ONTOLOGY-BASED KNOWLEDGE ACQUISITION SYSTEM FOR PRODUCT LIFE CYCLE TASK

Darya Plinere and Arkady Borisov

Riga Technical University
Department of Modelling and Simulation, Institute of Information Technology
1 Kalku Str., LV-1658, Riga
Latvia
to.darya@inbox.lv, Arkadijs.Borisovs@cs.rtu.lv

Abstract: *The product life cycle concept suggests that a product passes through four stages of evolution: introduction, growth, maturity and decline. As the product evolves and passes through these four stages, profit is affected, and different strategies have to be employed to ensure that the product is a success within its market, therefore there is a need to define in time at which of the stages the product is at the moment. We propose ontology-based knowledge acquisition system for product life cycle stage definition. Knowledge acquisition is the transformation of knowledge from the forms in which it exists into forms that can be used in a knowledge based system. The proposed knowledge system development sequence is the following: first, concepts and their relationships are defined by an ontology. Second, the domain experts enter their knowledge of the domain area using the domain-specific knowledge acquisition tool. Finally, problem-solving techniques are used to answer questions and problems of the domain using the knowledge base.*

Keywords: *ontology, knowledge acquisition, product life cycle, stage definition, Protégé, JessTab*

1. Introduction

The product life cycle stage definition issue is quite common nowadays. Company's profit depends at which of the stages the product is. It is not easy to tell which stage the product is at. The maturity stage is a good stage for every product, but wrong manager decision can change the stage to decline.
There are different strategies for each product life cycle stage in order to maximize profit and to delay decline stage; therefore there is a need to find a solution for product life cycle stage definition. This would help manager to take correct decisions in order to optimize company's work.
A domain ontology in Protégé and inference rules in Jess are combined into a knowledge system in which the elicitation and the validation of domain facts can be done at the same time. Ontology-based knowledge system development makes it possible to elicit domain facts after the implementation of the ontology-structured knowledge system [1].
This paper consists of 4 sections and it is organized as follows: Section 2 discusses product life cycle definitions and characteristics, Section 3 addresses the knowledge system (KS) development – classic and ontology-based KS development, Section 4 implements issues using Protégé and JessTab.

2. Product life cycle and its stage characteristics

The product life cycle concept suggests that a product passes through four stages of evolution: introduction, growth, maturity and decline. As a product evolves and passes through these four stages profit is affected and different strategies (see [2,3]) have to be employed to ensure that the product is a success within its market (see Figure 1).

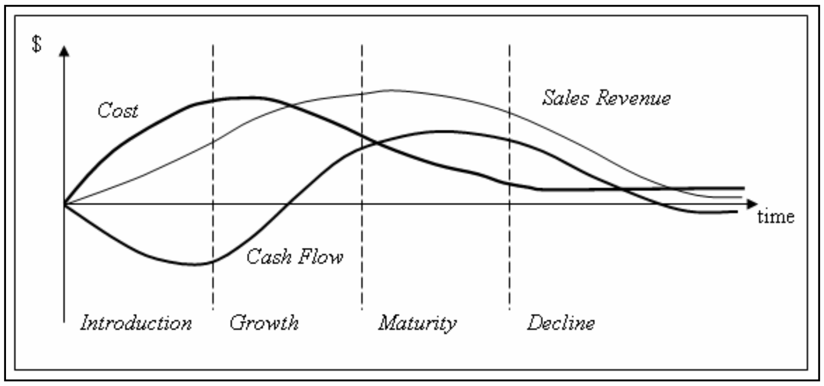


Fig. 1. Product life cycle

The product life cycle stage characteristics are:

1. Introduction – the product is introduced to customers. Sales are low, development and manufacture costs are high. Profit is negative due to this reason.
2. Growth – this stage is a period of rapid sales growth. Competitors see the opportunity and enter the market, some just copy the most successful product, or try to improve it to compete better. The profit grows.
3. Maturity – sales slow down as the product sales reach peak as it has been accepted by most buyers. Product lines are widened.
4. Decline – long-run drop in sales and falling demand are the main characteristics of this stage. A few small specialty firms may still manufacture the product.

