

Deanship of Graduate Studies
Al-Quds University



Ontology Based Business Rules Extraction Model & Algorithm (OBBREMA)

نموذج وخوارزمية استخراج قواعد الأعمال باستخدام الأنتولوجى

Maha Fawzy Osrof

M.Sc. Thesis

Jerusalem – Palestine

1432 / 2011

Deanship of Graduate Studies
Al-Quds University



Ontology Based Business Rules Extraction Model & Algorithm (OBBREMA)

نموذج وخوارزمية استخراج قواعد الأعمال باستخدام الأنتولوجى

Prepared By:
Maha Fawzy Osrof

Bsc. from Applied Science University, Jordan

Supervisor:
Dr. Raid Al-Zaghal

A thesis Submitted in Partial fulfillment of requirements for the Master degree of Computer Science from Computer Science department of Al-Quds University

Jerusalem – Palestine
1432 / 2011

Deanship of Graduate Studies
Al-Quds University



Thesis Approval

Ontology Based Business Rules Extraction Model & Algorithm(OBBREMA)

نموذج وخوارزمية استخراج قواعد الأعمال باستخدام الأنتولوجي

Prepared By: Maha Fawzy Osrof
Registration No: 20714018

Supervisor: Dr. Raid ALzaghal

Master thesis submitted and accepted. Date: 11/6/2011

The names and signatures of the examining committee members are as follows:

- | | |
|--|------------------|
| 1- Head of Committee: Dr. Raid al-Zaghal | Signature: |
| 2- Internal Examiner: Dr. Jehad Najjar | Signature: |
| 3- External Examiner: Dr. Radwan Tahboub | Signature: |

Jerusalem – Palestine
1432 / 2011

Dedication

*To my parents, who guided me to success with their wisdom,
To my husband, who supports and stands beside me,
To my kids, who I hope this thesis will be a motivation for them,
To my brothers and sisters,
To my colleagues and friends at Al-Quds University*

Maha Fawzy

Declaration

I certify that this thesis submitted for the degree of Master is the result of my own research, except where otherwise acknowledged, and that this thesis (or any part of the same) has not been submitted for a higher degree to any other university or institution.

Signed

Maha Fawzy Osrof

Date: 11/6 /2011

Acknowledgement

First of all, praises and thanks always to Allah, The creator, who gave me the power to complete this work.

Then, I'm heartily thankful for my supervisor Dr.Raid Al-Zaghal, without his guidance, support, and encouragement, this thesis will not have been completed.

Also, I owe my deepest gratitude to all professors and teachers at Al-Quds University/Computer Science department for their support. And I want to thank Dr.Badie Sartawi for his encouragement to our research class.

Lastly, I express my deepest thanks and blessing for my father, mother, husband brothers and sisters for supporting.

Maha Fawzy

Abstract

Software Engineers and developers need Business Rules to complete analysis process and developing applications consequently. Business Rule is a statement that defines or constrains some aspect of the business. Extracting Business Rules from legacy systems is a difficult process, since Business Rules are hidden in the code. And legacy systems keep changing all the time. In addition to that, many steps are needed to extract Business Rules from large systems, and it is not worthy in small systems. We suggest in this thesis to use Ontology, as a conceptual model that represent Business Rules expressively, for extracting Business Rules to solve extraction problems. First of all, we did a mapping using analysis and comparison between Business Rules Categories and Ontology Concepts to determine what exactly to extract. The case studies show how Ontology represents expressive and real world Business Rules and they help us in determining relationships between Ontology concepts. Our own case study was implemented in the qualified teacher domain, where we applied different types of Business Rules to implement the mapping.

Then we propose the Ontology Based Business Rules Extraction Model (OBBREM) that extracts Business Rules from Ontology depending on our one to one mapping and the case studies.

Finally we propose a translation for our model into an extraction algorithm Ontology Based Business Rules Extraction Algorithm (OBBREA) using backtracking analysis for the case studies. This algorithm helps in extracting Business Rules from Ontology in expressive way to help software engineers and analysts in the analysis process. Also this algorithm can be implemented with a parser in the future to fulfill the extraction from Web Ontology Language (OWL) code.

المخلص

يحتاج مهندسى البرمجيات ومطورى البرامج لقواعد الأعمال لاتمام عملية التحليل وتطوير التطبيقات المعمول بها. قواعد الأعمال هى التى تعرف وتحدد جوانب وشروط العمل. ان استخلاص قواعد الأعمال من الأنظمة القديمة المتوارثة عملية صعبة، لأن قواعد الأعمال تكون مخفية فى كود البرامج، كما أن الأنظمة المتوارثة تتغير طيلة الوقت. بالاضافة لذلك يلزم العديد من الخطوات لاستخلاص قواعد الأعمال من الأنظمة الكبيرة وهى عملية لا تستحق الجهد فى الأنظمة الصغيرة.

نقترح فى هذه الأطروحة استخدام الأنتولوجى -كنموذج مفاهيمى يمثل قواعد الأعمال بطريقة معبرة- لاستخلاص قواعد الأعمال مما يساعد فى حل مشاكل استخلاصها. فى البدء عملنا مقابلة بين تصنيفات قواعد الأعمال ومفاهيم الأنتولوجى لمعرفة ماذا سيتم استخلاصه بالضبط، حيث وجدنا مقابل كل تصنيف لقواعد الأعمال مفهوم يقابله فى الأنتولوجى نتيجة لعملية التحليل والمقارنة بين الطرفين.

ثم بينا فى الحالات الدراسية كيف تمثل الأنتولوجى قواعد الأعمال بطريقة معبرة وواقعية، ومن ضمن الحالات الدراسية الحالة التى نفذناها فى مجال المعلمين المؤهلين وتم تطبيق أنواع مختلفة من قواعد الأعمال فيها لتنفيذ عملية المقابلة مع الأنتولوجى. ولقد ساعدت هذه الحالات الدراسية فى معرفة العلاقات التى تربط مفاهيم الأنتولوجى -التي نريد استخلاصها- ببعضها البعض. ثم قدمنا نموذج لاستخلاص قواعد الأعمال باستخدام الأنتولوجى يعتمد على المقابلة الأحادية-مفهوم مقابل مفهوم- السابقة والحالات الدراسية فى عملية استنتاج نموذج الاستخلاص. وفى النهاية قدمنا ترجمة لنموذجنا من خلال خوارزمية لاستخلاص قواعد الأعمال باستخدام الأنتولوجى تساعد فى استخلاص قواعد الأعمال بطريقة معبرة تساعد مهندسى البرمجيات والمحللين فى عملية التحليل وذلك بتطبيق تحليل باكتراكنج على الحالات الدراسية. وأيضا يمكن تنفيذ هذه الخوارزمية مع بارسر فى المستقبل لاتمام عملية الاستخلاص من لغة OWL البرمجة فى الأنتولوجى.

Table of Contents

Contents

Chapter 1.....	1
Problems in Business Rules Extraction and Motivations	1
1.1 Introduction	1
1.2 Problem Statement	1
1.3 Problems in Extracting Business Rules.....	2
1.4 Potential Uses of the Extracted Business Rules	3
1.5 Conclusion	4
Chapter 2.....	5
Literature Review	5
2.1 Introduction	5
2.2 Business Rules Literature	5
2.3 Ontology Literature.....	7
2.4 Conclusion	10
Chapter 3.....	11
Theoretical Framework.....	11
3.1 Introduction	11
3.2 Business Rules Definition	11
3.2.1 Business Rules Categories.....	12
3.2.2 Types of Business Rule:	13
3.3 Ontology Definition.....	14
3.3.1 Developing an Ontology.....	15
3.3.2 Why developing an Ontology?.....	15
3.4 What is OWL?.....	18
Chapter 4.....	19
Mapping Business Rules Categories to Ontology Concepts.....	19
4.1 Introduction	19
4.2 Mapping Business Terms to Ontology Concepts (Classes).....	19
4.3 Mapping Facts relating terms to each other to Ontology Object Properties	20
4.4 Mapping Constraints to Ontology Restrictions & Necessary and Sufficient Conditions.....	26

4.5 Mapping Derivations to Ontology Closure Axioms	28
4.6 Mapping Summary Table	29
4.7 Conclusion	30
Chapter 5.....	31
Case Studies	31
5.1 Introduction	31
5.2 Qualified Teacher Ontology	31
5.2.1 Qualified Teacher Main Classes	32
5.2.2 Object Properties and Quantifier Restrictions.....	37
5.2.3 Cardinality Restrictions & Necessary and Sufficient Conditions.....	38
5.2.4 Value Partition.....	44
5.2.5 Data Properties	49
5.2.6 A Closure Axiom and Covering Axiom.....	51
5.3 Pizza Ontology.....	53
5.3.1 Classes and Subclasses	53
5.3.2 Object Properties	53
5.3.3 Datatype Properties	54
5.3.4 has Value Restrictions	56
5.3.5 Cardinality Restrictions	57
5.3.6 Quantifier Restrictions	57
5.3.7 Value Partitions.....	58
5.3.8 Covering Axiom and Closure Axiom	58
5.4 Family Ontology	59
5.4.1 Classes and Subclasses.....	60
5.4.2 Object Properties	60
5.4.3 Datatype properties	61
5.4.4 hasValue Restrictions	61
5.4.5 Cardinality Restrictions	62
5.4.6 Quantifier Restrictions	62
5.4.7 Value Partitions.....	62
5.4.8 Covering Axiom and Closure Axiom	62
5.5 Conclusion.....	63

Chapter 6.....	64
Ontology-Based Business Rules Extraction Model & Algorithm	64
6.1 Introduction	64
6.2 Ontology Based Business Rules Extraction Model (OBBREM)	65
6.3 Ontology Based Business Rules Extraction Algorithm (OBBERA).....	70
6.4 Future Work	75
References	76

List of Tables

Table No	Table Name	Page
Table 4.1:	Mapping Business Rules Concepts To Ontology Concepts.....	29

List of Figures

Figure No.	Figure Name	Page
Figure 4.1 :	First Case of N-ary Relationship.....	22
Figure 4.2 :	Second Case of N-ary Relationship.....	23
Figure 4.3 :	Third Case of N-ary Relationship.....	24
Figure 4.4 :	Fourth Case of N-ary Relationship.....	25
Figure 5.1 :	Teacher Ontology Main Classes.....	32
Figure 5.2 :	Teacher Categories.....	33
Figure 5.3 :	AfterSchoolTeacher Category (Class).....	34
Figure 5.4 :	Business Rules for AfterSchoolTeacher Class.....	36
Figure 5.5 :	Inherited Object Properties for Subclasses of Teacher Class.....	36
Figure 5.6 :	Object Properties For Teacher Class.....	38
Figure 5.7 :	Necessary and Sufficient Conditions to Define KindergardenTeacher Class.....	39
Figure 5.8 :	Necessary and Sufficient Conditions for LowBasicLevelTeacher Class.....	39
Figure 5.9 :	Difference between Necessary Conditions and Necessary & Sufficient Conditions.....	40
Figure 5.10:	HighBasicLevelTeacher Subclasses.....	41
Figure 5.11:	Business Rules for HighBasicLevelTeacherProgramA Subclass.....	41
Figure 5.12:	Business Rules for HighBasicLevelTeacherProgramB Class.....	42
Figure 5.13:	Business Rules for KindergardenTeacher Class.....	42
Figure 5.14:	Business Rules for LowBasicLevelTeacher Class.....	43
Figure 5.15:	Business Rules for SecondaryT eacher Class.....	43
Figure 5.16:	Business Rules for ExpertTeacher Class.....	45
Figure 5.17:	Business Rules for NoviceTeacher Class.....	45
Figure 5.18:	Subclasses for Certificate Class.....	46

Figure No.	Figure Name	Page
Figure 5.19:	Subclasses for Grade Class.....	47
Figure 5.20:	Subclasses for Level Class.....	48
Figure 5.21:	Subclasses for Subject Class.....	48
Figure 5.22:	Attributes at individuals level.....	49
Figure 5.23:	Attributes at class level.....	50
Figure 5.24:	Expert Teacher Attribute (hasyearsofexperience) range.....	50
Figure 5.25:	Novice teacher Attribute (hasyearsofexperience) range.....	51
Figure 5.26:	A closure Axiom for Secondary Teacher (11-12) Level.....	52
Figure 5.27:	Covering Axiom for class PromotionValue.....	52
Figure 5.28:	Pizza Ontology Classes.....	53
Figure 5.29:	Object Properties for Cheesey Pizza Class.....	54
Figure 5.30	Data Properties at Individual Level.....	55
Figure 5.31	Data Properties at Class Level.....	55
Figure 5.32	Data Property Restriction.....	56
Figure 5.33	HasValue Restriction.....	56
Figure 5.34	Minimum Cardinality Restriction.....	57
Figure 5.35	Quantifier Restrictions.....	58
Figure 5.36	Spiciness Value Partion.....	58
Figure 5.37	Spiciness Covering Axiom.....	59
Figure 5.38	Margherita Closure Axiom.....	59
Figure 5.39	Classes and Subclasses of Family Ontology.....	60
Figure 5.40	hasParent Object Property.....	61
Figure 5.41	hasValue Restriction.....	61

Figure No.	Figure Name	Page
Figure 5.42	Minimum Cardinality Restriction.....	62
Figure 5.43	Covering Axiom.....	63
Figure 6.1 :	Ontology Based Business Rules Extraction Model (OBBREM).....	65

List of Appendices

Appendix No.	Appendix Name	Page
	Appendix 1: OWL Rendering XML Code for Qualified Teacher Ontology.....	83

Acronyms

BRMS	Business Rules Management Systems
OBBREM	Ontology Based Business Rules Extraction Model
OBBREA	Ontology Based Business Rules Extraction Algorithm
ORM	Object Role Model
IT	Information Technology
XML	Extensible Markup Language
W3C	World Wide Web Consortium
RDF	Resource Description Framework
DARPA	Defense Advanced Research Projects Agency
DAML	DARPA Agent Markup Language
ECA	Event Condition Action
DBMS	Database Management Systems
PAL	Protégé Axiom Language
SQL	Standard Query Language
OWL	Web Ontology Language
ERM	Entity Relational Model
BA	Bachelor
MA	Master

Chapter 1

Problems in Business Rules Extraction and Motivations

1.1 Introduction

The motivation for this thesis is to help Software Engineers and System Analysts in discovering and extracting relevant Business Rules for new applications that belong to a certain domain using the Ontology as a tool. Since Ontology is a conceptual model that expresses a certain domain in terms of a set of well-defined rules, it would be easier and faster to discover the Business Rules related to the target domain and reuse whatever is useful for the new application instead of beginning the analysis process from scratch to extract Business Rules for the new application. In addition, extracting such rules from legacy systems have many problems because legacy systems keep changing all the time as described by (Baxter and Hendryx, 2005, page3) “legacy systems need ongoing enhancement, desired functionality changes, must integrate with other systems”. Besides, in such systems, Business Rules are imbedded (hidden) inside the code so they become much difficult to extract.

In this thesis, we suggest that Ontology can be used as a tool to extract Business Rules for a certain domain by suggesting a model and algorithm for extracting Business Rules from Ontology.

1.2 Problem Statement

Can Ontology as a conceptual model be a solution for extracting Business Rules problems, and what exactly do we have to extract from Ontology?

