

# On the application of the Semantic Web Rule Language in the Definition of Policies for System Security Management

Félix J. García Clemente, Gregorio Martínez Pérez, Juan A. Botía Blaya and  
Antonio F. Gómez Skarmeta

Departamento de Ingeniería de la Información y las Comunicaciones  
University of Murcia, Spain  
{fgarcia, gregorio, skarmeta}@dif.um.es, juanbot@um.es

**Abstract.** The adoption of a policy-based approach for the dynamic regulation of a system or service (e.g. security, QoS or mobility service) requires an appropriate policy representation and processing. In the context of the Semantic Web, the representation power of languages enriched with semantics (i.e. semantic languages), together with the availability of suitable interpreters, make such kind of languages well suited for policies representation. In this paper, we describe our proposal for the combination of the CIM-OWL ontology (i.e., the mapping of the DMTF Common Information Model into OWL) with the Semantic Web Rule Language as the basis for a semantically-rich security policy language that can be used to formally describe the desired security behaviour of a system or service. An example of security policy in this language and its reasoning are also presented.

## 1. Introduction

A policy-based management enables a system administrator to specify rules that describe domain-wide policies. These specifications are defined by using high level languages in such an extent that they are totally decoupled of any possible implementation of the system. There are multiple approaches for policy specification. Two examples are (1) formal policy languages that a computer can easily and directly process and interpret and (2) rule-based policy languages based on conventional if-then rules which includes the representation of policies based on a deontic logic for the expression of rules related to obligation and permissibility.

The adoption of a policy-based approach for security management requires an appropriate policy representation and an engine for policy processing that enables runtime adaptability and extensibility of the system, as well as the possibility of enabling analysis of policies relating to entities described at different levels of abstraction. In this sense, semantic approaches based on the combination of ontology languages [6] and rule-based languages satisfy these requirements. For example, organizations may utilize a common ontology that can be shared amongst services and service clients to define rule-based security policies. The use of ontologies also facilitates the process of reasoning over the structure and interrelations of a set of

policies easing the detection of conflicts. This functionality is especially relevant in medium and large organizations which may manage tens or hundreds of different policy rules at the same time.

One standard that provides a common model for describing the necessary concepts to be considered within a policy domain is the DMTF Common Information Model (CIM). The CIM specification is a semi-formal ontology that does not support interoperability and reasoning. These limitations are due, in part, to the constraints imposed by the languages in which the CIM model is specified (e.g. XML and XML Schema). To overcome the mentioned drawbacks we may model, represent and share CIM in the form of a formal ontology. In this sense, the first part of the paper presents a CIM-OWL ontology (i.e., the specification of the DMTF Common Information Model in OWL, which stands for Ontology Web Language) in section 2. With this idea in mind, there is still a necessity of specifying the policy rules themselves. This is something that one may afford with OWL. However, OWL does not allow for specifying rules directly. Instead, one has to define rules skeleton as regular classes in the ontology. Clearly, this is not a suitable approach. For the policy reasoning part we have decided to adopt a specific language for defining and interpreting rules. There are multiples rule languages, and normally each one is designed and developed together with a concrete inference engine. One language that has been designed as a semantic interoperable vehicle for heterogeneous policy languages is Semantic Web Rule Language (SWRL). The SWRL language provides intermediate mark-up syntax, with associated deep knowledge representation semantics for interchange between those languages. For interchange between policy languages that are already XML-based, this may, for instance, be achieved using XSL transformations (XSLT), e.g., to translate into and then out of SWRL. This part of the work is described in section 3. The different implications of this approach are analyzed in section 4. Section 5 shows an example of a security policy, and how and what may be reasoned from it. Finally, section 6 outlines most important conclusions we have obtained and points out some new work directions we are now involved in.

## **2. A Summary of the CIM-OWL Ontology**

CIM [2] is an approach from the DMTF that applies the basic modelling techniques of the object-oriented paradigm to provide a common definition of management-related information. It comprises a core model that defines a basic classification of elements and associations for a managed environment (e.g., logical and physical elements, capabilities, settings and profiles) as well as more specific models that define concepts that are common to particular management areas (e.g., applications, systems, devices and users).