Some examples are presented where the well-known products are currently at different stages of the product life cycle:

1. Introduction: Third generation mobile phones, E-conferencing;
2. Growth: Portable DVD Players, E-mail;
3. Maturity: Personal Computers, Faxes;
4. Decline: Typewriters, Handwritten letters [2].

According to [3, 4], in reality very few products follow such prescriptive cycle. The length of each stage varies enormously. The decisions of marketers can change the stage, for example from maturity to decline by price-cutting. It is not easy to tell which stage the product is in. Companies often try to use extension strategies in order to try to delay the decline stage of the product life cycle. The maturity stage is a good stage for the company in terms of generating cash. The costs of developing the product and establishing it in the market are paid and it tends to be then at a profitable stage. The longer the company can extend this stage, the better it will be for them. [3, 4]

3. Knowledge system development

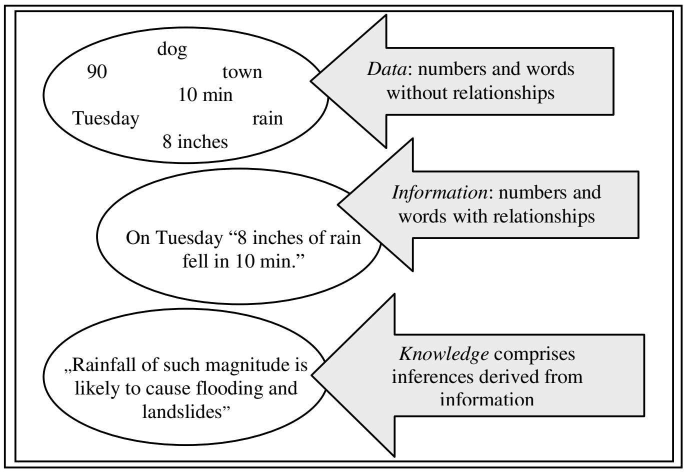

dog
90
town
10 min
Tuesday
rain
8 inches
Data: numbers and words without relationships
Information: numbers and words with relationships
On Tuesday "8 inches of rain fell in 10 min."
Knowledge comprises inferences derived from information
„Rainfall of such magnitude is likely to cause flooding and landslides"

Fig. 2. Data, information and knowledge

Data are defined as numbers and words without relationships. In reference to Figure 2, the words "town", "dog", "Tuesday", "rain", "inches", and "min", have little if any meaning without relationships. However, linked together in the sentence, "On Tuesday, 8 inches of rain fell in 10 min." they become *information*. If we then add the context of a particular geographical region and historical climatic records, we could perhaps infer that "Rainfall of such magnitude is likely to cause flooding and landslides." This becomes *knowledge* [5].

This section describes the development of the knowledge system (KS). The section is organized as follows: section 3.1. describes classical knowledge development, section 3.2. describes ontology-based knowledge system development and section 3.3. gives an idea of the proposed knowledge system development.

3.1. Classical knowledge system development

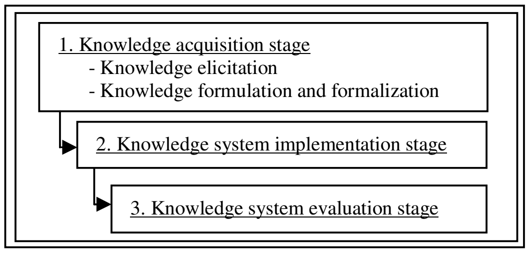

1. Knowledge acquisition stage
- Knowledge elicitation
- Knowledge formulation and formalization
2. Knowledge system implementation stage
3. Knowledge system evaluation stage

Fig. 3. Classical KS development

Knowledge acquisition is the transformation of knowledge from the forms in which it exists into forms that can be used in a knowledge based system (KBS). [6]
Classical knowledge system development takes place in three stages (see Figure 3).
The first is the knowledge acquisition stage, which subsumes the processes of "eliciting, analyzing, and formalizing the patterns of thought underlying some subject matter." After knowledge is elicited, concepts are analyzed with the goal of formalizing the domain in such a way that it can be used by a problem solver to reason about the domain. In the second stage, knowledge rules are extracted from the formalized knowledge and implemented in a knowledge system. Finally, the system is evaluated and the cycle of acquiring, implementing and evaluating is iterated [1].