1.3 Problems in Extracting Business Rules

Extracting Business Rules can be a difficult process because it depends on legacy systems that are changing all the time and they need to integrate with other systems. In addition, currently, extraction depends on complex and poorly documented software. Business Rules depend only on business vocabulary which is hidden in the code. Also extraction needs an interactive process with business analysts, they use clues in the code to extract Business Rules and they depend on their understanding of the business context to interpret these clues business vocabulary code. Extracting Business Rules is done in three steps. “First get business vocabulary, then build rules using vocabulary, interleave activities in practice” (Baxter, Hendryx, 2005, page 7). Also analysts need specific tools to extract clues from the code like compilers with level detail across application languages then they apply business judgment to formulate business vocabulary in English and record this vocabulary and code connections. Also business analysts need analysis-based code browser or annotation tools to identify Business Rules and write Business Rules in English.

Some engines for extracting and expressing Business Rules such as Business Rules Management Systems (BRMS) require that business analysts to be involved all the time to determine Business Rules which can be too much for small applications (Owen, 2004).

In (Ulrich, 2005) the author explained how extracting Business Rules needs several steps and a lot of effort done like interviews with business analysts, reviewing inputs and outputs, examining documentation to map business processes to various program clusters. Also transitioning from extracted business logic to a Business Rule involves an extracting, filtering and packaging process. This shows that extracting Business Rules is not an easy process.

Moreover, extracting Business Rules from code is not simple because Business Rules have different forms so to extract them first we have to classify them as input data, conditions, loops, access rights and so on (Eswaran, 2010).

1.4 Potential Uses of the Extracted Business Rules

There are several benefits and uses for extracted business rules, including:

- Should help in the analysis process by determining the requirements for certain applications.
- “Understand the functions (services) of the current system expressed in the in organization’s terms.
- Maintain Business Rules in plain English by the organization’s business analysts.
- Integrate legacy vocabulary and rules with new vocabulary and rules.
- Reuse in other applications and departments that use the same vocabulary and rules (from the same domain or that use the same data dictionary).
- Train new personnel on the vocabulary and rules.
- Support audits for regulatory compliance.
- Develop system requirements, design validation, and acceptance test specifications for systems based on the vocabularies and rules.” (Baxter and Hendryx, 2005, page 15)
- When Business Rules are not known very well, the developers can code them and return them back to business analysts for verification. The business analysts most of the times are not satisfied with the results so they review them over and over with the developers, after months they can accept the result they got. So extracting the right Business Rules will save time and effort.

- Business Rules can be reused in other applications.
- Business Rules can help in documentation of business decisions.
- Business Rules help in easy maintenance for the application.
- Business Rules make the maintenance cost of the application lower (Rosenberg and Dustdar, 2005).
- Having explicit Business Rules prevents loss of knowledge when employees leave the organization.

1.5 Conclusion

We will try to solve the problems associated with extracting Business Rules by suggesting the use of Ontology which represent Business Rules in expressive way. We will show how Ontology concepts can be mapped to Business Rules categories. Then suggest a model and an algorithm that are used to determine what exactly to extract from Ontology to get Business Rules.

Chapter 2

Literature Review

2.1 Introduction

Extracting Business Rules can be a difficult process that involves many steps. It requires a lot of business analysts and developers efforts. Also it costs long time consumed in extraction and coding. In the following, we summarize some related work for extracting Business Rules from software systems then we will explain how Ontology can be used to extract information and user requirements so it could be a suitable tool to extract Business Rules.

2.2 Business Rules Literature

In (Baxter, Hendryx, 2005), it is explained how extracting Business Rules can be a difficult process. They showed that extracting Business Rules from legacy systems is difficult since they keep changing all the time, and their software is complex and poorly documented. In addition to that, Business Rules are hidden in the code, so they used software tools to help them in extracting Business Rules in interactive process with the help of business analysts.

In (Ulrich, 2005), the author explained that extracting Business Rules needs many processes and steps such as interviewing with business analysts, reviewing system inputs and outputs, filtering business logic and packaging process.

In (Owen, 2004), the author explained that the basic law of technology is to get smart results when using smart tools by smart people and developers must not make misleading decisions about Business Rules and business analysts must do best rather than trying to translate business decisions into painfully detailed requirements documents. So they

explained that Business Rules Management Systems (BRMS) can be a bridge between IT developers and business analysts by letting business analysts determine the business logic because BRMS allows business analysts to see, understand, code and maintain Business Rules without help or with a little help of IT developers.

In (Joukhadar and Al-Maghout, 2006), the authors produce a multilingual solution to improve agility in business application by enabling the domain experts to specify Business Rules to the business application directly in many natural languages depending on Elixir MDA Framework and business model.

In (Demey, et al, 2002), the authors invented a new approach for modeling Business Rules called ORM-ML that represents ORM models textually and the syntax of the resulting model is marked-up by XML tags syntax.

In (Sneed and Erdos, 1996), the authors presented a method to extract Business Rules using data as the key that identify the rules since knowing outputs of a given Business Rule makes it possible to determine how those results were calculated and which arguments were used. So their method is backtracking from results till reaching Business Rules. The purpose is to determine for each application what statements change it and affect it and where these statements are located and under what conditions do they work. That is why they considered Business Rule composed of four elements: results, arguments, assignments and conditions.

In (Wang, et al, 2004), the authors produced a new framework to extract Business Rules from large complex legacy systems. This framework consists of five steps: slicing program,

identifying domain variables, data analysis, presenting Business Rules and business validation. And they applied their framework on a large financial legacy system.

In (Tobon and Franco, 2010), the authors developed a tool to extract Business Rules from process specifications written in natural language using a set of linguistic patterns and keywords in addition to grammatical heuristics. This tool includes a natural language parser for working out the grammatical structure of sentences in a documented specification.

2.3 Ontology Literature

In recent years, the development of Ontologies—explicit formal specifications of the terms in the domain and relations among them —(Noy and McGuinness, 2000) has been moving from the area of Artificial-Intelligence laboratories to the desktops of domain experts.

Ontologies have become common on the World-Wide Web. The Ontologies on the Web range from large taxonomies categorizing web sites (such as on Yahoo!) to categorizations of products for sale and their features (such as on Amazon.com). The WWW Consortium (W3C) is developing the Resource Description Framework (RDF), a language for encoding knowledge on Web pages to make it understandable to electronic agents searching for information. The Defense Advanced Research Projects Agency (DARPA), in combination with the W3C, is developing DARPA Agent Markup Language (DAML) by extending RDF with more expressive constructs aimed at facilitating agent interaction on the Web. Many regulations now develop standardized Ontologies that domain experts can use to share and annotate information in their fields. Medicine, for example, has produced large, standardized, structured vocabularies such as SNOMED and the semantic network of the Unified Medical Language System. Broad general-purpose Ontologies are emerging as well. For example, the United Nations Development Program and Dun & Bradstreet

combined their efforts to develop the UNSPSC ontology which provides terminology for products and services (www.unspsc.org) (Noy and McGuinness, 2000).

In prior work (Braun, et al, 2007), the authors explains that it is not enough to build an Ontology relying on specialized knowledge engineers only, because this will not reflect the real-world settings, that is why they introduce an Ontology Maturing methodology which takes advantage of expert users in addition to specialized engineers.

In another work (Honour, 2006), the authors suggested the Ontology as a methodology for measuring the correlation between the amount, types and quality of systems engineering efforts used during a program and the success of the program, so this measurement will yield more specific relationships between systems engineering activities, such as requirements management effort, and the cost/schedule compliance of the program.

In the work of (Chen, et al, 2006), the authors created an Ontology to represent user domain in a deposit system for a banking application so their results show that it can construct user-driven software for defending frequent requirement changes.

In (Breaux, 2006), the authors used an Ontology to provide means to identify and document certain and possibly conflicting interpretations of regulatory requirements.

Authors such (Saggion, 2007), introduce Ontology as a tool for extracting information for decision makers by collecting data from different applications in the same domain and summarize it after making sure it is for the same entity. This can help decision makers in business investment but it was a real problem for them to make sure that the related data is for the same entity.

In (Spyns, et al, 2002) the authors introduced Ontology as domain rules contain the semantics of concepts and conceptual relationships of a particular application domain on contrary to data model which represents the structure and integrity of the data elements of a specific application. So Ontology is a general Concept.

In (Bugaite, et al, 2005) the authors explained that Ontology represent a domain of real world concepts. So this domain must include Business Rules as a part of it. Their analysis of mathematical models of ontology and Business Rules shows that these models are compatible. Therefore, domain ontology can be used to elicit a set of Business Rules. They propose a framework, which can be used for the domain ontology axioms transformation into the Event-Condition-Action ECA rules and then into the active DBMS triggers.

In (Olegas, et al, 2007), the authors suggested an algorithm to transform Ontology Axioms to a rule model. This algorithm was applied to the transformation of a particular ontology axioms defined using Protégé Axiom Language (PAL) into rules presented in the form of SQL triggers.

In (Vasilecas and Būgaitė, 2005), the authors suggested that Ontology represents Business Rules since it represents real world domain. Their analysis showed that structural assertions are captured by Ontology Terms and Relationships and the other complex rules are represented by Ontology axioms. They showed this is true by Ontology mathematical Definition.

In (Hodrob and Jarrar, 2010) they represented a mapping between Object Role Model ORM which is a conceptual modeling language used in Ontology engineering. It contains

group of constraints can represent an Ontology using rich graphical notation. And OWL 2 DL Web Ontology Language to utilize benefits of both ORM and OWL 2 DL.

2.4 Conclusion

As we can see a lot of work has been done on Ontology but each one has focused on a separate idea like sharing user information, or using Ontology to know user requirements, or using Ontology to eliminate the ambiguity in interpretation of regulations ,or extracting information for business intelligence domain, but none of them have actually employed the Ontology as a conceptual model and used it for extracting Business Rules for specific domain to help Analysts and Software Engineers in analysis process, and to solve problems of extracting Business Rules. This is what we will explain and develop in this study.

Chapter 3

Theoretical Framework

3.1 Introduction

In this chapter, we will explain definitions of Business Rules and Ontology. And explain Business Rules categories and types. We will also explain how Ontology is developed and why. So we can understand these terms.

3.2 Business Rules Definition

“ A Business Rule is a statement that defines or constrains some aspect of the business. This must be either a term or fact (described as Structural Assertion), a constraint (described as Action Assertion), or Derivation. It is atomic in that it cannot be broken down or decomposed further into more detailed Business Rules. If reduced any further, there would be loss of important information about the business..” (The Business Rules Group,2000, page 14). Put it in another way “Business Rules define business polices which are specified by domain experts and input to the business application by programmers” (Joukhadar and Al-Maghout,2006, page1).

So Business Rules can express specific constraints on the creation updating and deleting constant data in an information system. There are different categorizations for Business Rules. In our thesis we consider the Business Rules Group categorization. A statement of Business Rule can be categorized into four categories: Business Terms, Facts relating terms to each other, Constraints, and Derivations (Business Rules Group, 2000). The definition of each Business Rule category is explained in the following subsection:

3.2.1 Business Rules Categories

1- Business Terms

The most basic element of a Business Rule is the language used to express it. The term is a Business Rule that describes how people think and talk about things. Terms are considered as entities in Entity Relational Model ERM or in glossaries.

2- Facts relating terms to each other

Facts are documented as relationships, attributes, and generalization structures in a graphical model or as natural language sentences. They can be described as facts that relate terms to each other. For example “Student pays school fees”.

3- Constraints

Constraints are used to control actions or behavior in business enterprise, they determine what data to be updated, deleted etc. they prevent a record from being created in certain conditions, or prevent an action from taking place.

4- Derivations

Information could be inferred or calculated from another information and this what we call derived information. Business Rule could be inferred or calculated from other Business Rule (term, fact, and other derivations) so it would be a derived Business Rule.

The different types of Business Rules are introduced in the coming subsection:

3.2.2 Types of Business Rule:

These types of Business Rules as explained in (Business Rules Group, 2000)

A Structural Assertion: defined concept or a statement of a fact that expresses some aspect of the structure of an enterprise. This includes terms and facts.

An Action Assertion: a statement of constraint or condition that controls the actions in an enterprise. It controls the results that produced by actions.

A Derivation: a statement of information that is derived from other information in the business. It could be inferred or calculated mathematically.

3.3 Ontology Definition

“The term “ontology” is borrowed from philosophy, where Ontology means a systematic account of Existence. Gruber T. introduced the most popular definition of ontology, where in the context of knowledge sharing, the term “ontology” means a specification of a conceptualization (Gruber, 1993). In other words, ontology is a description (like a formal specification of a program) of the concepts and relationships that are typical of an agent or a community of agents.

The subject of ontology is the study of the categories of things that exist or may exist in some domain (Sowa, 2000)” (Vasilecas & Būgaitė, 2005, page184).

There are many other definitions for Ontology it can be defined as “Shared understandings of a particular domain that have to be constructed within social processes among the stakeholders” (Braun, 2007, page 1). Also the Artificial-Intelligence literature contains many definitions of an Ontology; many of these disagree with each other. One of these definitions “ An Ontology is a formal explicit of concepts in a domain of discourse(classes(sometimes called concepts)), properties of each concept describing various features and attributes of the concept (slots(sometimes called roles or properties)), and restrictions on slots(facets(sometimes called role restrictions)). An Ontology together with a set of individual instances of classes constitutes a knowledge base. In reality, there is a fine line where the Ontology ends and the knowledge base begins” (Noy and McGuinness, 2000, page 3).

Classes are the most important part of Ontologies, because classes describe concepts in the domain. For example, in the domain of newspaper a class of Authors represents all

Authors. Specific authors are instances of this class. Mr. Ali in Alquds newspaper is an instance of the class of Authors. A class can have subclasses that represent concepts that are more specific than the superclass. For example, we can divide the class of all Authors into Editors, Reporters, and Columnists.

Slots describe properties and attributes of classes and instances; Mr. Ali has Hiring Date, Salary etc. His Salary value is 2000\$. We have two of slots describe Authors Hire Date, and Salary and we will have more slots to describe authors such as Name, Birth Date, Address and so on.

3.3.1 Developing an Ontology

In order to develop an ontology the following steps maybe taken:

- Defining classes in the Ontology,
- Arranging the classes in a taxonomic (subclass-superclass) hierarchy,
- Defining attributes (slots) and describing acceptable values for these slots,
- Filling in the values for slots (attributes) for instances.

We can then create a knowledge base by defining individual instances of these classes filling in specific slot (property) value information and additional attribute restrictions (Noy and McGuinness, 2000).

3.3.2 Why developing an Ontology?

The information in this subsection is based on (Noy and McGuinness, 2000). An Ontology defines a common terminology for researchers who need to share information in a specific domain. It includes machine-interpretable definitions of basic concepts in the domain and relations among them.

There are several motives to develop an Ontology:

- To share common understanding of the structure of information between people or software agents (agreement on syntax).
- To enable reuse of domain knowledge.
- To make domain assumptions explicit.
- To separate domain knowledge from the operational knowledge.
- To analyze domain knowledge.

Sharing common understanding of the structure of information between people or software agents is one of the most common goals in developing Ontologies (Noy and McGuinness, 2000). For example, suppose several different Web sites contain books information or provide books e-commerce services. If these Web sites share and publish the same fundamental Ontology of the terms they all use, then computer agents can extract and aggregate information from these different sites. The agents can use this aggregated information to answer user queries or as input data to other applications (Noy and McGuinness, 2000).

