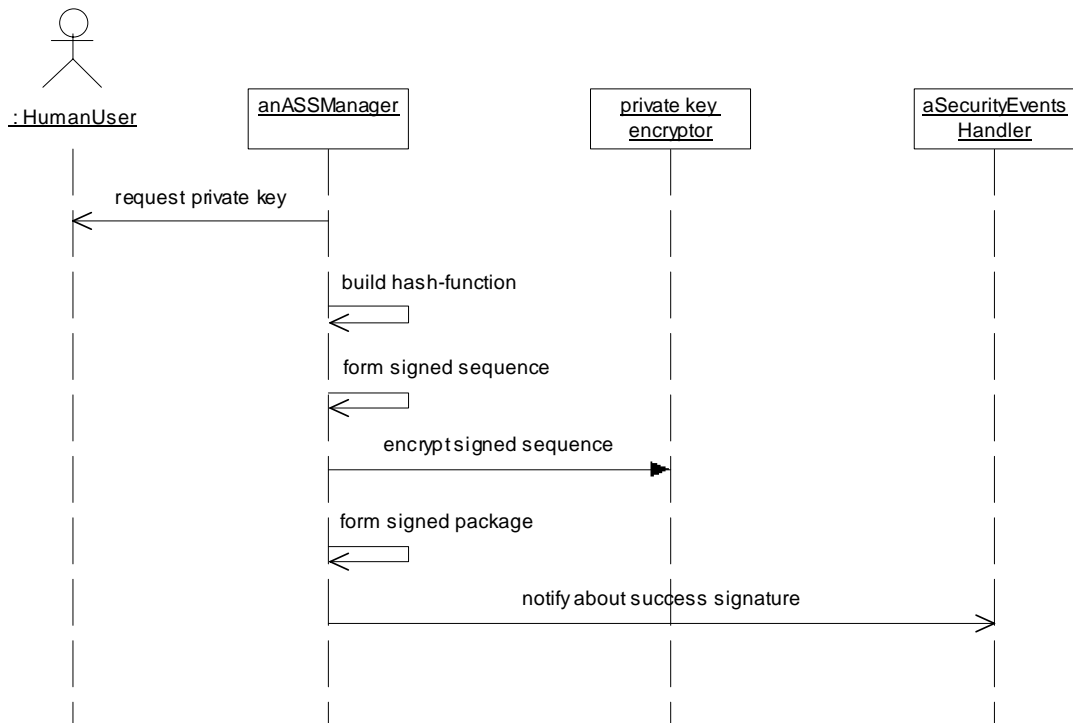



Fig. 31. Signature Control Flow


	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>References - 8.7 Signature Information Package</b>	<b>28/02/2004</b>

## References

- [AKS96] **Arens, Y., Knoblock, C.A., Shen, W.:** Query Reformulation for Dynamic Information Integration. Journal of Intelligent Information Systems. 1996.
- [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, Arisona, May 1997.
- [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.
- [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.
- [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
- [DEF99] **Decker, S., Erdmann, M., Fensel, D., and 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.
- [DOL02] **Masolo, C., Borgo, S., Gangemi A., Guarino, N., Oltramari, A., and Schneider, L.:** WonderWeb Project Deliverable D17. The WonderWeb Library of Foundational Ontologies and the DOLCE ontology. Available from: <http://www.compscipreprints.com/comp/Preprint/stborgo/20021127/2/WondeWebD17v2.0.pdf>
- [FEN00] **Fensel, D.:** Ontologies: Silver Bullet for Knowledge Management and Electronic Commerce, Springer-Verlag, 2000, ISBN 3-540-41602-1
- [GAM95] **Garcia-Molino, H. et. al.:** The TSIMMIS Approach to Mediation: Data Models and Languages. In Proceedings of the NGITS (Next Generation Information Technologies and Systems), June 1995.
- [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.
- [IEDI-ASS04] UNIT-NET IEDI Authentication and Security Specification. Working Draft (to be further elaborated in 2004)
- [IEDI-DOS04] UNIT-NET IEDI Upper Level and Domain Semantics Specification. Working Draft (to be further elaborated in 2004)
- [IEDI-QFLS04] UNIT-NET IEDI Query Formulation Language Specification. Working Draft (to be further elaborated in 2004)
- [IEDI-RWS04] UNIT-NET IEDI Information Resource Wrapper Specification. Working Draft (to be further elaborated in 2004)
- [JENA04] Jena – a Semantic Web Framework for Java. Available from: <http://jena.sourceforge.net/>
- [KS00] **Kashyap V., Sheth A.:** Information Brokering across Heterogeneous Digital Data: A Metadata-based Approach. Kluwer Academic Publishers, 2000
- [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.
- [NSD01] **F.-Noy, N., Sintek, M., Decker, S., Crubezy, M., Ferguson, R. W., and Musen, M. A.:** Creating Semantic Web Contents with Protege-2000. IEEE Intelligent Systems 16(2):60-71, 2001.
- [OWL03] OWL Web Ontology Language Reference. W3C Proposed Recommendation 15 December 2003. Available from: <http://www.w3.org/TR/owl-ref/>
- [RDQL04] RDQL - A Query Language for RDF. W3C Member Submission, 9 January 2004. Available from: <http://www.w3.org/Submission/2004/SUBM-RDQL-20040109/>
- [SOAP03] SOAP Version 1.2 Part 1: Messaging Framework. W3C Recommendation 24 June 2003. Available from: <http://www.w3.org/TR/SOAP/>