3.2. Ontology-based knowledge system development
An *ontology* is a specification of a conceptualization. [7]
The ontology is an attempt of universal and detailed formalization of some field of knowledge by means of the conceptual scheme. Usually such scheme consists of the hierarchical data structure containing all relevant classes of objects, their communication and rules (theorems, restrictions) accepted in this area.
Usually ontologies consist of instances, concepts, attributes and relations:
1. instances- the basic, low-level components of ontologies. Instances can be represented as physical objects (people, houses, planets), and abstract (numbers, words).
2. concepts- abstract groups, collections or sets of objects. They can include individuals, other classes, or combinations.
3. attributes- each attribute has at least a name and value and is used for the object specific information storage.
4. relations- the important role of attributes consists in defining dependences (relations) between objects of the ontology. Usually the relation is the attribute whose value is other object. [8]

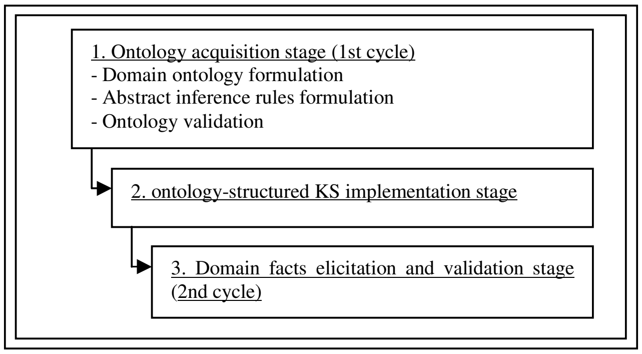

1. Ontology acquisition stage (1st cycle)
- Domain ontology formulation
- Abstract inference rules formulation
- Ontology validation
2. ontology-structured KS implementation stage
3. Domain facts elicitation and validation stage (2nd cycle)

Fig. 4. Ontology-based KS development

The first stage in the first cycle is the ontology acquisition stage in which the ontology of the domain of interest is built (see Figure 4). This stage is similar to the knowledge acquisition stage in classical development, with the difference that an ontology of domain facts is obtained and not yet the domain facts themselves. Furthermore, the ontology of the abstract inference rules is elicited. Both ontologies are refined and formulated in a (semi)formal language.
In the implementation stage the ontology of the domain facts is implemented in an ontology tool and the abstract inference rules are implemented in a rule engine.
The elicitation and validation of domain facts stage is the third stage and the second development cycle. (For further information refer to [1]).

3.3. Proposed ontology-based knowledge acquisition system development
The knowledge acquisition process in Protégé consists of three steps:
1. A class and its template slot have to be defined;
2. The form to acquire the instances of the class has to be laid out.
3. The class instances are acquired.
Each class has an associated form and is used to get the instances of the class [9, 11].
The proposed knowledge system development sequence is the following: first, concepts and their relationships are defined by an ontology. Second, the domain experts enter the information of the domain area using the domain-specific knowledge acquisition tool. Finally, problem-solving techniques are used to answer questions and problems of the domain using the knowledge base.

4. Implementation
This section describes the implementation of product life cycle problem solving technique using the Protégé tool. The section is organized as follows: section 4.1. discusses Protégé knowledge-base framework, section 4.2. describes Jess (Java Expert System Shell) and JessTab – plug-in for Protégé and section 4.3. gives an overview of the tool functionality.