Enabling reuse of domain knowledge was one of the powerful forces behind recent flow in Ontology research. For example, models for many different domains need to represent the notion of time. This representation includes the notions of time intervals, points in time, relative measures of time, and so on. If one group of researchers develops such an Ontology in detail, others can simply reuse it for their domains. In Addition, if we need to build a large Ontology, we can integrate several existing Ontologies describing sections of the large domain (Noy and McGuinness, 2000).

Making explicit domain assumptions underlying an implementation makes it possible to change these assumptions easily if our knowledge about the domain changes. Hard-coding assumptions about the world in programming-language code makes these assumptions not only hard to find and understand but also hard to change, in particular for someone without programming knowledge. In addition, explicit specifications of domain knowledge are useful for new users who must learn the terms in the domain implication.

Separating the domain knowledge from the operational knowledge is another common use of Ontologies. We can describe a job of arranging and designing (constructing) a product from its components according to a required specification and implement a program that makes this design independent from the products and components themselves. We can then develop an Ontology of Car-components and characteristics and apply the algorithm to construct made-to-order Cars. We can also use the same algorithm to construct televisions if we provide Ontology for a television component (Noy and McGuinness, 2000).

Analyzing domain knowledge is possible once a declarative and clear requirement of the terms is available. Formal analysis of terms is extremely valuable when we attempt to reuse existing Ontologies and extend them.

“ Often an Ontology of the domain is not a goal in itself. Developing an Ontology is akin to defining a set of data and their structure for other programs to use. Problem-solving methods, domain-independent applications, and software agents use Ontologies and knowledge bases built from Ontologies as data” (Noy and McGuinness,2000, page 2).

3.4 What is OWL?

OWL is Web Ontology Language. OWL is the new ontology language produced by the W3C Web Ontology Working Group. OWL is thus balanced to be a major formalism for the design and distribution of ontology information, particularly in the Semantic Web (Patel-Schneider, 2004). OWL was designed to provide a common way to process the content of web information instead of displaying it. OWL was designed to be read by computer applications (instead of humans) (www.w3schools.com).

Chapter 4

Mapping Business Rules Categories to Ontology Concepts

4.1 Introduction

Business Rules must be expressive enough to capture business complexity and they should be easy and suitable for business analysts to update and maintain (Demey, et al, 2000).

Ontology can serve as a knowledge base that defines Business Rules in an expressive way.

In this chapter, we are going to make mapping between Ontology concepts and Business Rules categories to see how Ontology expresses each kind of Business Rules. This mapping is depending on our comparison and analysis for the definition of each Business Rule and the definition of each Ontology Concept. Also this mapping will help us in determining what to extract from Ontology.

4.2 Mapping Business Terms to Ontology Concepts (Classes)

As explained earlier Business Rules fall into four categories: Definitions of business terms, Facts relating terms to each other, Constraints or Action Assertions and Derivations. The first category describes how people think and talk about things. They are structured Business Rules, and called Structural Assertions. The terms are of two types: Business Terms that have a specific meaning for the business in some designated Context. For example, Business Terms in rental car context are booking, reservation, and rental request. The other type, Common Terms are considered as parts of basic vocabulary, such as, car, city etc. (Business Rules group, 2000). Terms are represented as Entities in Entity Relational Model ERM , but how are they represented in Ontology?. Ontology Concepts that are represented as Classes can be used to express Business Terms since Classes are interpreted as sets that contain individuals which describes Entity. They are built up of

descriptions that specify the conditions that must be satisfied by an individual for it to be a member of the Class. Also Classes define or express specific Concepts and Concepts identify the way of people think and talk about things.

It is important to note that Ontology main blocks are Classes and Ontology has an important Class called Thing. This Class represents the set containing all individuals.

Because of this all classes in Ontology are considered as subclasses of Thing (Horridge, et al, 2009).

4.3 Mapping Facts relating terms to each other to Ontology Object Properties

Facts relating terms to each other can be represented as relationships that assert an association between two –binary relationships- or more terms –N-ary relationships- or as attributes (Business Rules Group, 2000). In Ontology binary Relationships that link individuals from one Entity (Class) to individuals from another Entity (Class). Ontology Object Properties used to represent Binary Relationships. Also Ontology Object Properties have some characteristics to enrich the meaning of Object Properties such as Functional, Transitive, Reflexive, Irreflexive, Symmetric, Antisymmetric, Inverse Property. We use Functional Object Properties to represent limited Binary Relationship that have a single value. Functional Object Property for a given individual means there can be at most one individual in Entity (Class) that is related to the other Entity (Class) individual via the property. For example the relationship hasBirthMother is a functional property because any individual can only have one birth mother. Also if we want to express the inverse of our Business Rule we can use Ontology Inverse Property. For example the property –we will use the term property instead of relationship- hasChild relationship and its inverse is hasParent. For functional property hasBirthMother, its inverse property in Ontology is also

functional, it is expressed as `isBirthMotherof`. Ontology object properties can be transitive to represent transitivity in some Business Rules. For example has ancestor relationship. Consider ‘Ali hasancestor Ahmad’ and ‘Ahmad hasancestor Mona’ then the relationship ‘Ali hasancestor Mona’ is true. So transitivity means if property relates individual A to individual B and the same property relates individual B to individual C then we can conclude that the same property relates individual A to individual C. Also Ontology Object Property can be Symmetric to express symmetric relationships between Entities (Classes). Symmetric relationship (Property) means when property X relates individual A to individual B then individual B relates to individual A via same Property X. put it in another way symmetric property X means the Property X and its inverse property are the same. For example the relationship hasSibling is Symmetric Property since if ‘Ahmad hasSibling Kamal’ then ‘Kamal hasSibling Ahmad’. For Antisymmetric Property if individual A related via property X to individual B then Property B cannot be related to individual A via the same property X such as isChildOf property. For cyclic relationship, it is represented by Reflexive Properties. A property X is said to be reflexive when the property X relates individual A to itself. For example the relationship Knows ‘Kamal Knows Ahmad’ also ‘Kamal know himself Kamal’. Also properties can be Irreflexive i.e the property X cannot relate individual A to itself as in isMotherof relationship, the person cannot be the mother of itself (Horridge, et al, 2009).

Also some Facts are expressed by compound associations with more than two components that are called N-ary relationships. For example in rental car system “ a customer may request a model of car from a rental branch on a date”, this fact includes four terms: customer, car model, rental branch and date (Business Rules group, 2000). Ontology

represents N-ary relationships in different ways depending on the case. For example we may create an additional attribute that describing a relation instance itself, with links from the subject of the relation to this instance and with links from this instance to all participants that represent additional information about this instance. As explained in Figure 4.1

In this Example we have the N-ary relationship ‘Alia has breast tumor with high probability’ the individual Alia has a property has_diagnosis that has another object (Diagnosis, an instance of the class Diagnosis) as its value:

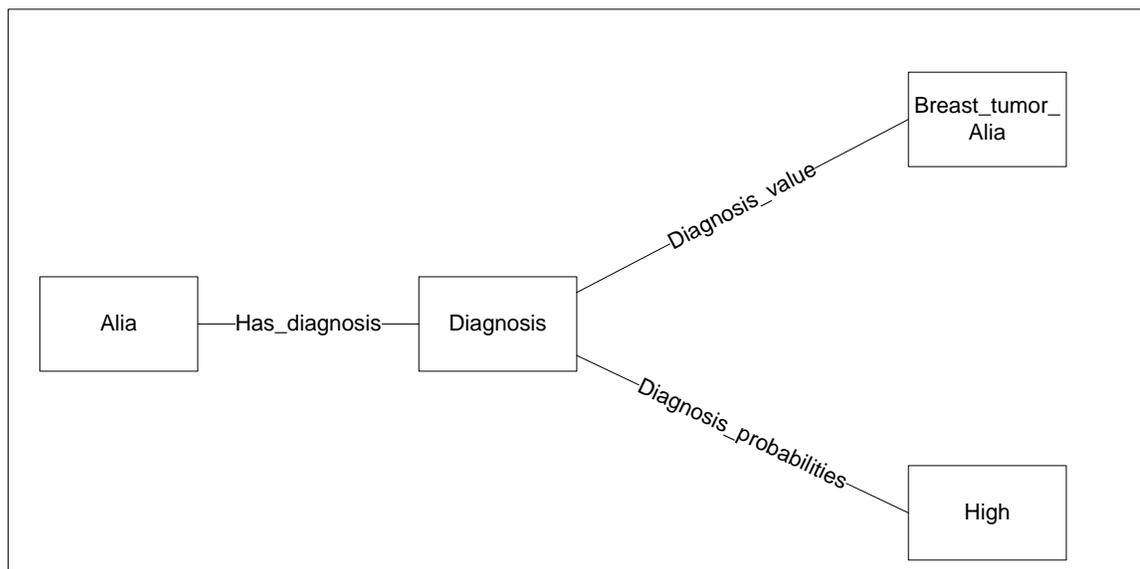


Figure 4.1 First Case of N-ary Relationship

The individual Diagnosis here represents a single object encapsulating both the diagnosis (Breast_Tumor_Alia, a specific instance of Disease) and the probability of the diagnosis (High). It contains all the information held in the original 3 arguments: who is being diagnosed, what the diagnosis is, and what the probability is (www.w3.org).

Another case of N-ary relationship links individuals that play different roles in a structure without any single individual standing out as the subject or the “owner” of the relation,

such as Purchase in Business Rule (*Ali buys a “Lenny the Lion” book from books.example.com for \$15 as a birthday gift*). Here, the relation explicitly has more than one participant, and, in many contexts, none of them can be considered a primary one. In this case, we represent it in Ontology by creating an individual to represent the relation instance with links to all participants (www.w3.org) as its explained in Figure 4.2

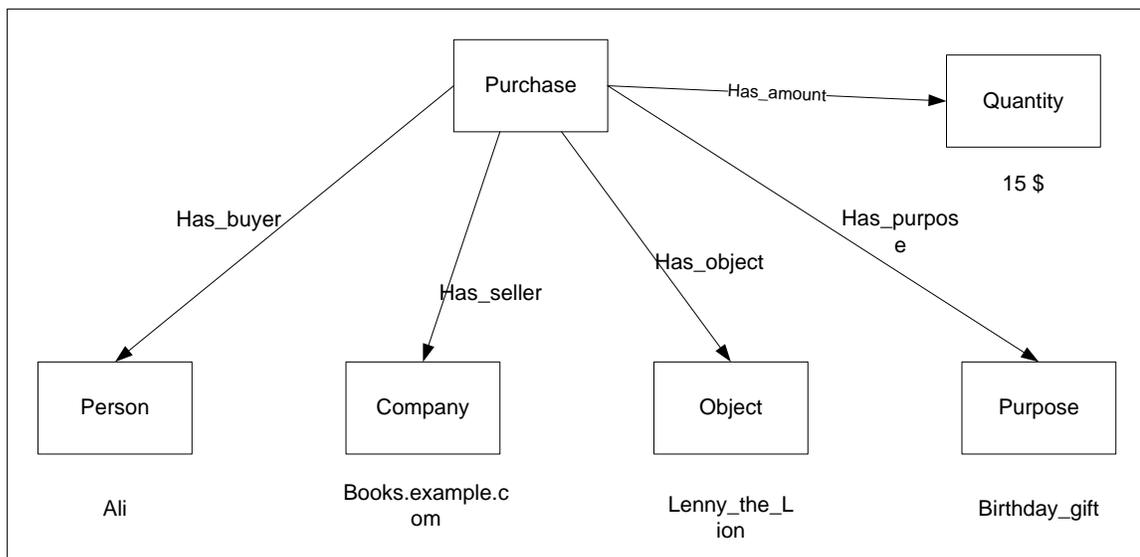


Figure 4.2 Second Case of N-ary Relationship

We broke N-ary relationship to multiple Binary relationships.

Another case of N-ary relationship is represented in Business Rule (Ahmed has temperature which is high but falling) here the relation Temperature-Observation has two properties temperature-value and temperature-trend as explained in Figure 4.3.

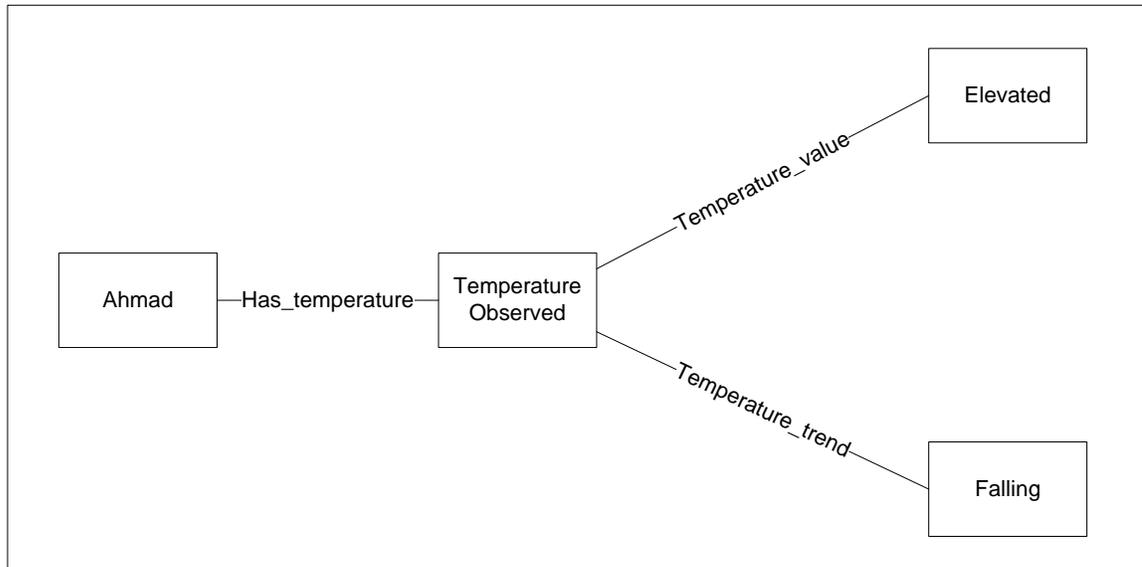


Figure 4.3 Third Case of N-ary Relationship

Another case of N-ary relationship listing relationship when the relationship represented by a sequence of ordered list for example (Jordanian Airlines flight 3177 visits the following airports: Amman, Istanbul, and Milan) in this order. Since the order is important we represent it in Ontology using an ordering relation (`nextSegment`) between instances of the `FlightSegment` class. Each flight segment has a property for the destination of that segment. And we add a special subclass of flight segment, `FinalFlightSegment`, with a maximum cardinality of 0 on the `nextSegment` property, to indicate the end of the sequence (www.w3.org) as explained in Figure 4.4.