CIM is independent of any implementation or specific specification. However, for an information model to be useful, it has to be mapped into some implementation. In this sense, CIM can be mapped to multiple structured specifications. For example, in [1] the authors describe a mapping from the CIM resource model and related operations to the Web services paradigm using XML and WSDL. This specification permits one to model the management of web services using the DMTF methodology

and hence to obtain its standard representation. Whereas the XML-encoded CIM specification can not be arbitrarily combined with other specifications in a flexible manner and, with respect to more advanced operations, XML-encoded specifications do not embody the constructs for facilitating tasks like parsing, logical deduction or semantic interpretation.

Other specific specification of CIM may be performed by using OWL [7]. This representation of CIM makes easy to perform useful reasoning tasks. This is due to the fact that OWL is based on description logics [8]. This simple and yet powerful kind of first order like logic allows for reasoning not only on individuals but also on the structure of them with efficient and sound algorithms. In this sense, we presented a proposal for expressing CIM objects in OWL, named CIM-OWL, in [3]. We have now extended this proposal with the mapping of all CIM qualifiers (i.e. meta, standard and optional) to OWL, as it is necessary to preserve all information appearing at the original model. Table 1 shows the proposed mapping of CIM qualifiers to OWL.

Table 1: **CIM qualifiers to OWL**

| <i>Qualifier</i>          | <i>How it can be mapped</i>  |
|---------------------------|--|
| <i>Meta Qualifier</i>     |  |
| Association               | Indicates that the object class is defining an association. The association class will include the <owl:ObjectProperty> tag, while any other type of classes will not. |
| Indication                | Indicates that the object class is defining an indication. It is mapped to this RDF scheme feature: <rdfs:subClassOf rdf:resource="#Indication" />                     |
| <i>Standard Qualifier</i> |  |
| Abstract                  | Indicates that the class is abstract and serves only as a base for new classes; it is mapped to this RDF scheme feature: <rdfs:subClassOf rdf:resource="#Abstract" />  |
| ArrayType                 | Indicates the type of the qualified array; it is mapped to <rdf:type rdf:resource="#ArrayType" />  |
| Deprecated                | Indication that the entity is deprecated; it is related with a versioning feature: <owl:deprecatedClass> or <owl:deprecatedProperty>                                   |
| Description               | Provides a description of a property, an operation or a class; it is mapped to <rdfs:comment>  |
| <i>Optional Qualifier</i> |  |
| Alias                     | Establishes an alternate name for a property; it results in an equality feature: <owl:equivalentProperty>  |
| Invisible                 | Indicates that the element is defined only for internal purposes; it is mapped to this RDF scheme feature: <rdfs:subClassOf rdf:resource="#Invisible" />               |

The main rules for this mapping are the following:

- If a qualifier has an equivalent representation in OWL, it is used; it is the case of the qualifiers Deprecated or Description, for example.
- If a class qualifier does not have an equivalent representation in OWL, then we propose a new class to represent it, as with the qualifiers Indication or Abstract, for example.
- If a property qualifier does not have an equivalent representation in OWL, then we propose a new class that represents a new property which model the qualifier. It is the case of the qualifier ArrayType, for example.

The automatic transformation between the XML and OWL representations of CIM can be made by defining XSL templates implementing the indicated transformations.

### **3. Specification of Security Policies with Semantic and Rule oriented Languages**

Semantic Web Rule Language (SWRL) [4] is based on a combination of the OWL DL Lite language of the OWL Web Ontology Language family with the Unary/Binary Datalog RuleML sublanguages. SWRL extends the set of OWL axioms to include a high-level abstract syntax for Horn-like rules that can be combined with an OWL knowledge base. In this manner, OWL is used to define the pieces of knowledge appearing at the logic expressions within the body and head of rules and RuleML is used to define the reasoning procedure over that knowledge.

The SWRL rules are of the form of an implication between an antecedent (body) and consequent (head). The intended meaning can be read as: whenever the conditions specified in the antecedent hold, then the conditions specified in the consequent must also hold.

A useful restriction in the form of the rules is to limit antecedent and consequent atoms to be named classes, where the classes are defined purely in OWL. Adhering to this format makes it easier to translate rules to or from future or existing rule systems, including Prolog, and Jena [5]. In the case of Jena, version 2.0 was used with the two rule systems available: forward and backward chaining, being the rules automatically transformed from SWRL version 0.5 to Jena 2.0.