4.1. Protégé

The ontology has been implemented in the Protégé ontology tool from Stanford University. Protégé is a free, open source ontology editor and knowledge–base framework. (For further information refer to [10])

The knowledge model of Protégé builds on open knowledge-base connectivity (OKBC), a language for frame-based systems. A frame is a data structure to represent typical constructs in the description of the application. Figure 5 shows a screenshot of the user interface of Protégé.

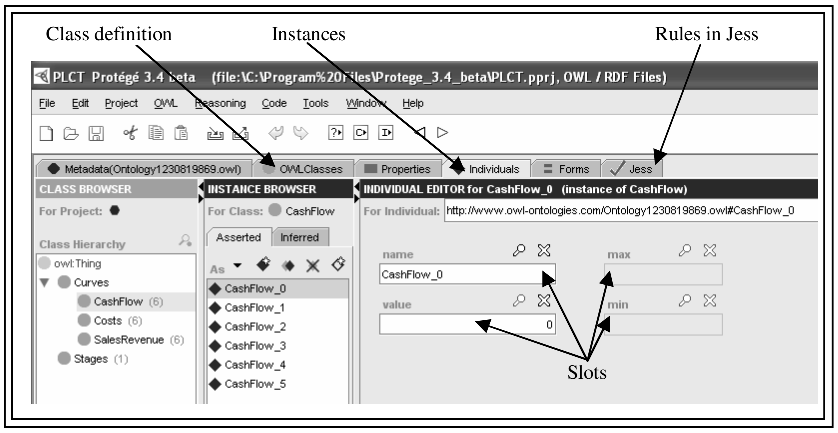

Class definition
Instances
Rules in Jess
PLCT Protégé 3.4 beta (file:\C:\Program%20Files\Protege_3.4_beta\PLCT.pprj, OWL / RDF Files)
File Edit Project OWL Reasoning Code Tools Window Help
Metadata(Ontology1230819869.owl) OWLClasses Properties Individuals Forms Jess
CLASS BROWSER
INSTANCE BROWSER
INDIVIDUAL EDITOR for CashFlow_0 (instance of CashFlow)
For Project:
For Class: CashFlow
For Individual: http://www.owl-ontologies.com/Ontology1230819869.owl#CashFlow_0
Class Hierarchy
owl:Thing
Curves
CashFlow (6)
Costs (6)
SalesRevenue (6)
Stages (1)
Asserted Inferred
CashFlow_0
CashFlow_1
CashFlow_2
CashFlow_3
CashFlow_4
CashFlow_5
name
CashFlow_0
value
0
max
min
Slots

Fig. 5. The representation of Protégé

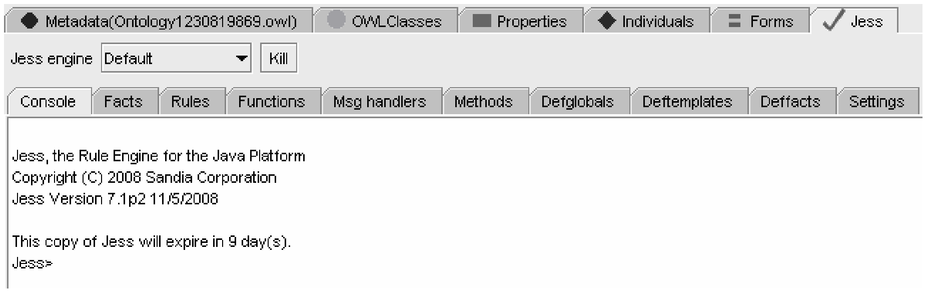
A Protégé ontology consists of classes, slots, and facets. Classes and slots are frames. Classes in Protégé constitute a taxonomic hierarchy. Slots attached to frames describe properties of that particular frame. Facets describe properties and restrictions of slots (i.e., slot type). (For further information refer to [11]). The CashFlow_0 frame above has four slots - name, value, max and min.