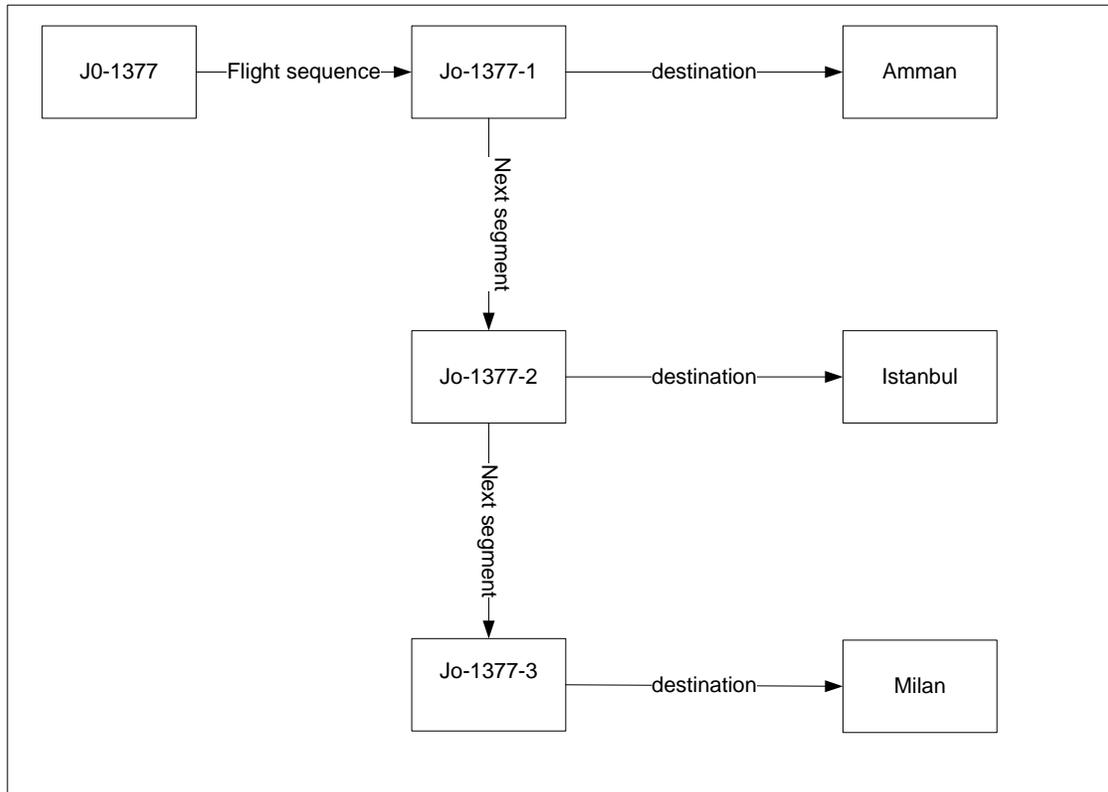


Figure 4.4 Fourth Case of N-ary Relationship

Another kind of Ontology Properties is Datatype Properties (Attributes that are the other type of Business Facts) that link individual to a data value of different types as (integer, string, boolean etc.). For example to represent the Business Fact 'Ahmad's age is 25' in the Ontology we used individual Ahmad hasAge 25.

Also Object Properties have hasValue Property Restriction which "describes the set of individuals that have at least one relationship along a specified property to a specific individual" (Horridge, et al, 2009, page 92) and this is mapped to attributes that describe classes.

4.4 Mapping Constraints to Ontology Restrictions & Necessary and Sufficient Conditions

Constraints on Business Rules are made to prevent some actions from taking place or to prevent a record from being created and they are called Action Assertions (the Business Group, 2000). An Action Assertion can be either a Condition, an Integrity Constraint or An Authorization. First Action Assertion identified a Condition which would be depicted Graphically on Entity Relational Diagram ERM, as constraints that constrain a relationship to applying ‘at least one’ or ‘ no more than’ one occurrence of an Entity. In Ontology these constraints are expressed as Property Restrictions. A property restriction describes a class of individuals based on relationships of which members of the class participate in. Put in another way the restrictions constrain relationship between two individuals from different classes. The Ontology restrictions are of three categories: Quantifier Restrictions, Cardinality Restrictions and hasValue Restrictions. First category Quantifier Restrictions has two types Existential Restrictions and Universal Restrictions. “Existential Restrictions describe classes of individuals that participate in at least one relationship along a specified property to individuals that are members of a specified class” (Horridge, et al, 2009, page38). For example ‘the class of individuals that have at least one hasCertificate relationship to members of Educational Certificate’.

“Universal Restrictions describe classes of individuals that for a given property only have relationships along this property to individuals that are members of a specified class” (Horridge, et al, 2009, page38). For example ‘the class of individuals that only have teachGrades relationships to members of Grades’.

Some Business Rules need to be determined by an exact value. For example in Ontology “we can describe the class of individuals that have at least or at most or exactly a specified

number of relationships with other individuals or data type values. The restrictions that describe these classes are known as Cardinality Restrictions.” (Horridge, et al, 2009, page73). For a given property P we can determine the minimum number of P relationships that an individual must participate in by minimum Cardinality Restriction. Also we can determine the maximum number of P relationships that an individual must participate in by maximum Cardinality Restriction. And we can determine the exact number of P relationships that an individual must participate in by using Exact Cardinality Restriction (Horridge, et al, 2009). An example for Cardinality Restrictions ‘Secondary teacher hasCertificate min 2 Educational Diploma and Ba in Science or Arts’.

Also Datatype Properties at class level has Datatype Restriction that is used to specify restrictions on possible values such as specifying a range of values for a number (Horridge, et al, 2009). We can map conditional constraints to Datatype Restrictions.

Ontology Necessary and Sufficient Conditions on a class that is made when we did properties at superclass level to make Equivalent Class. These conditions are used to make sure that any random individual that satisfies these conditions belongs to that class. In addition to that all individuals of that class must satisfy these conditions. That means these conditions are constraints to determine what are the constructed individuals in each class. So it is mapped to Action Assertion Integrity Constraint. Since An Integrity Constraint is an Assertion that must always be true, it is considered to have immediate enforcement power because it prohibits any actions which would result in a false value (Business Rules Group, 2000).

For conditional constraint “if Statement” it can be represented in the OWL-based Web Service Ontology OWL-S Language which is extension of the Web Ontology Language OWL. “OWL-S is considered as supporting tools and agent technology to enable automation of services on the Semantic Web” (www.daml.org).

4.5 Mapping Derivations to Ontology Closure Axioms

Terms are base facts that are given in the world and stored in the system. A derived fact is created by inference or a mathematical calculation from Terms, Facts, other Derivations, or Action Assertion (Business Rules Group, 2000). For example when we calculate the salary of an employee this is a derived fact from base salary and allowances plus deductions, also when we infer that the teacher is qualified this is also a derived fact built on qualifications that the teacher had. These derived facts or Derivations can be mapped to Ontology Axioms.

The closure Axiom is defined as “A closure Axiom on a property consists of a universal restriction that acts along the property to say that it can only be filled by the specified fillers” (Horridge, et al, 2009, page 64). To put it in another way when we define a closure axiom in Ontology we restrict the value for the relationship (object property) to belong to specific domains. So we always can infer and derive that this object property has these specific domains. For example, when we make a closure axiom on a SecondaryTeacher that TeachesLevel only Level, then we can infer and derive the fact that SecondaryTeacher can teach this certain level only.

Also when we make Covering Axiom in Ontology we can derive new information. Since the definition of Covering Axiom as follows, having three classes A,B, and C with B and C being subclasses of A. Which means any member of B or C is also a member of A. Making

class A covered by class B and C makes us derive that any member of A is also a member of classes B union C. Also class A would be a super class for classes (B U C). A covering Axiom marked itself as a class that is the union of the classes being covered.

In addition to that, the characteristics of Ontology Object Properties such as symmetric, antisymmetric, reflexive, transitive, functional, inverse, irreflexive which are explained in section 4.2.2 can help in deriving new information from it as their definitions explained.

4.6 Mapping Summary Table

In the following the summary table for our mapping between Business Rules Categories and Ontology Concepts:

Table 4.1 Mapping Business Rules Concepts Into Ontology Concepts

Business Rules concept	Ontology Concept
Business Rules Terms and Sub Terms	Ontology Classes and Subclasses
Business Rules Facts	Ontology Object Properties
Business Attributes	Ontology Data Properties
Conditional Constraints	Property Restrictions & Cardinality Restrictions
Action Assertion Integrity Constraints	Necessary & Sufficient Conditions
If Statement Constraints	If Statement Constraints
Derivations	Ontology Axioms & Object Property characteristics

4.7 Conclusion

In this chapter, we did the mapping between Business Rules categories and Ontology concepts depending on our analysis and comparison between both sides Business Rules concepts and Ontology concepts. We found for each concept in Business Rules categories a matched concept in Ontology. This mapping helps in determining what concepts do we have to extract from Ontology to get Business Rules. In (Hodrob and Jarrar, 2010) they represented a mapping between Object Role Model ORM which is a conceptual modeling language used in Ontology engineering. It contains group of constraints that can represent an Ontology using rich graphical notation. And OWL 2 DL Web Ontology Language to utilize benefits of both ORM and OWL 2 DL.

Chapter 5

Case Studies

5.1 Introduction

In this chapter, we explain how different types of Business Rules are represented in an Ontology using Protégé tool. We illustrate different case studies to help us in concluding our Business Rules extraction model by determining relationships between Ontology concepts. First case study is our own and we created it from the teachers domain. This Ontology domain is built to determine if the teacher is qualified or not qualified. We do not focus on results but on the way Business Rules are represented to explain, how easy and clear it is to represent Business Rules in Ontology and to implement the mapping between Business Rules categories and Ontology Concepts. And other case studies are already implemented, we will explain how Business Rules represented in them to help us in generalizing our extraction model.

5.2 Qualified Teacher Ontology

We will have some case studies to explain the idea and purpose of this thesis. In our own case study we chose a specific domain to build Ontology which is called Qualified Teacher. Ontology of Qualified Teacher will determine if a specific teacher is qualified or not depending on “Teacher Education Strategy in Palestine” so each teacher considered as a member of our Ontology classes will be a Qualified Teacher because that means this teacher satisfies the necessary conditions to be qualified and any teacher does not fit to be a member of our Ontology classes will not be qualified.

5.2.1 Qualified Teacher Main Classes

Teacher Ontology contains six main classes as described in Figure 5.1 .

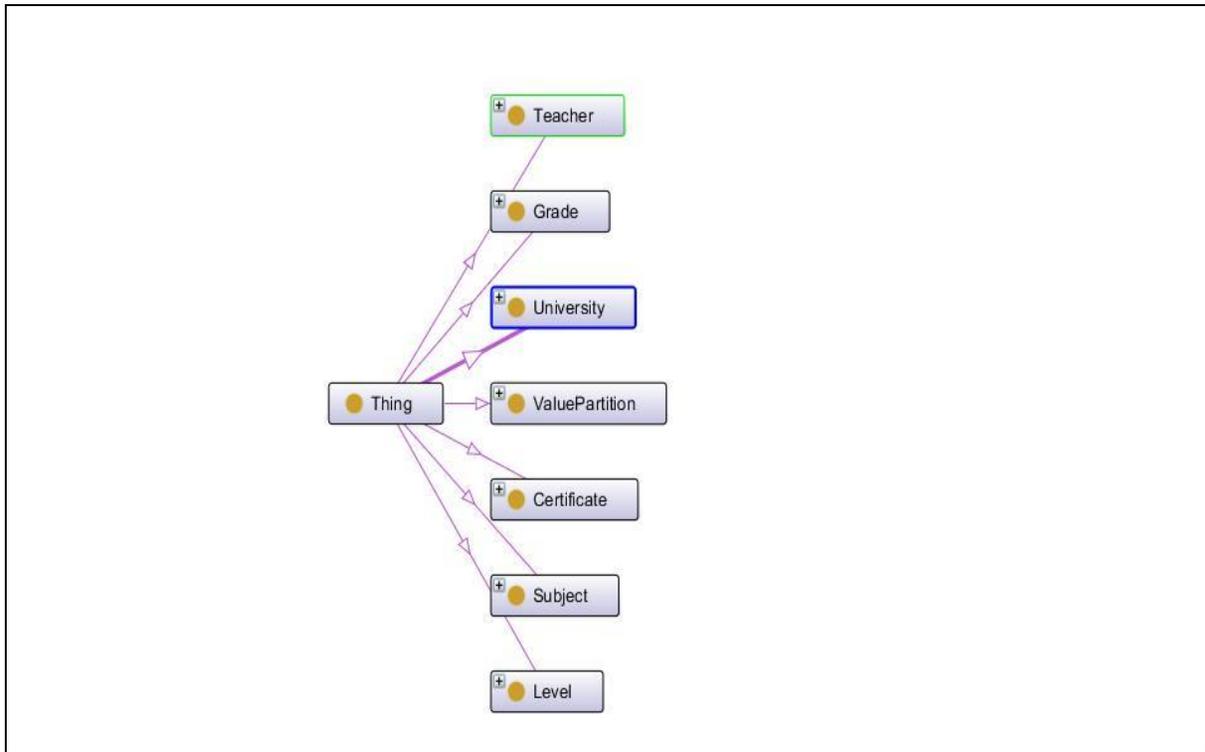


Figure 5.1 Teacher Ontology Main Classes

As explained in Figure 5.1 our Ontology has six main classes which are “*Teacher*” class which holds the information about the teacher. “*Grade*” class that explains different grades that teacher teaches. “*University*” class that holds information about different universities that teacher graduated from. “*Certificate*” class one of important classes for our Ontology since it holds information about certificates that teachers had and is considered a main factor to determine if a teacher is qualified or not. “*Subject*” class which holds information about different subjects that teacher teaches. And finally “*Level*” class that holds information about different levels that teacher teaches. And these classes are made disjoint from each other which means they don’t have multiple inheritance i.e. a member of one class cannot be a member of another class.

One of main classes is: “*Teacher*” class, which has five subclasses to express different categories of teachers as appeared in Figure 5.2.