Since CIM permits to represent both high-level and low-level concepts (e.g. application-level data versus network-level end points), we can use OWL-CIM plus SWRL to represent both high-level and low-level policies (e.g. “the integrity of all application data must be guaranteed” versus “block UDP connections to port 53 on host X”). Thus, policy administrators have enough flexibility to choose the appropriate level for their needs.

The specification of policies is performed in two different phases. The first one consists on selecting the correct OWL-CIM concepts to model the real system components and, if necessary, network topology. The second one deals with using the previously defined concepts to specify policy rules and, subsequently, the policies.

### **4. Automated Reasoning on Security Policies**

The combination of the CIM-OWL ontology and SWRL for specifying behaviour rules for policies offers a clear advantage: it allows two different types of automated reasoning. The first one is ontology reasoning (i.e. reasoning over the structure and instances of the ontology) and the second one is rule-based reasoning (i.e. applying policy rules in systems management tasks). Therefore, we identify an OWL reasoner and a rule-based reasoner, where we use the term reasoner to refer to a specific program that performs the task of inference (i.e. process of deriving additional information not explicitly specified).

In the first phase of policy specification, the OWL reasoner may be used for:

- *Validation*. The OWL ontology language allows constraints to be expressed; the validation operation is used to detect when such constraints are violated by some data set, i.e., the validation consists on a global check across the schema and instance data looking for inconsistencies.

- *Query the model for* instance recognition (i.e., testing if an individual is an instance of a class expression) and inheritance recognition (i.e., testing if a class is a subclass of another class and if a property is a sub-property of another).

In the second phase of policy specification, the rule reasoner is used to obtain additional knowledge not explicitly specified from the system definition. It provides forward and backward chaining reasoning and while the OWL reasoner performs inference about OWL-CIM ontology, the rule reasoner does about SWRL rules.

Moreover, the rule reasoner eases the detection of conflicts. A conflict occurs when the policy definition assigns different specifications on the behaviour of system component, e.g., one allows the user to start the service and another prohibits the same user from starting the service. It may be used to detect both static and dynamic conflicts. Static conflict detection aims to detect all types of potential conflicts (possible or definite) which clearly could cause conflicts from the policy specification. This static conflict detection is performed on the process of policy definition. Unlike static conflict detection, dynamic conflict detection is performed at run time by dynamically detecting all conflicts whenever the system definition is modified.

## 5. An Example

This section shows both ontology reasoning and rule-based reasoning examples over a particular authorization policy defined from a portion of the CIM-OWL ontology.

### The Ontology for the Example

This example shows the subset of CIM classes necessary to express authorization policies between computer systems and roles. We use the CIM classes depicted in Figure 1 to represent the management-related concepts regarding the authorization security service, computer systems, and roles.

The *Privilege* object class is the base for all types of activities, which are granted or denied to a subject by a target. *AuthorizedPrivilege* is the specific subclass for the authorization activity. Whether an individual Privilege is granted or denied is defined using the *PrivilegeGranted* boolean. The association of subjects to *AuhorizedPrivileges* is accomplished explicitly via the association *AuthorizedSubject*. The entities that are protected (targets) can be similarly defined via the association *AuthorizedTarget*. Note that *AuthorizedPrivilege* and its *AuthorizedSubject/Target* associations provide a static mechanism to represent authorization policies.

The *Role* object class is used to represent a position or set of responsibilities within an organization, organizational unit or system administration scope. It is filled by a person or persons (or non-human entities represented by *ManagedSystemElement* subclasses) that may be explicitly or implicitly members of this collection subclass.

The *ComputerSystem* object class is derived from *System* that is a special collection of *ManagedSystemElements* that provides computing capabilities. The *Dedicated* property is an enumeration indicating whether the *ComputerSystem* is a special-purpose System (i.e., dedicated to a particular use), versus being general purpose. For example, one could specify that the System is dedicated to "Print" (value=11) or acts as a "Hub" (value=8).