4.2. Jess and JessTab

Jess is a rule engine and scripting environment written entirely in Sun's Java language by Ernest Friedman-Hill at Sandia National Laboratories in Livermore, CA. Using Jess, you can build Java software that has the capacity to "reason" using knowledge you supply in the form of declarative rules. Jess is small, light, and one of the fastest rule engines available. Its powerful scripting language gives you access to all of Java's APIs. Jess includes a full-featured development environment based on the award-winning Eclipse platform. (For further information refer to [12]).

JessTab is a plug-in for Protégé that allows you to use Jess and Protégé together. It gives you the best of both worlds. JessTab provides a Jess console window where you can interact with Jess while running Protégé. Furthermore, JessTab extends Jess with additional functions that allow you to map Protégé knowledge bases to Jess facts. Also, there are functions for manipulating Protégé knowledge bases from Jess [13].

Using JessTab, programs that manage Protégé knowledge bases can be developed directly, and rules can be written that fire by matching patterns in the knowledge base. It is possible to use JessTab as an object-oriented extension to Jess by defining Protégé classes and instantiating them. Since Protégé has explicit metaclasses you can map them to facts as well and perform pattern matching on properties of classes (see Figure 6).


Metadata(Ontology1230819869.owl) OWLClasses Properties Individuals Forms Jess
Jess engine Default Kill
Console Facts Rules Functions Msg handlers Methods Defglobals Deftemplates Deffacts Settings

```
Jess, the Rule Engine for the Java Platform
Copyright (C) 2008 Sandia Corporation
Jess Version 7.1p2 11/5/2008

This copy of Jess will expire in 9 day(s).
Jess>
```

Fig. 6. JessTab console in Protégé

4.3. The functionality of the approach

The concept of proposed system is shown in Figure 7.

Being based on proposed knowledge system development, the first step is to define classes and their relationships by an ontology.

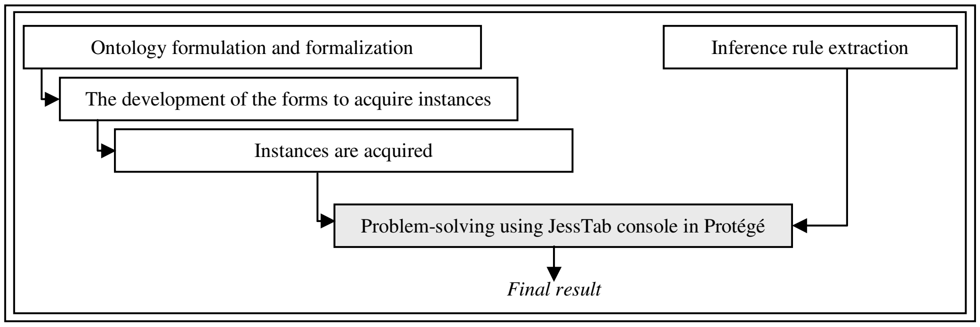
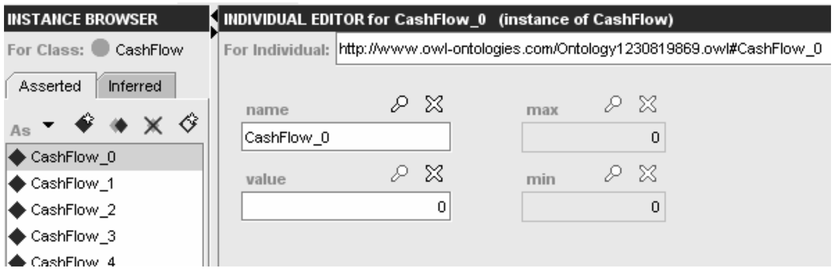

Ontology formulation and formalization
Inference rule extraction
The development of the forms to acquire instances
Instances are acquired
Problem-solving using JessTab console in Protégé
Final result