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>References - 8.7 Signature Information Package</b>	<b>28/02/2004</b>

- [STU00] **Stuckenschmidt H., Wache H., Voegelé T., Visser U.**: Enabling technologies for interoperability. In: (Visser, U., Pundt H. Eds.) Workshop on the 14<sup>th</sup> International Symposium of Computer Science for Environmental Protection, Bonn, Germany, 2000, pp. 35-46.
- [VRU03] The Law of Ukraine "On electronic digital signature". Verhovna Rada of Ukraine, 22.05.2003 № 852-IV
- [WAC01] **Wache, H., Voegelé, T., Visser, U., Stuckenschmidt, H., Schuster, G., Neumann, H., and 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>
- [WIE92] **Wiederhold, G.**: Mediators in the Architecture of Future Information Systems. IEEE Computer, 25, 3 (March), 1992, 38–49.
- [WSA03] Web Services Architecture. W3C Working Draft. 8 August 2003. Available from: <http://www.w3.org/TR/2003/WD-ws-arch-20030808/>

	MP JEP 23010-2003 "IT in University Management NETWORK"		IEDI-Ref-Arch-DR-10.doc
	Document	IEDI Reference Architecture Specification. Draft Recommendation	Version 1.0
	Topic	Appendices - A. OWL code for the walkthrough example fragments of the University Entrant IRO, the University Faculty Students IRO and MDO	28/02/2004

## Appendices

### A. OWL code for the walkthrough example fragments of the University Entrant IRO, the University Faculty Students IRO and MDO


Example of the University Entrant IRO (fragment)

```

<rdf:RDF
  xmlns:rss="http://purl.org/rss/1.0/"
  xmlns:jms="http://jena.hpl.hp.com/2003/08/jms#"
  xmlns:j.0="http://protege.stanford.edu/plugins/owl/protege#"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:owl="http://www.w3.org/2002/07/owl#"
  xmlns:vcard="http://www.w3.org/2001/vcard-rdf/3.0#"
  xmlns:daml="http://www.daml.org/2001/03/daml+oil#"
  xmlns:dc="http://purl.org/dc/elements/1.1/">
  <owl:Ontology rdf:about="">
    <owl:imports
rdf:resource="http://protege.stanford.edu/plugins/owl/protege"/>
  </owl:Ontology>
  <owl:Class rdf:ID="Director_Supervision_Relation">
    <rdfs:label>Director Supervision Relation</rdfs:label>
    <rdfs:subClassOf>
      <owl:Class rdf:about="#Manager_Supervision_Relation"/>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty
rdf:resource="http://protege.stanford.edu/plugins/owl/protege#from"/>
          <owl:allValuesFrom>
            <owl:Class>
              <owl:oneOf rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"
                rdf:type="http://www.w3.org/1999/02/22-rdf-syntax-ns#List"/>
            </owl:Class>
          </owl:allValuesFrom>
        </owl:Restriction>
      </rdfs:subClassOf>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty
rdf:resource="http://protege.stanford.edu/plugins/owl/protege#from"/>
          <owl:minCardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
            >1</owl:minCardinality>
        </owl:Restriction>
      </rdfs:subClassOf>

```



	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>Appendices - A. OWL code for the walkthrough example fragments of the University Entrant IRO, the University Faculty Students IRO and MDO</b>	<b>28/02/2004</b>

```

<rdfs:subClassOf>
  <owl:Restriction>
    <owl:onProperty
rdf:resource="http://protege.stanford.edu/plugins/owl/protege#to"/>
    <owl:allValuesFrom>
      <owl:Class>
        <owl:oneOf rdf:resource="http://www.w3.org/1999/02/22-rdf-
syntax-ns#nil"/>
      </owl:Class>
    </owl:allValuesFrom>
  </owl:Restriction>
</rdfs:subClassOf>
<rdfs:subClassOf>
  <owl:Restriction>
    <owl:onProperty
rdf:resource="http://protege.stanford.edu/plugins/owl/protege#to"/>
    <owl:minCardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
    >1</owl:minCardinality>
  </owl:Restriction>