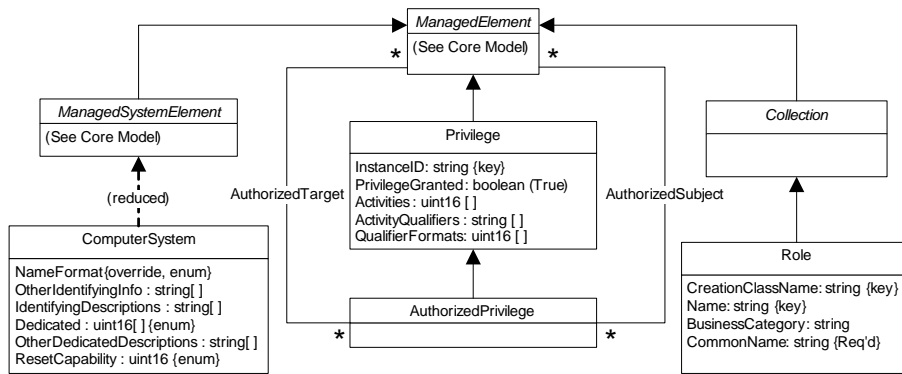


Figure 1. UML diagram of CIM classes

Since both *ComputerSystem* and *Role* are specializations of *ManagementElement*, they may be either the target or the subject of *Privilege*. Our example uses *Role* as subject and *ComputerSystem* as target.

In this particular example, the administrative domain is composed by a system dedicated to print, a role that represent the set of students of Computer Science, and a role that represent the set of students of Philosophy. It occurs that students of Philosophy have the privilege granted to print within the system, whereas students of Computer Science do not have it. Figure 2 shows the OWL-CIM representation of the administrative domain.

```

<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
  xmlns:owl="http://www.w3.org/2002/07/owl#"
  xml:base="http://www.positif.com/cim#"
  xmlns:cim="http://www.positif.com/cim#"
  xmlns="http://www.positif.com/cim#">
  <CIM_ComputerSystem rdf:about="#printserver">
    <Caption>Print Server</Caption>
    <ElementName>Print Server</ElementName>
    <Name>printserver.positif.org </Name>
    <Dedicated>Print</Dedicated>
  </CIM_ComputerSystem>
  <CIM_AuthorizedTarget rdf:about="#authtarget1">
    <CATPrivilege rdf:resource="#printauth" />
    <TargetElement rdf:resource="#printserver" />
  </CIM_AuthorizedTarget>

```

```

<CIM_AuthorizedPrivilege rdf:about="#printauth">
  <Caption>Access Authorization for the print server</Caption>
  <ElementName>Print Authorization</ElementName>
  <InstanceID>POSITIF:PrintAuth</InstanceID>
  <PrivilegeGranted>true</PrivilegeGranted>
  <Activities>Create</Activities>
</CIM_AuthorizedPrivilege>
<CIM_AuthorizedSubject rdf:about="#authsubject1">
  <CASPrivilege rdf:resource="#printauth" />
  <PrivilegedElement rdf:resource="#ST_PHY" />
</CIM_AuthorizedSubject>
<CIM_Role rdf:about="#ST_PHY">
  <Caption>Students of Philosophy</Caption>
  <ElementName>PhilosophyStudents</ElementName>
  <CreationClassName>CIM_Role</CreationClassName>
  <Name>ST_PHY</Name>
  <BusinessCategory>Students</BusinessCategory>
</CIM_Role>
<CIM_Role rdf:about="#ST_COMP">
  <Caption>Students of Computer Science</Caption>
  <ElementName>ComputerStudents</ElementName>
  <CreationClassName>CIM_Role</CreationClassName>
  <Name>ST_COMP</Name>
  <BusinessCategory>Students</BusinessCategory>
</CIM_Role>
</rdf:RDF>

```

**Figure 2. Representation in OWL-CIM of the administrative domain**

### Reasoning over the Ontology

For this administrative domain, the system administrator may validate the system definition and find inconsistencies by using the inference engine. For example, Figure 3 shows a OWL definition with one inconsistency. The inference engine infers that the *CATPrivilege* property references a resource with a non expected type, since the resource *ST\_PHY* is a *Role*, and it must be an *AuthorizedPrivilege*. Other inconsistencies that the OWL reasoner can detect are, for example, property cardinality and data types.