Fig. 7. The concept of proposed system

The domain area consists of two main classes - *Stages* and *Curves*, the *Stages* class has asserted instance *result* with slots *Costs*, *SRevenue* and *Cash*, and *Final_Result* as well. *Costs*, *SRevenue* and *Cash* types are strings and they represent the movement of the curves, i.e., „grows". The *Final_Result* slot represents the final answer of problem solving, the type of *Final_Result* slot is string, and the forms of all slots of *result* are read-only – the user cannot edit these fields.
The class *Curves* has subclasses: *Costs*, *SalesRevenue* and *CashFlow*. These in turn have asserted instances with slots *name*, *value*, maximum value (*max*) and minimum value (*min*). The curves are splitted in time, i.e., instance *CashFlow*_0 is a point of the curve in initial time (time=0) with the value=0. *Max* and *min* values are read-only, the rules written in JessTab console can edit these values.
These classes can be asserted using Protégé, or using JessTab console in Protégé:

(defclass Curves (is-a :THING) (slot name (type string)) (slot value (type integer)) (slot max (type integer)) (slot min (type integer)))
(defclass Stages (is-a :THING) (slot Cost (type string)) (slot SRevenue (type string)) (slot Cash (type string)) (slot Final_Result (type string)))

The second step is the following: the domain experts enter the values of curves using the domain-specific knowledge acquisition tool provided by Protégé (see Figure 8). The values can also be asserted using JessTab console in Protégé by loading facts from a file.



For Individual: http://www.owl-ontologies.com/Ontology1230819869.owl#CashFlow_0

CashFlow_0
CashFlow_1
CashFlow_2
CashFlow_3
CashFlow_4

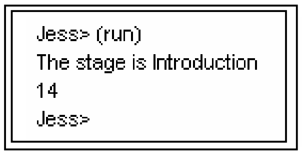
name	CashFlow_0
value	0
max	0
min	0

Fig. 8. The values are entered in Protégé forms.

Finally, problem-solving technique (JessTab) is used to answer questions and problems of the domain using the knowledge base.
The inference rules of this domain are represented in JessTab format, in addition it should be told that the construction of Jess rules and JessTab rules is a little bit different. The rule construction is „IF ... THEN", i.e., „If the cost of development and manufacture grow and sales revenue grow, and cash flow falls, then the stage of product is introduction". In JessTab console in Protégé this rule looks like:

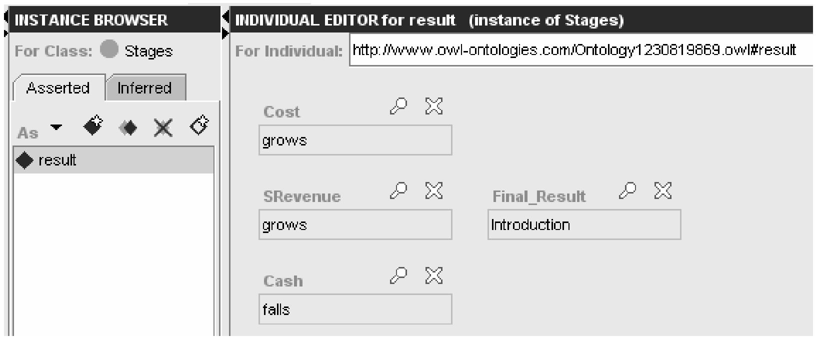
(defrule stage_definition1 (object (is-a Stages) (Cost "grows") (SRevenue "grows") (Cash "falls")) => (slot-set result Final_Result "Introduction") (printout t "The stage is Introduction" crlf))

The other rules for this domain are much more difficult. In order to find the movement of the curves, initial and last ones values are used in rules, and maximum and minimum values have to be found for these rules as well.
The screenshots of JessTab console and Protégé window with the final result are shown in Figures 9 and Figure 10.



```
Jess> (run)
The stage is Introduction
14
Jess>
```

Fig. 9. The answer of product life cycle stage definition („The stage is Introduction") after execution of the rules. "14" – is the quantity of fired rules for problem solving



For Individual: http://www.owl-ontologies.com/Ontology1230819869.owl#result

result

Cost: grows

SRevenue: grows

Final_Result: Introduction

Cash: falls

Fig. 10. Final_Result is visible now in Protégé – Introduction

Figure 10 shows the instance *result* with fulfilled fields of its slots. The final result was found after 14 rules fired- one of them was showed above, rules for maximum and minimum values were fired as well, and also the others rules in order to find the movemements of the curves. After finding the movements of the curves it was easy to find the stage, here it was "Introduction".