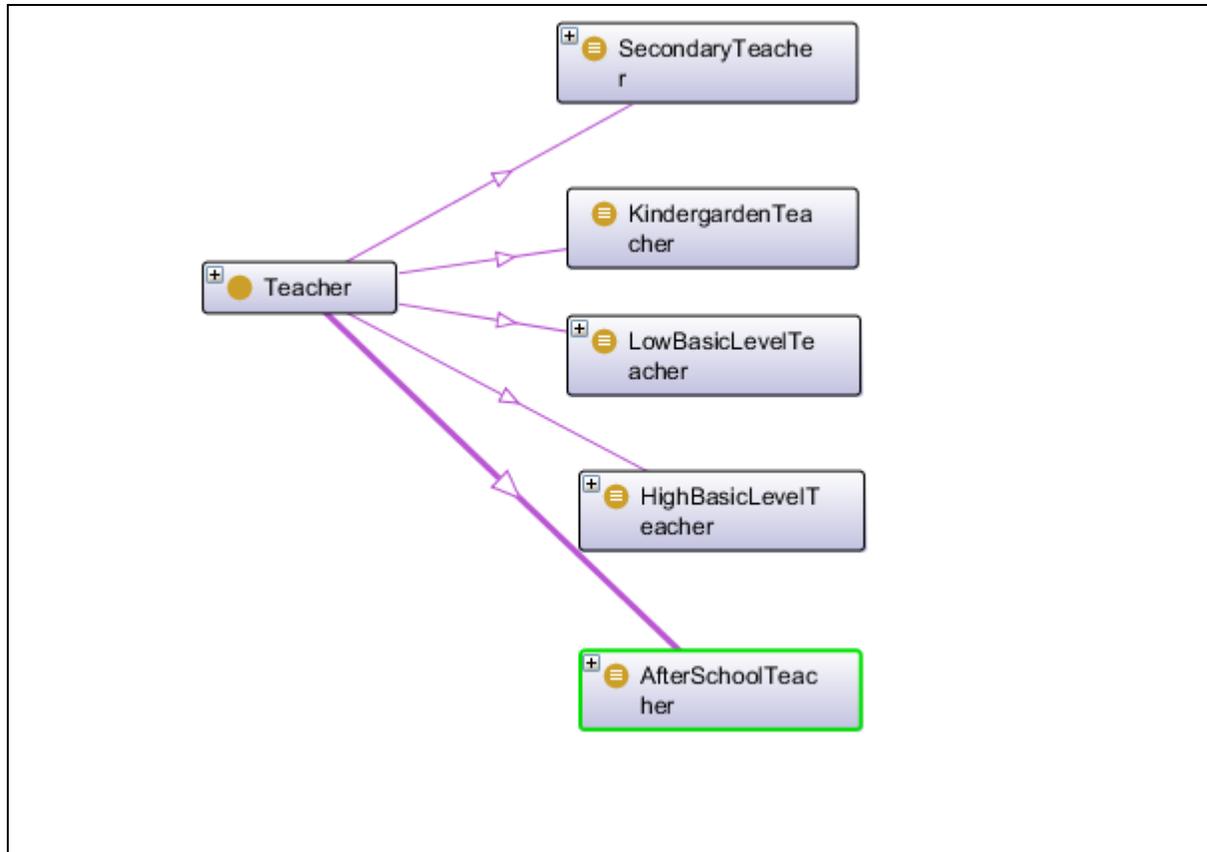


Figure 5.2 Teacher Categories

These subclasses are determined upon qualification categories for a teacher based on “Teacher Education Strategy in Palestine”. These subclasses are “*SecondaryTeacher*” that teaches secondary level, i.e. grades 11 and 12 and must have BA degree in Science or Arts specializing in a subject taught at schools and a Diploma in Education for secondary level specializing in teaching a specific subject or (MA degree in Education). Another subclass “*KindergartenTeacher*” who teaches pre-school level and must have a BA degree in Education with a major in Pre-school Education. Notice that as interim stage a Two-year Diploma would be accepted as a certificate for Kindergarten teacher. Another subclass

“*LowBasicLevelTeacher*” which must have a BA in Education with a major in teaching Lower Basic Level. Also “*LowBasicLevelTeacher*” is supposed to teach all subjects for grades (1-4). Another subclass, “*HighBasicLevelTeacher*”, that teaches grades from 5 to 10 and is classified for two programs. “*HighBasicLevelTeacherProgramA*” must have a BA in Education with a major in upper basic level and teaching a specific subject such as (Arabic, Math, English, Physic etc...). “*HighBasicLevelTeacherProgramB*” must have a BA degree in Sciences or Arts or other (such as engineering for TVE schools), in a subject taught at schools and a Diploma in Education for upper basic level: teaching a specific subject as previous (or an MA).

The last subclass is “*AfterSchoolTeacher*” as explained in Figure 5.3. This class defines the teachers who are responsible for training educators and trainers for tertiary level and these teachers must have at least Diploma in Teacher Education Specializing in Higher or Adult education. “*AfterSchoolTeacher*” program is to qualify university teachers and adult educators. Next to Figure 5.3 is the OWL code for declaration of this class.

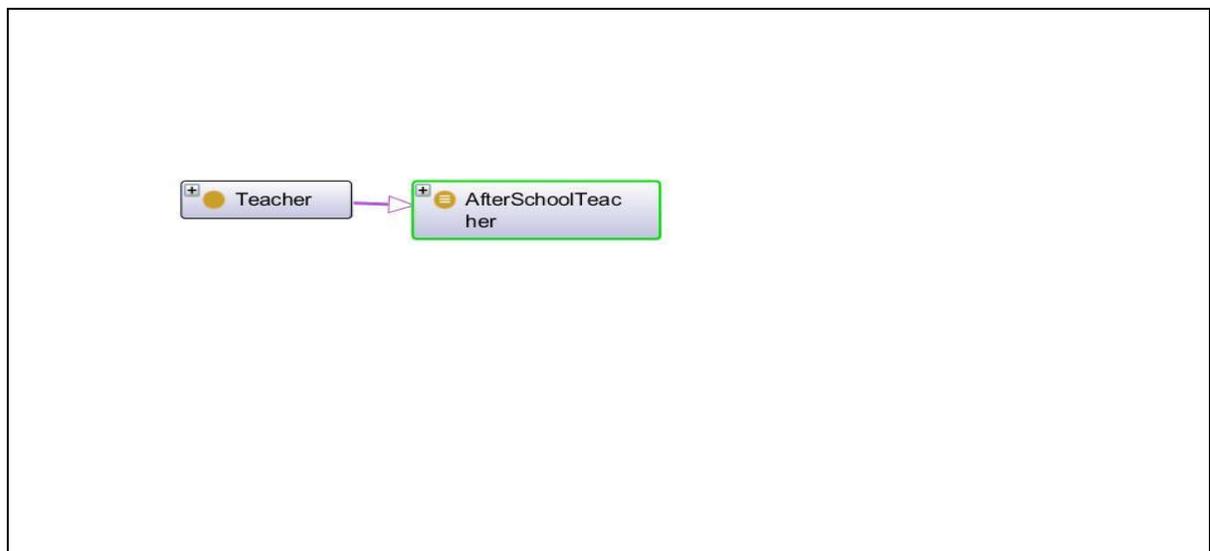


Figure 5.3 AfterSchoolTeacher Category (Class)

```

<Declaration>

  <Class IRI="#AfterSchoolTeacher"/>

</Declaration>

<EquivalentClasses>

  <Class IRI="#AfterSchoolTeacher"/>

  <ObjectIntersectionOf>

    <Class IRI="#Teacher"/>

    <ObjectSomeValuesFrom>

      <ObjectProperty IRI="#hascertificate"/>

      <Class IRI="#DiplomainTeacherEducation"/>

    </ObjectSomeValuesFrom>

  </ObjectIntersectionOf>

</EquivalentClasses>

```

OWL Rendering XML Code for AfterSchoolTeacher Class

And we made it EquivalentClass to let any random member that satisfies the conditions that is a teacher and has Diploma in Teacher Education Specializing in Higher or Adult Education be a member of “*AfterSchoolTeacher*” class. i.e. change necessary conditions to necessary and sufficient conditions as explained in chapter 4. In Addition to that each

member of the “*AfterSchoolTeacher*” satisfies these conditions. And these Business Rules are determined as explained in the following Figure 5.4.

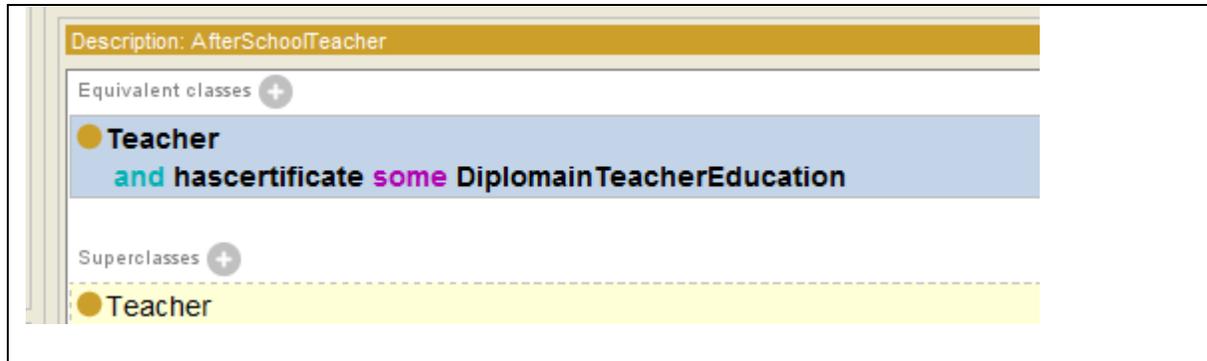


Figure 5.4 Business Rules for AfterSchoolTeacher Class

So “*AfterSchoolTeacher*” is a Teacher and has certificate Diploma in Teacher Education.

And here are the Business Rules that inherited from the super class Teacher as described in Figure 5.5

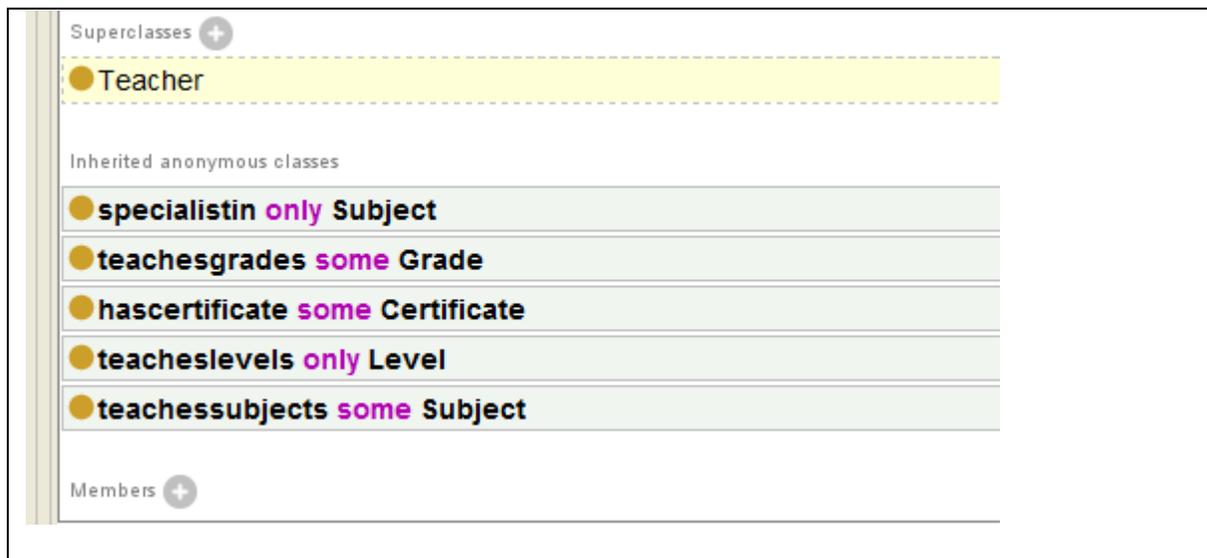


Figure 5.5 Inherited Object Properties for Subclasses of Teacher Class

5.2.2 Object Properties and Quantifier Restrictions

“*AfterSchoolTeacher*” and other subclasses of class *Teacher* such as (*SecondaryTeacher*, *HighBasicLevelTeacher*, *LowbasicLevelTeacher* and *KindergardenTeacher*) inherit some relationships (Object Properties) from *Teacher* Super class such as:”specialist in only Subject” and this means the teacher is specialized in one subject only that is what “only” means as we explained in chapter4. Only is “Quantifier Universal Restriction describes classes of individuals that for a given property only have relationships along this property to individuals that are members of specified class” (Horridge, et al, 2009). As here we restrict the teacher to have specialist relationship with only one subject.

Another Object Property (relationship) “teachesgrades some Grade” which means teacher can teach one or more grades that is why we used “some” which means the relationship existence i.e. the teacher can teach at least one grade. Since some is Quantifier Existential Restriction and “Existential Restrictions describe classes of individuals that participate in at least one relationship along a specified property to individuals that are members of a specified class” (Horridge, et al, 2009).

Another Object Property “hascertificate some Certificate” which means the teacher can have one or more certificates.

Another Object Property “teacheslevels only level” this means that the teacher can teach only one study level only.

And finally Object Property “teachessubjects some subject” means that the teacher can teach one or more subjects. Object Properties in Ontology are considered the same relationships link between Entities.

Figure 5.6 shows Object Properties for the superclass “*Teacher*” to explain similarity between them and inherited Object Properties that subclasses of “*Teacher*” class have.

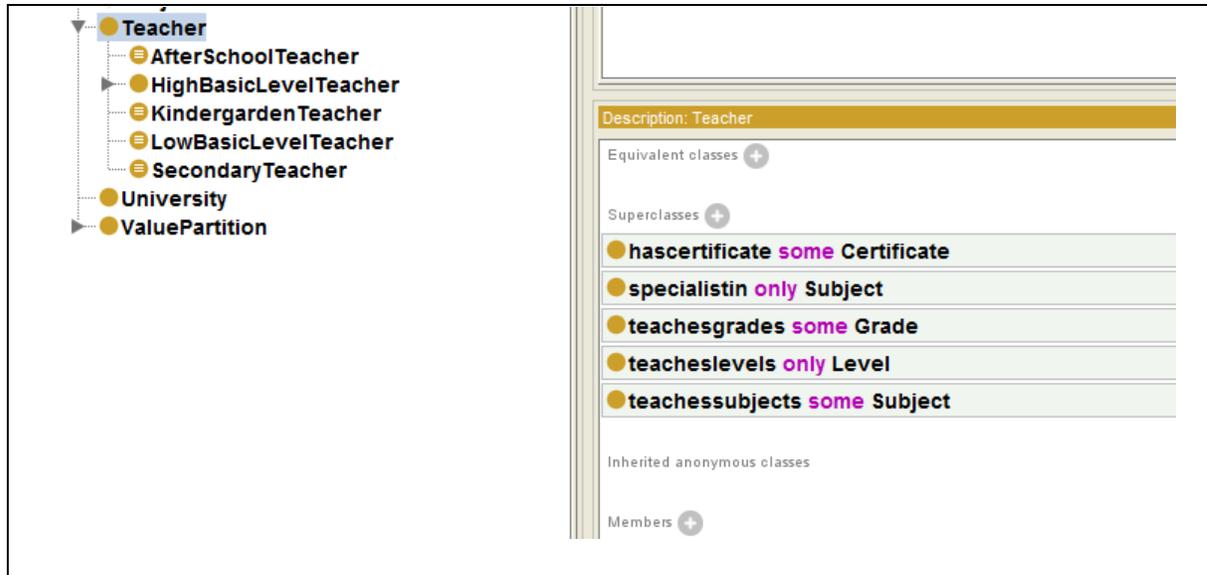


Figure 5.6 Object Properties For Teacher Class

All Object Properties (Relationships) explained so far are binary relationships, that connect two classes. For N-ary relationships we gave examples in detail in chapter 4.

5.2.3 Cardinality Restrictions & Necessary and Sufficient Conditions

In Figure 5.7 we explain the Cardinality Restriction (exactly) explained in chapter 4. We define “*KindergardenTeacher*” class to be restricted to teach exactly one kindergarten grade. Since each kindergarten teacher must be responsible for one grade, in addition to basic qualifications required for kindergarten teacher. Since kindergarten teacher must have BA in Education with major in pre-school Education or temporary a two year diploma. Also these Business Rules are made as equivalent classes to show that these are necessary and sufficient conditions for “*KindergardenTeacher*” class to be a member of the class.