The policy administrator may also test the definition by querying about classes and instances. For example, he/she may perform the following question: “*Is PrintServer a ManagedSystemElement?*” The OWL reasoner recognizes the instance as *ManagementElement*. More questions that the OWL reasoner may answer are related to CIM qualifiers, e.g., “*Is ManagedSystemElement abstract?*”

```

<CIM_AuthorizedTarget rdf:about="#authtarget1">
  <CATPrivilege rdf:resource="#ST_PHY" />
  <TargetElement rdf:resource="#printserver" />
</CIM_AuthorizedTarget>

```

**Figure 3. Representation in OWL-CIM with one inconsistency**

## Reasoning with Policy Rules

For this administrative domain, the policy administrator may decide the following authorization policy: *“If a system permits to print to the students of Philosophy, then the students of Computer Science can also use this system to print”*. Figure 4 shows the SWRL representation of this authorization policy.

```
<ruleml:imp>
<ruleml:_rlab ruleml:href="#exampleRule"/>
<ruleml:_body>
  <swrlx:classAtom>
    <owlx:Class owlx:name="#CIM_AuthorizedTarget" />
    <ruleml:var>authtarget</ruleml:var>
  </swrlx:classAtom>
  <swrlx:classAtom>
    <owlx:Class owlx:name="#CIM_AuthorizedSubject" />
    <ruleml:var>authsubject1</ruleml:var>
  </swrlx:classAtom>
  <swrlx:classAtom>
    <owlx:Class owlx:name="#CIM_ComputerSystem" />
    <ruleml:var>printer</ruleml:var>
  </swrlx:classAtom>
  <swrlx:classAtom>
    <owlx:Class owlx:name="#CIM_AuthorizedPrivilege" />
    <ruleml:var>privilege</ruleml:var>
  </swrlx:classAtom>
  <swrlx:datavaluedPropertyAtom swrlx:property="#Dedicated">
    <ruleml:var>printer</ruleml:var>
    <owlx:DataValue
      owlx:datatype="xsd:string">Print</owlx:DataValue>
  </swrlx:datavaluedPropertyAtom>
  <swrlx:individualPropertyAtom swrlx:property="#CATPrivilege">
    <ruleml:var>authtarget</ruleml:var>
    <ruleml:var>privilege</ruleml:var>
  </swrlx:individualPropertyAtom>
  <swrlx:individualPropertyAtom swrlx:property="#TargetElement">
    <ruleml:var>authtarget</ruleml:var>
    <ruleml:var>printer</ruleml:var>
  </swrlx:individualPropertyAtom>
  <swrlx:individualPropertyAtom swrlx:property="#CASPrivilege">
    <ruleml:var>authsubject1</ruleml:var>
    <ruleml:var>privilege</ruleml:var>
  </swrlx:individualPropertyAtom>
  <swrlx:individualPropertyAtom swrlx:property="#PrivilegedElement">
    <ruleml:var>authsubject1</ruleml:var>
    <owlx:Individual owlx:name="#ST_PHY" />
  </swrlx:individualPropertyAtom>
</ruleml:_body>
<ruleml:_head>
  <swrlx:classAtom>
    <owlx:Class owlx:name="#CIM_AuthorizedSubject" />
    <ruleml:var>authsubject2</ruleml:var>
  </swrlx:classAtom>
  <swrlx:individualPropertyAtom swrlx:property="#CASPrivilege">
    <ruleml:var>authsubject2</ruleml:var>
    <ruleml:var>privilege</ruleml:var>
  </swrlx:individualPropertyAtom>
  <swrlx:individualPropertyAtom swrlx:property="#PrivilegedElement">
    <ruleml:var>authsubject2</ruleml:var>
    <owlx:Individual owlx:name="#ST_COMP" />
  </swrlx:individualPropertyAtom>
</ruleml:_head>
</ruleml:imp>
```