```

#### Example of the Faculty Student IRO (fragment)

```

<rdf:RDF
  xmlns:rss="http://purl.org/rss/1.0/"
  xmlns:jms="http://jena.hpl.hp.com/2003/08/jms#"
  xmlns:j.0="http://protege.stanford.edu/plugins/owl/protege#"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:owl="http://www.w3.org/2002/07/owl#"
  xmlns:vcard="http://www.w3.org/2001/vcard-rdf/3.0#"
  xmlns:daml="http://www.daml.org/2001/03/daml+oil#"
  xmlns:dc="http://purl.org/dc/elements/1.1/">
  <owl:Ontology rdf:about="">
    <owl:imports
rdf:resource="http://protege.stanford.edu/plugins/owl/protege"/>
  </owl:Ontology>
  <owl:Class rdf:ID="StudentOnSpec">
    <rdfs:subClassOf rdf:resource="#Student"
      rdf:type="http://www.w3.org/2002/07/owl#Class"/>
  </owl:Class>
  <owl:Class rdf:ID="SessionExam">
    <rdfs:subClassOf rdf:resource="#Exams"
      rdf:type="http://www.w3.org/2002/07/owl#Class"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty>
        <owl:DatatypeProperty rdf:about="#semesterNum"/>

```

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>Appendices - A. OWL code for the walkthrough example fragments of the University Entrant IRO, the University Faculty Students IRO and MDO</b>	<b>28/02/2004</b>

```

</owl:onProperty>
<owl:hasValue rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
>1</owl:hasValue>
</owl:Restriction>
</rdfs:subClassOf>
<rdfs:subClassOf>
<owl:Restriction>
<owl:onProperty>
<owl:DatatypeProperty rdf:about="#semesterNum"/>
</owl:onProperty>
<owl:hasValue rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
>2</owl:hasValue>
</owl:Restriction>
</rdfs:subClassOf>
<rdfs:subClassOf>
<owl:Restriction>
<owl:onProperty>
<owl:DatatypeProperty rdf:about="#semesterNum"/>
</owl:onProperty>
<owl:hasValue rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
>3</owl:hasValue>
</owl:Restriction>
</rdfs:subClassOf>
<rdfs:subClassOf>
<owl:Restriction>
<owl:onProperty>


```

#### Example of the MDO (fragment)

```

<rdf:RDF
xmlns:rss="http://purl.org/rss/1.0/"
xmlns:jms="http://jena.hpl.hp.com/2003/08/jms#"
xmlns:j.0="http://protege.stanford.edu/plugins/owl/protege#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:vcard="http://www.w3.org/2001/vcard-rdf/3.0#"
xmlns:daml="http://www.daml.org/2001/03/daml+oil#"
xmlns:dc="http://purl.org/dc/elements/1.1/">
<owl:Ontology rdf:about="">
<owl:imports
rdf:resource="http://protege.stanford.edu/plugins/owl/protege"/>
</owl:Ontology>
<owl:Class rdf:ID="Person"/>
<owl:Class rdf:ID="Discipline"/>
<owl:Class rdf:ID="SessionExam">
<rdfs:subClassOf rdf:resource="#Exam"

```

	<b>MP JEP 23010-2003 "IT in University Management NETWORK"</b>		IEDI-Ref-Arch-DR-10.doc
	Document	<b>IEDI Reference Architecture Specification. Draft Recommendation</b>	<b>Version 1.0</b>
	Topic	<b>Appendices - A. OWL code for the walkthrough example fragments of the University Entrant IRO, the University Faculty Students IRO and MDO</b>	<b>28/02/2004</b>

```

    rdf:type="http://www.w3.org/2002/07/owl#Class"/>
</owl:Class>
<owl:Class rdf:ID="Speciality"/>
<owl:Class rdf:ID="PersonOnSpeciality">
  <rdfs:subClassOf rdf:resource="#Person"/>
</owl:Class>
<owl:Class rdf:ID="EntrantExam">
  <rdfs:subClassOf rdf:resource="#Exam"/>
</owl:Class>
<owl:Class rdf:ID="SertificationExam">
  <rdfs:subClassOf rdf:resource="#Exam"/>
</owl:Class>
<owl:ObjectProperty rdf:ID="on_spec">
  <rdfs:range rdf:resource="#Speciality"/>
  <rdfs:domain rdf:resource="#PersonOnSpeciality"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="includes"
  rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
  <rdfs:range rdf:resource="#Discipline"/>
  <rdfs:domain rdf:resource="#Discipline"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="grade"/>
<owl:ObjectProperty rdf:ID="exams_passes">
  <rdfs:range rdf:resource="#Exam"/>
  <rdfs:domain rdf:resource="#PersonOnSpeciality"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="exam_spec"
  rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
  <rdfs:range rdf:resource="#Exam"/>
  <rdfs:domain rdf:resource="#Speciality"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="examOnDiscipline"

```