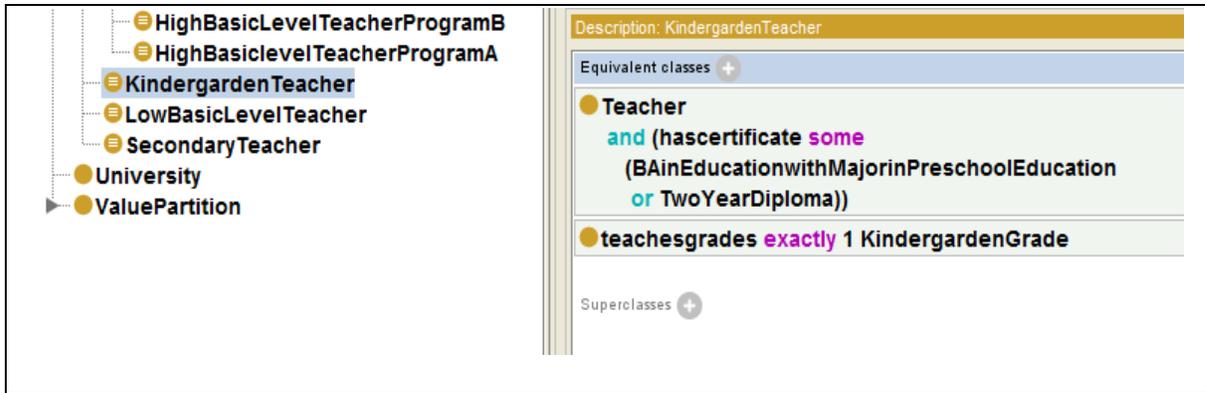


Figure 5.7 Necessary and Sufficient Conditions to Define KindergartenTeacher Class

Another example of necessary and sufficient conditions in Figure 5.8. This figure shows the necessary and sufficient conditions that should be satisfied to be a member of class “*LowBasicLevelTeacher*”. Since it is necessary for each member of the class “*LowBasicLevelTeacher*” to teach grades from grade 1 to grade 4 and to have BA degree in Education with a major in teaching low basic level. And any random member satisfy these conditions must be a member of the class “*LowBasicLevelTeacher*”, that is why these conditions (Business Rules) were done in Equivalent Class part to complete the definition of the class.

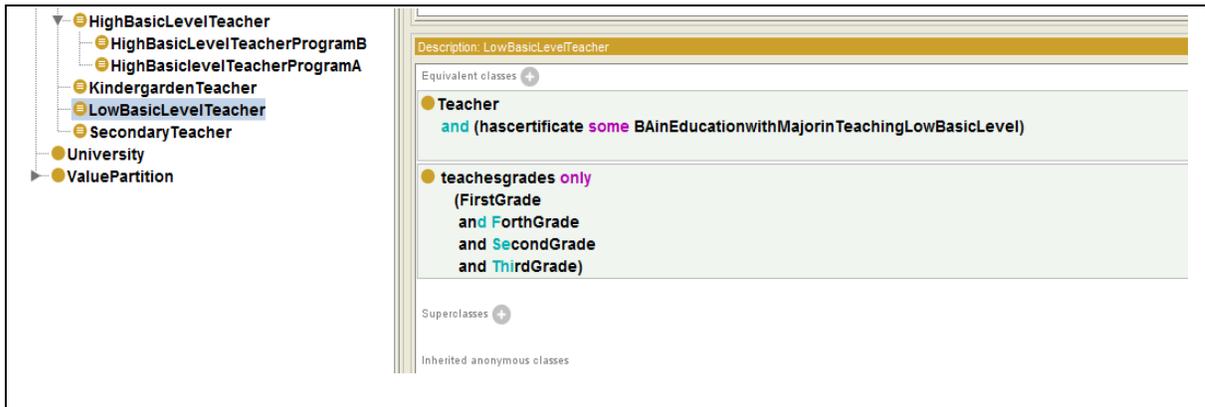


Figure 5.8 Necessary and Sufficient Conditions for LowBasicLevelTeacher Class

The same thing is done for all other subclasses of Teacher except for the “*SecondaryTeacher*”. We change the rules to explain the difference between defined classes that have necessary and sufficient conditions and primitive classes that have only necessary conditions. Necessary conditions mean each member of the class must satisfy these conditions, but didn’t imply that any random member that satisfy them would be a member of the class. Figure 5.9 shows necessary conditions for “*SecondaryTeacher*”. Such as teachesgrades only (EleventhGrade and TwelveGrade) These conditions are done in superclass part. And Figure 5.9 shows Necessary and Sufficient Conditions in Equivalent class part. Such as (hascertificate some DiplomainEducationSecondaryLevel).

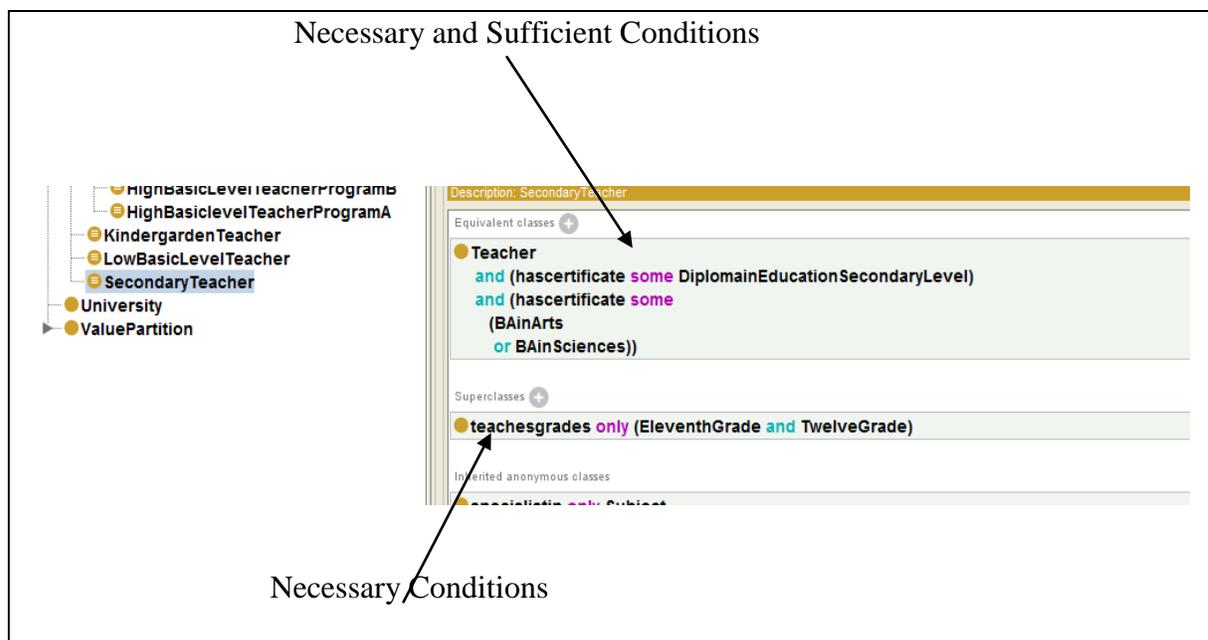


Figure 5.9 Difference between Necessary Conditions and Necessary & Sufficient Conditions

Another Teacher category is “*HighBasicLevelTeacher*” which has two Subclasses as described in Figure 5.10

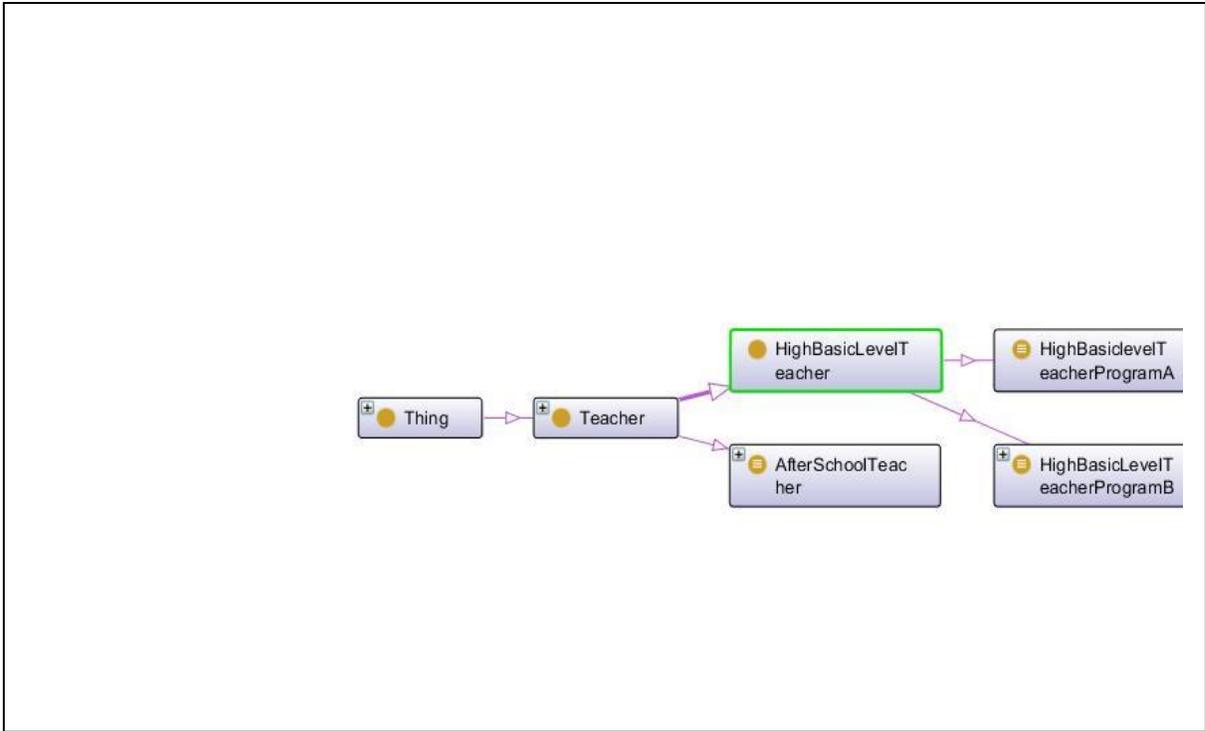


Figure 5.10 HighBasicLevelTeacher Subclasses

Here are some Business Rules for subclass “HighBasicLevelTeacherProgramA”. It is equivalent to class “HighBasicLevelTeacher” which means it inherits all the class properties and rules and there are specific Business Rules that related to “HighBasicLevelTeacherProgramA”. Such as any member belongs to this class must has certificate in “BA in Education with Major in Teaching High Basic Level” or has certificate in “Teaching a subject” as its described in Figure 5.11

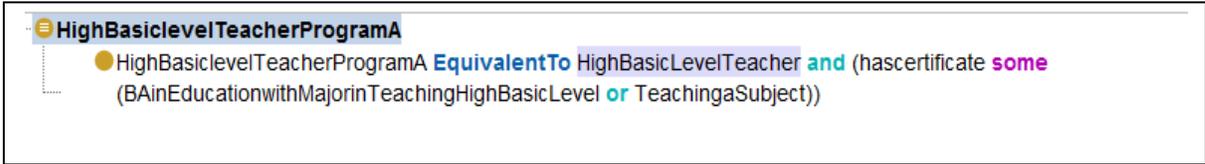


Figure 5.11 Business Rules for HighBasicLevelTeacherProgramA Subclass

And the Business Rules for the subclass “HighBasicLevelTeacherProgramB” are described in Figure 5.12

Description: HighBasicLevelTeacherProgramB

Equivalent classes +

- **HighBasicLevelTeacher**
 - and hascertificate some (BAinArts or BAinSciences)
 - and hascertificate some DiplomainEducationHighBasicLevel

Superclasses +

- HighBasicLevelTeacher

Inherited anonymous classes

- **specialistin only Subject**
- **teachesgrades some Grade**
- **hascertificate some Certificate**
- **teacheslevels only Level**
- **teachessubjects some Subject**

Figure 5.12 Business Rules for HighBasicLevelTeacherProgramB class

Figure 5.13 describes Business Rules for the “*KindergardenTeacher*” class as follows.

KindergardenTeacher

- HighBasicLevelTeacher
- KindergardenTeacher
- LowBasicLevelTeacher
- SecondaryTeacher
- University
- ValuePartition

Description: KindergardenTeacher

Equivalent classes +

- **Teacher**
 - and (hascertificate some (BAinEducationwithMajorinPreschoolEducation or TwoYearDiploma))
 - teachesgrades exactly 1 KindergardenGrade

Superclasses +

Inherited anonymous classes

- **specialistin only Subject**
- **teachesgrades some Grade**
- **hascertificate some Certificate**
- **teacheslevels only Level**

Figure 5.13 Business Rules for KindergardenTeacher class

Figure 5.13 shows the “*KindergartenTeacher*” must have at least a Two year Diploma or BA degree in Education with major Pre-school Education, and it shows that the Kindergarten teacher teaches exactly one kindergarten grade. Also Figure 5.13 shows the inherited Business Rules that inherited from the super class Teacher.

The Business Rules for “*LowBasicLevelTeacher*” are described in Figure 5.14

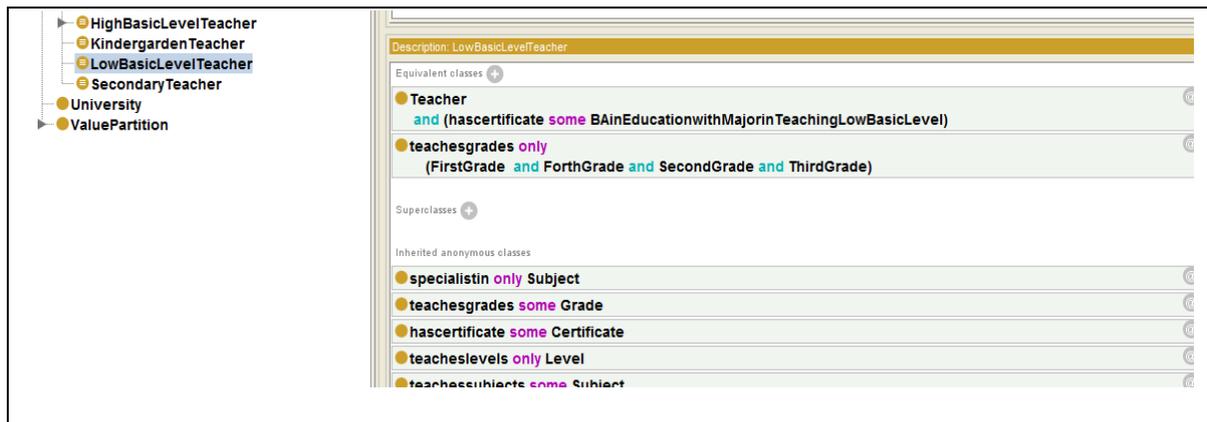


Figure 5.14 Business Rules for LowBasicLevelTeacher Class

And the Business Rules for last category of teachers which is “*SecondaryTeacher*” is described in Figure 5.15