Figure 4. SWRL representation of an authorization policy

This SWRL rule together with the ontology can be load into a rule reasoner, for example into the Jena reasoner (the one used during this research work). Figure 5 shows the Jena representation of this authorization policy. Other reasoners such as Pellet or CLIPS can also be used.

```
#exampleRule:
( ?authtarget http://www.w3.org/1999/02/22-rdf-syntax-ns#type
  http://www.positif.com/cim#CIM_AuthorizedTarget )
( ?authsubject1 http://www.w3.org/1999/02/22-rdf-syntax-ns#type
  http://www.positif.com/cim#CIM_AuthorizedSubject )
( ?printer http://www.w3.org/1999/02/22-rdf-syntax-ns#type
  http://www.positif.com/cim#CIM_ComputerSystem )
( ?privilege http://www.w3.org/1999/02/22-rdf-syntax-ns#type
  http://www.positif.com/cim#CIM_AuthorizedPrivilege )

( ?printer http://www.positif.com/cim#Dedicated 'Print' )
( ?authtarget http://www.positif.com/cim#CATPrivilege ?privilege )
( ?authsubject1 http://www.positif.com/cim#CASPrivilege ?privilege )
( ?authsubject1 http://www.positif.com/cim#PrivilegedElement
  http://www.positif.com/cim#ST_PHY )
->
( ?authsubject2 http://www.w3.org/1999/02/22-rdf-syntax-ns#type
  http://www.positif.com/cim#CIM_AuthorizedSubject )
( ?authsubject2 http://www.positif.com/cim#CASPrivilege ?privilege )
( ?authsubject2 http://www.positif.com/cim#PrivilegedElement
  http://www.positif.com/cim#ST_COMP )
]
```

**Figure 5. Jena representation of an authorization policy**

The rule reasoner infers new data that grants the privilege to print by the system to the students of Computer Science. Figure 6 shows the representation OWL-CIM of the data inferred by rule reasoner.

```
<CIM_AuthorizedSubject rdf:about="#authsubject2">
  <CASPrivilege rdf:resource="#printauth" />
  <PrivilegedElement rdf:resource="#ST_COMP" />
</CIM_AuthorizedSubject>
```

**Figure 6. Data inferred by rule reasoner**

## 6. Conclusions and Future Work

This paper describes a semantic approach for the representation of security policies based in OWL and SWRL. It emphasizes two main ideas. The first one is that, due to the use of OWL it is possible to perform some reasoning over the knowledge model. The second one is that, due to the use of SWRL, it is possible to perform also reasoning over the rules to correctly handle system management policies. We believe that this approach for system management is a strong bid for the future of distributed and complex systems like, for example, Web services based systems, multi-agent systems and peer to peer systems.

Currently, we have a preliminary version of the OWL reasoner and rule-based reasoner for security policies represented by the combination of OWL-CIM ontology and SWRL. Our deployment is based in the use of Jena 2 inference subsystem. The

CIM model has still work to be done on the completion of the whole model as it is very complex in terms of the number of modelling entities involved. We are also developing a graphical editor for policies. This represents ontologies as hierarchies of elements, and rules may be easily defined by just selecting elements on the ontology to be part either of the body or of the head or each rule. This is very convenient as the operator (i.e., the person which is in charge of defining and administrating policies) does not need to be familiar with technological aspects like ontologies and automated reasoning. He/she only needs to know about the knowledge model shown by the ontology. Moreover, we are also working to determine if the power of SWRL is enough to capture all the policy concerns.

## Acknowledgements

The work presented in this article has been partially funded by the EU in the context of the POSITIF (Policy-based Security Tools and Framework) IST project (IST-2002-002314) and also by the ENCUESTRO (00511/PI/04) Spanish Seneca project.

## References

1. García, F. J., G. Martínez, O. Cánovas, and A.F. Gómez-Skarmeta: A Proposal of a CIM-Based Policy Management Model for the OGSA Security Architecture, GADA Workshop, OTM Workshops 2004, 10/2004.
2. Common Information Model (CIM) Standards, DMTF, <http://www.dmtf.org/standards/cim>, WWW, 2005.
3. García, F. J., G. Martínez and A. F. Gómez-Skarmeta, A Semantically-Rich Management System based on CIM for the OGSA Security Services, In Knowledge and Data Mining Grid Workshop, 3rd Atlantic Web Intelligence Conference, 6/2005.
4. SWRL: A Semantic Web Rule Language Combining OWL and RuleML, The Rule Markup Initiative, <http://www.ruleml.org/swrl/>, WWW, 2005.
5. Jena – A Semantic Web Framework for Java, <http://jena.sourceforge.net/>, WWW, 2005
6. Fensel, D. Ontologies: Silver Bullet for Knowledge Management and Electronic Commerce. Springer-Verlag, 2004.
7. Smith, M.K., and C. Welty and D.L. McGuinness. OWL Web Ontology Language Guide. W3C Recommendation. W3C, 2004.
8. Baader F., D. Calvanese , D.L. McGuinness, D. Nardi, and P.F. Patel-Schneider. An Introduction to Description Logics, In Description Logic Handbook. Cambridge University Press, 2002.