5. Conclusion

The product life cycle stage definition task is quite common nowadays, different techniques are applied in order to find at which of the stages the product is at the moment. The proposed system has shown its functionality for the considered issue and can be used for other issues as well. The main idea was to structurize all the information about product life cycle stage definitions into the ontology and then to apply problem solving technique to the acquired knowledge system. Here, for ontology development Protégé 3.4. beta was used and for the problem solving JessTab was used. JessTab is plug-in for Protégé. Jess uses an enhanced version of the Rete algorithm to process rules. Rete is a very efficient mechanism for solving the difficult many-to-many matching problem. Jess has many unique features including backwards chaining and working memory queries, and of course Jess can directly manipulate and reason about Java objects. The rules were represented in IF… THEN form; they used the knowledge base (ontology with asserted instances) for problem solving. The ontology-based knowledge acquisition system has shown good results, therefore it will be used in future researches for product life cycle and other issues as well.

References

[1] H. J. Lebbink, C. L. M. Witteman and J. -J. Ch. Meyer, Ontology-based knowledge acquisition for knowledge systems. In Blockdeel, H. & Denecker, M. (Ed.), *Proceedings of the 14th Dutch-Belgian Artificial Intelligence Conference (BNAIC'02)*, pp. 195-202, Leuven, Belgium.
[2] D. Rudenko, A. Borisov, Blackboard architecture for product life cycle stage definition, *MENDEL 2008 - 14th International Conference on Soft Computing,* pp.252-257, 2008, Brno, Czech Republic.
[3] Marketing teacher, Est 2000, http://www.marketingteacher.com/Lessons/lesson_plc.htm, Last visit day May 2009.
[4] Online postgraduate courses for the electronic industry, Design and Manufacture of Electronic Systems, Engineering Design, http://www.ami.ac.uk/courses/ami4900_ed/u01/unit_1_sec_2/index.asp, Last visit day May 2009.
[5] J. G. Pohl, Transition From Data to Information, *Collaborative Agent Design Research Center Technical Report - RESU72,* 2001, pp 1-8.
[6] http://www.absoluteastronomy.com Exploring the universe of knowledge. AbsoluteAstronomy.com is a free online information portal which provides reference information and interactive features for over 900,000 topics. AbsoluteAstronomy.com was founded in 1999.
[7] T. R. Gruber, A translation approach to portable ontologies, *Knowledge Acquisition*, 5(2), 1993, pp. 199-220.
[8] http://dic.academic.ru/dic.nsf/ruwiki/16269 Dictionaries and enciclopedias on Academic (in Russian).
[9] M.S. Abdullah, I. Benest, A. Evans and C. Kimble, Knowledge Modelling Techniques For Developing Knowledge Management Systems, *3rd European Conference on Knowledge Management*, Dublin, Ireland, 2002, pp. 15-25.
[10] http://protege.stanford.edu/ Protégé is a national resource for biomedical ontologies and knowledge bases supported by the National Library of Medicine.
[11] N. Friedman Noy, R. W. Fergerson, and M. A. Musen. The knowledge model of Protege-2000: Combining interoperability and flexibility, *Proceedings of the 12th International Conference on Knowledge Engineering and Knowledge Management (EKAW'2000)*, 2000, pp 69-82.
[12] http://www.jessrules.com/jess/index.shtml Jess, the Rule Engine for the JavaTM Platform.
[13] http://www.ida.liu.se/~her/JessTab/ Henrik Eriksson - JessTab: Integrating Protégé and Jess.