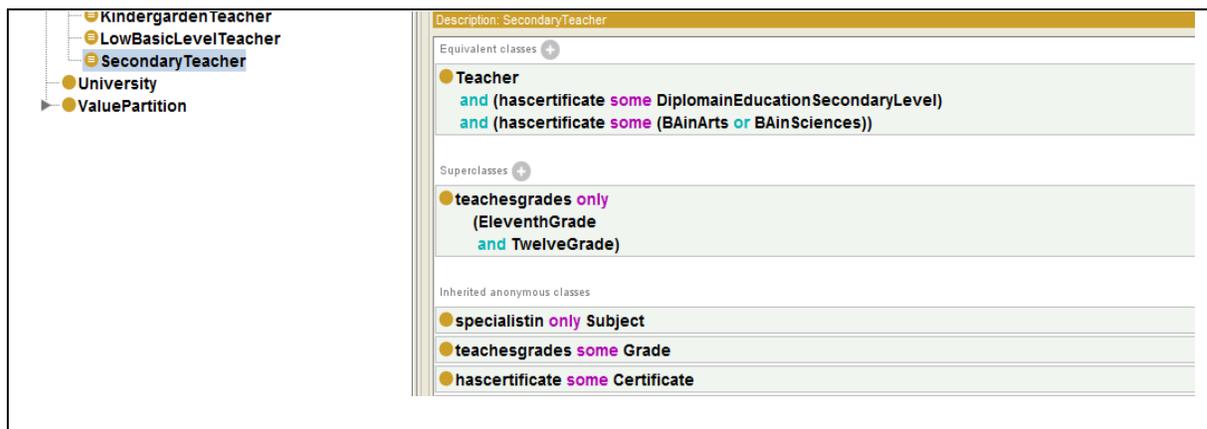


Figure 5.15 Business Rules for SecondaryTeacher Class

In here, the “*SecondaryTeacher*” members must satisfy two conditions first having certificate in Diploma Education in Secondary Level, and second having BA. Degree in Sciences Or Arts. Also “*SecondaryTeacher*” teaches only two grades: eleventh grade and twelve grade

5.2.4 Value Partition

Also Business Rules could be limitation for some values as “*PromotionValue*” class describes some values for subclass Expert teacher regarding years of experience. Expert teacher must have at least five years of experience as it is described in Figure 5.16. and Novice Teacher has years of experience less than five years. This can be done by creating value partitions. Value partitions can be created to refine classes descriptions. In our case study we wanted to describe the experience period for teacher. We did this by creating a “*ValuePartition*” class as a sub class of Thing. Then we created a “*PromotionValue*” as a subclass of “*ValuePartition*” to restrict the range of possible values for teachers regarding their experience. Teachers were classified for two categories regarding experience years. Expert teachers that have more than or equal to five years of experience and Novice teachers that have less than five years of experience, we did this using Cardinality Restrictions as its shown in Figure 5.16 and Figure5.17

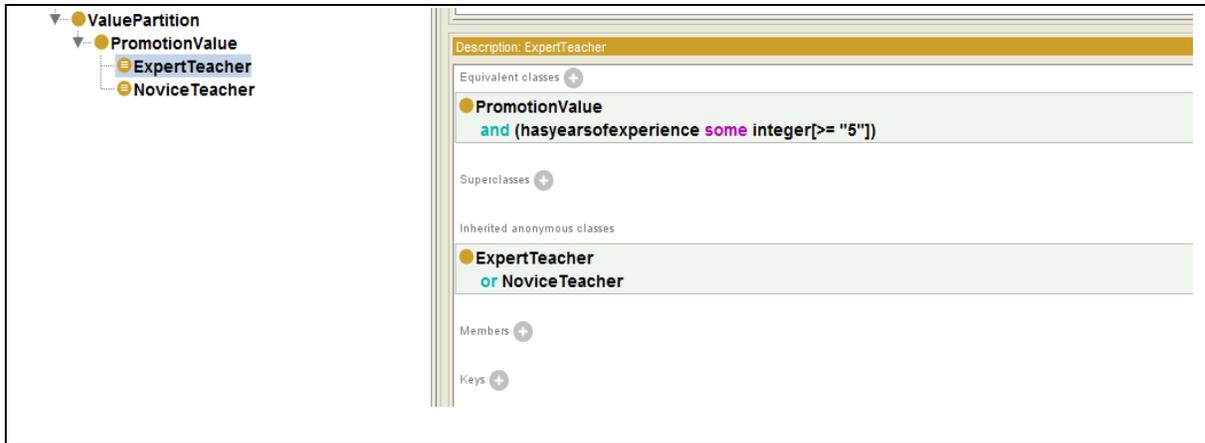


Figure 5.16 Business Rules for ExpertTeacher class

And we determine the value for any new teacher which we called “*NoviceTeacher*” to be less than five years of experience as Figure 5.17 shows.

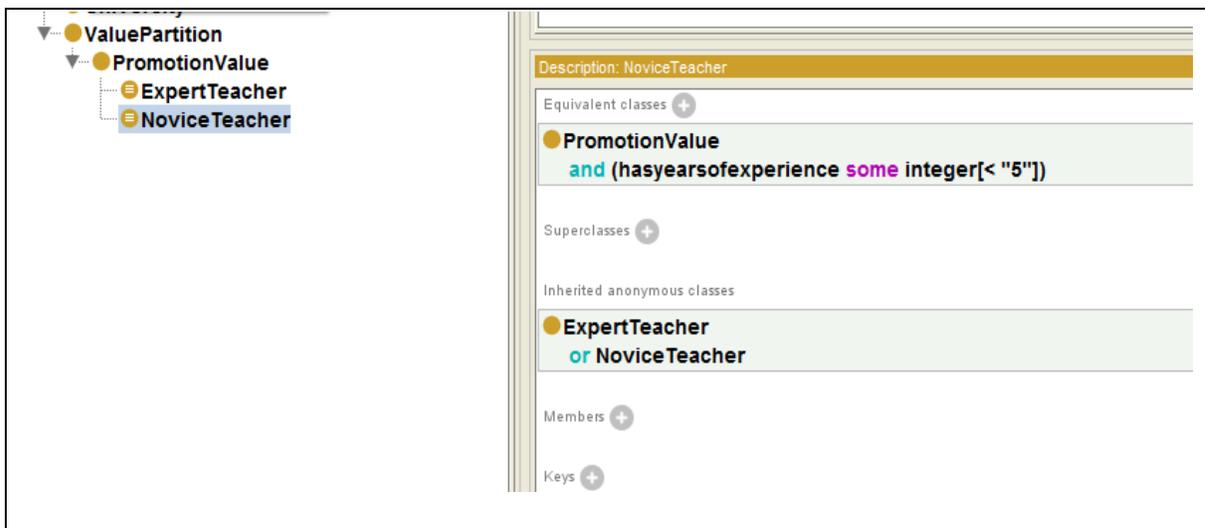


Figure 5.17 Business Rules for NoviceTeacher class

Other classes in Qualified Teacher Ontology are described in the following figures. For example, “*Certificate*” class have certain subclasses that help in determining the education degree for each qualified teacher regarding his level as described in Figure 5.18

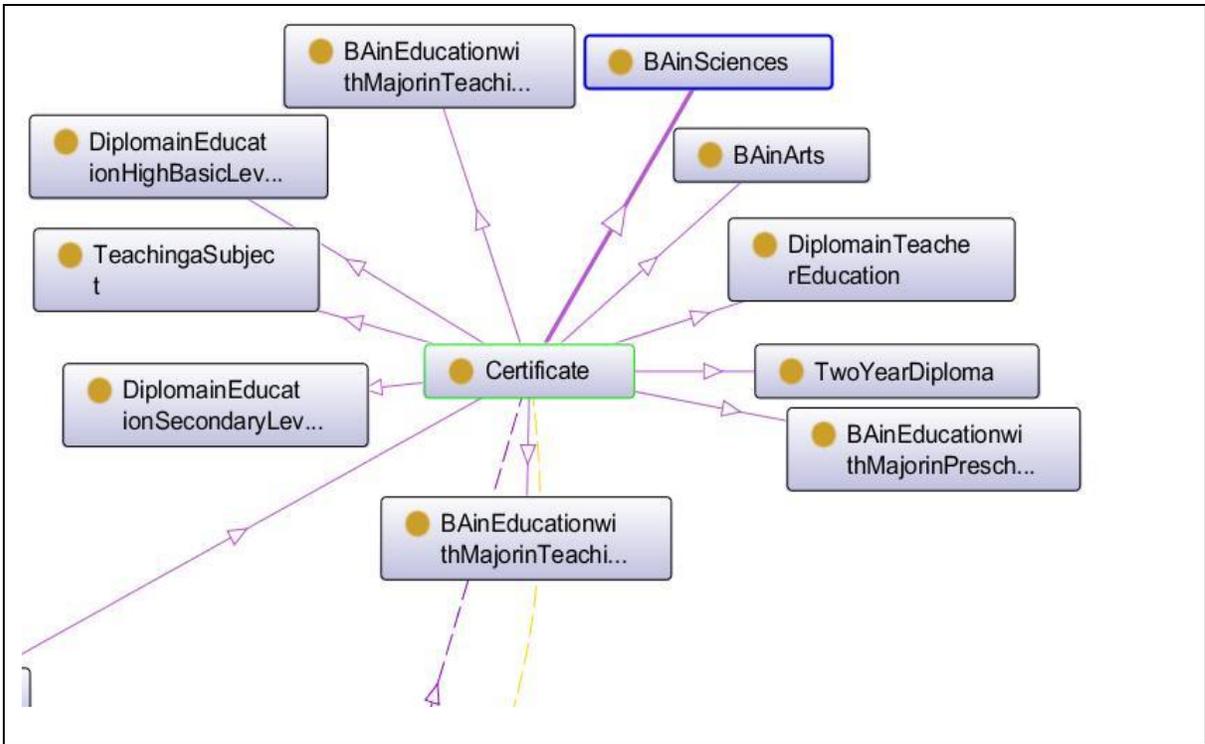


Figure 5.18 Subclasses for Certificate class

“Grade” class describes the classes that can be taught by the teacher. It contains twelve subclasses in addition to kindergarten grade as described in Figure 5.19

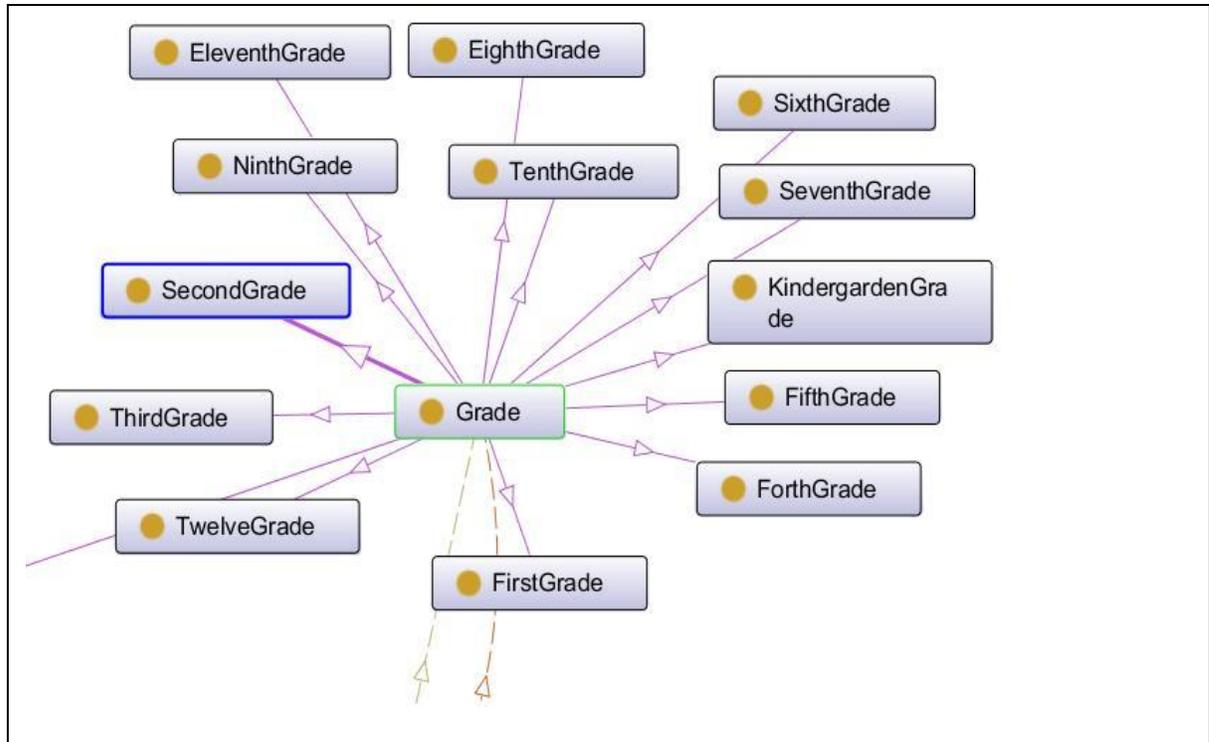


Figure 5.19 Subclasses for Grade Class

“*Level*” class describes main levels that the teacher can teach regarding Teacher Education Strategy in Palestine as Figure 5.20 shows. “*Level1-4*” means teacher teaches grades from grade one to grade four, “*Level5-10*” the teacher teaches from fifth grade to tenth grade, “*Level11-12*” the teacher teaches from eleventh grade to twelve grade, “*Level1-10*” the teacher teaches from first grade to tenth grade, “*Level5-12*” the teacher teaches from fifth grade to twelve grade, “*Level8-12*” the teacher teaches from eighth grade to twelve grade, *Level1-12* the teacher teaches from first grade to twelve grade and “*KindergardenLevel*” the teacher teaches pre-school grades.

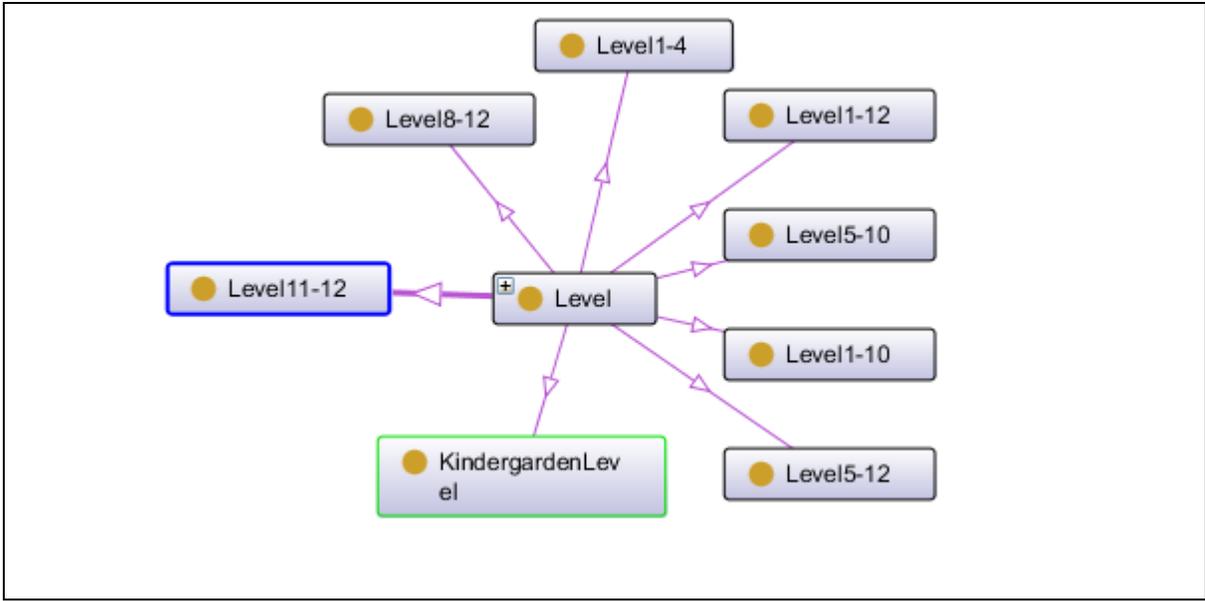


Figure 5.20 Subclasses for Level Class

“Subject” class describes the subjects taught at schools as Figure 5.21 shows.

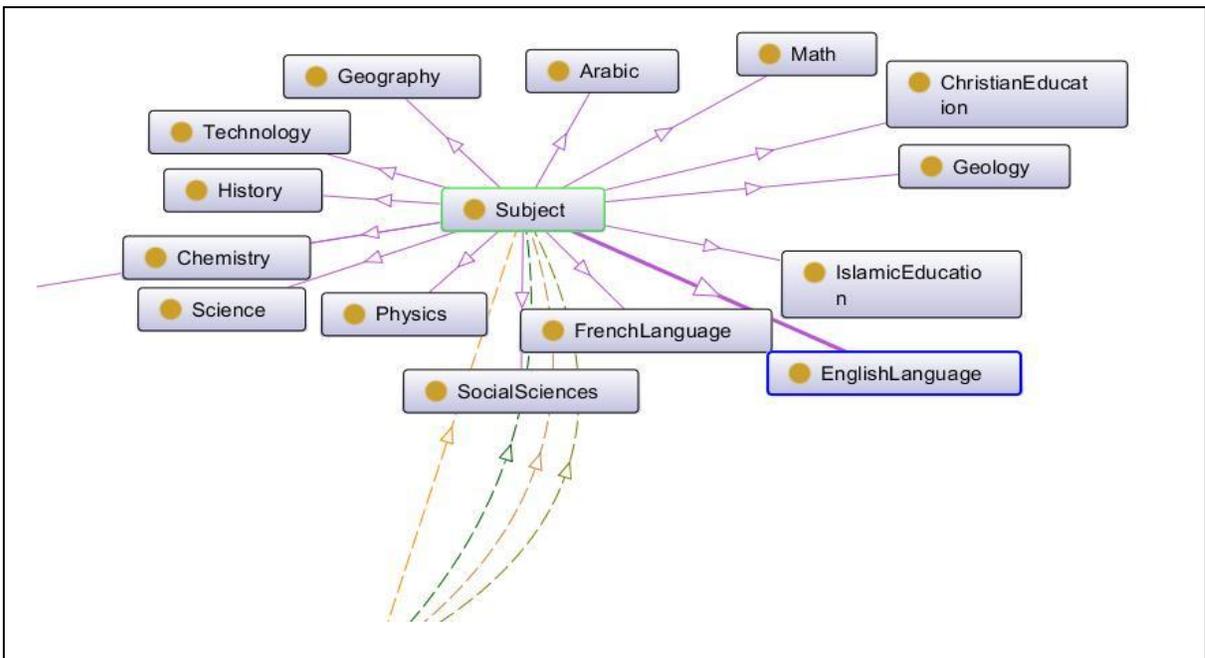


Figure 5.21 Subclasses for Subject Class

5.2.5 Data Properties

To know how Ontology represents attributes of classes see Figure 5.22

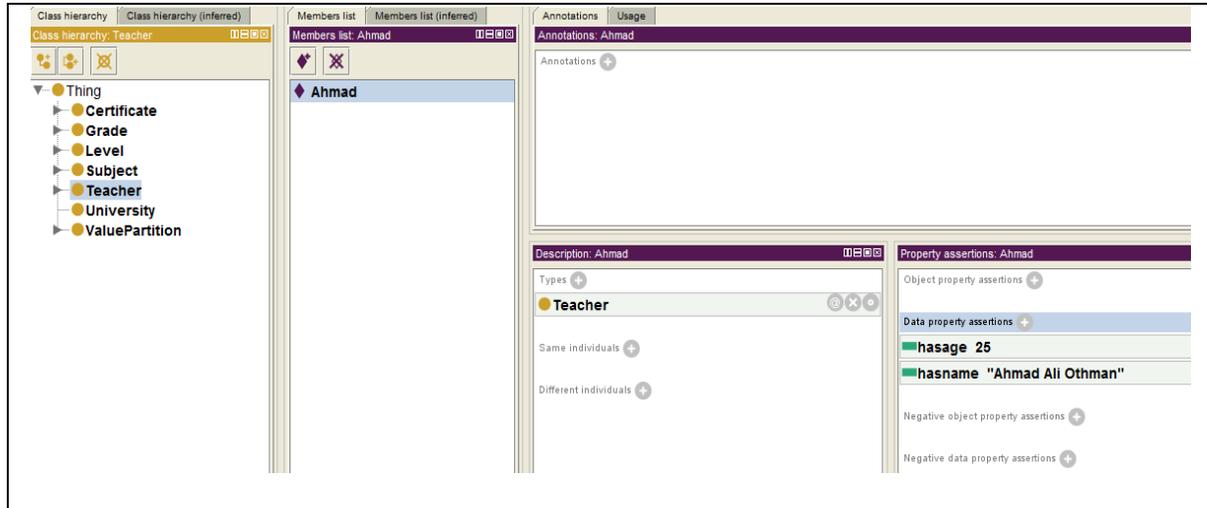


Figure 5.22 Attributes at individuals level

As seen in Figure 5.22 the Attributes or (Data Properties) are represented as relationships at the level of individuals (members) of class. For example teacher (Ahmad hasname “Ahmad ali Othman”) and his age is 25 as its clear in the Data Property (has age 25).

To generalize this Data Property we make it at class level as shown in Figure 5.23. For example the Data Property (hasname some string) and the Data Property (hasage some integer).

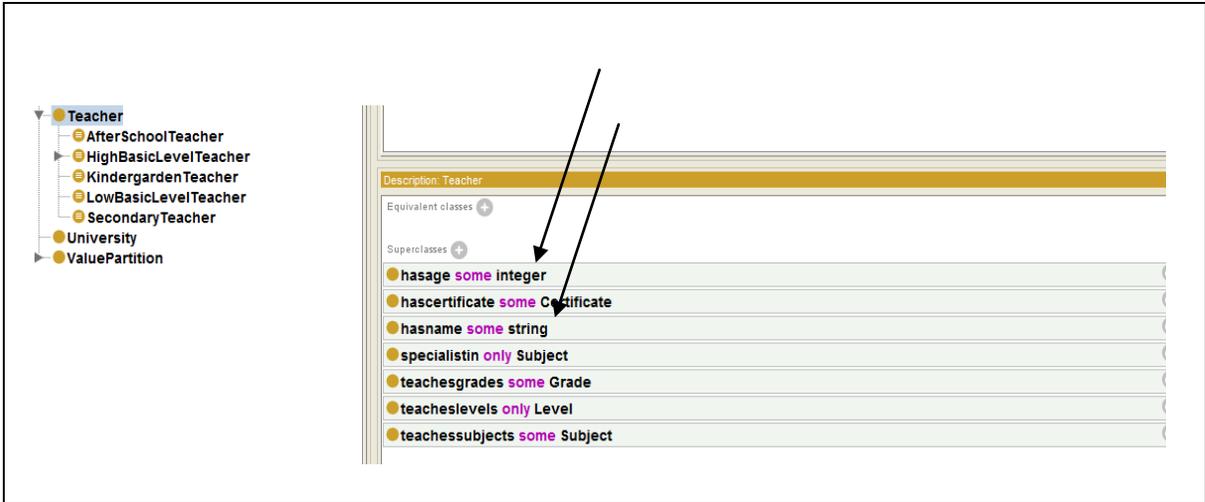


Figure 5.23 Attributes at class level

Also we can use Data Properties Restrictions to define ranges for classes. For example we have another datatype or Data Property called `hasyearsofexperience`. We restrict Expert teacher to have years of experience more than five years and we restrict Novice teacher to have years of experience less than five years. As shown in Figure 5.24 and Figure 5.25.

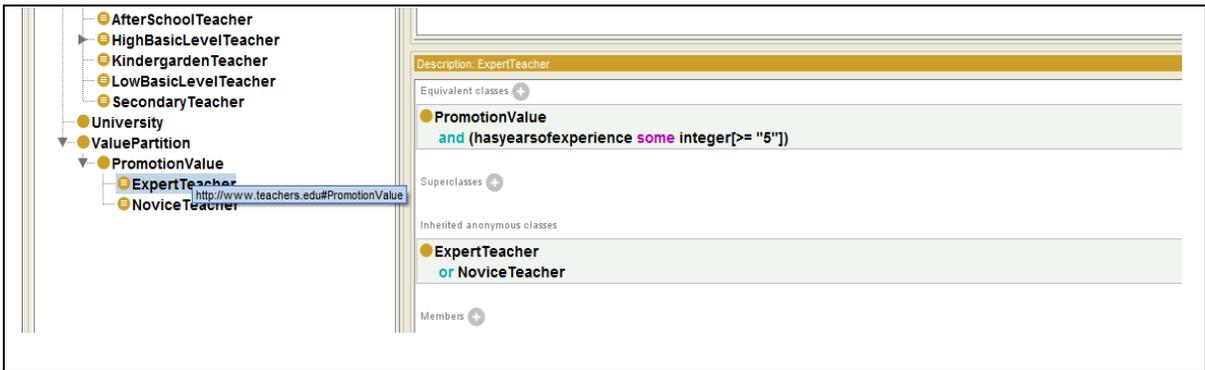


Figure 5.24 Expert Teacher Attribute (`hasyearsofexperience`) range

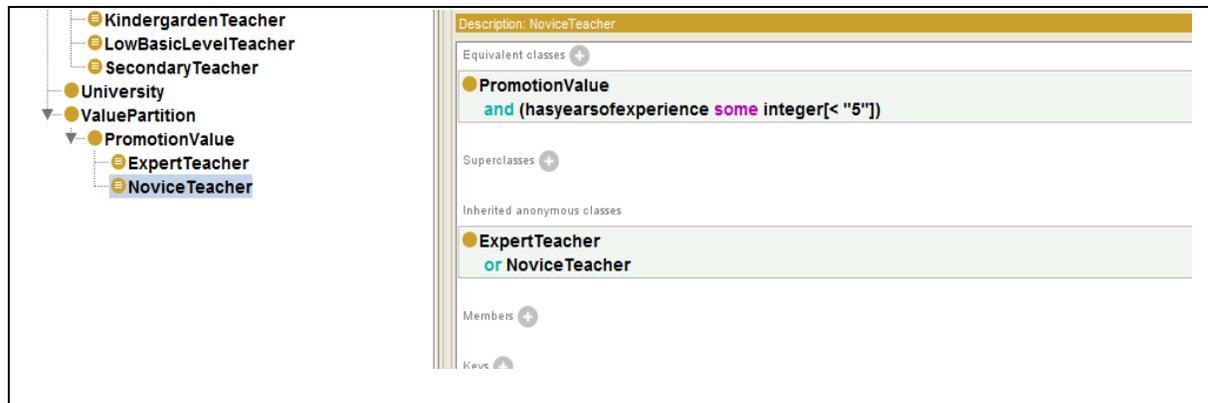


Figure 5.25 Novice teacher Attribute (hasyearsofexperience) range

Any individual had Data Property (hasyearsofexperience) more than five would be classified automatically as Expert teacher. And any individual had Data Property (hasyearsofexperience) less than five years would be classified automatically as Novice teacher.

It is important to know that Data properties are supposed to be functional. For example teacher cannot have two or more ages, or two or more names etc.

5.2.6 A Closure Axiom and Covering Axiom

We use closure axiom on property to restrict the property with universal restriction to say that it can only be filled by the specified fillers. For example in our Ontology, a teacher can teach different levels. In “Level11-12” the only grades must be taught are eleven and twelve grades. To restrict these grades only we make a closure axiom as shown in

Figure 5.26

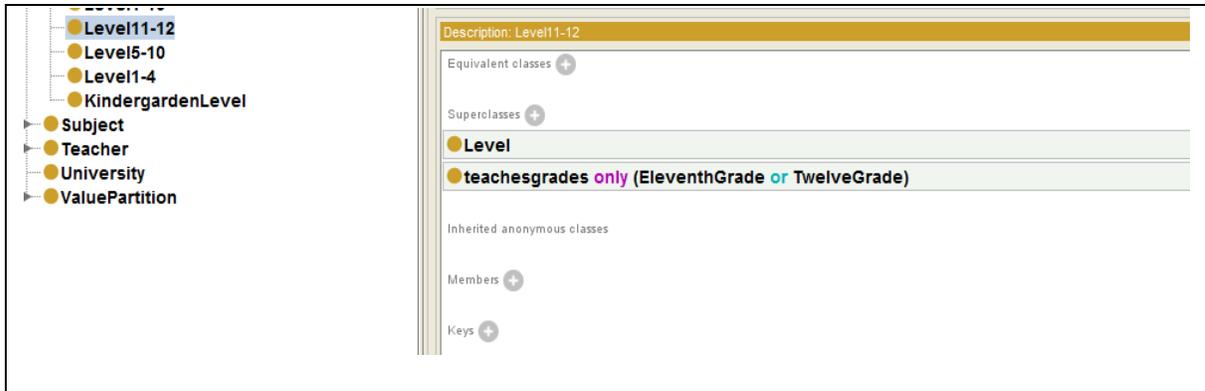


Figure 5.26 A closure Axiom for Secondary Teacher (11-12) Level

We explained in chapter 4 how covering axiom can derive new information. In our case study we covered class “*PromotionValue*” by the classes “*ExpertTeacher*” and “*NoviceTeacher*”. We can derive that to be a member of “*PromotionValue*” class it must be a member of “*ExpertTeacher*” class or “*NoviceTeacher*” class as shown in Figure 5.27

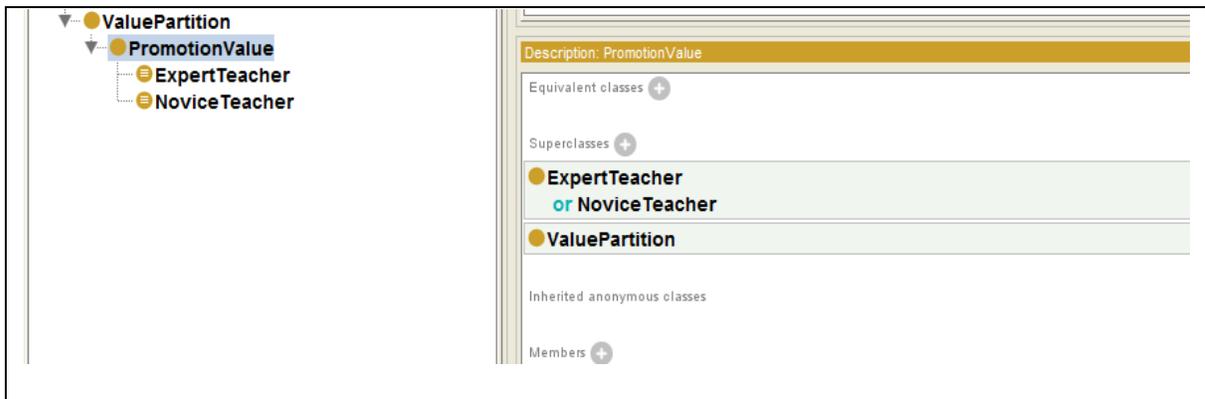


Figure 5.27 Covering Axiom for class PromotionValue

In Appendix 1 all OWL code for our Ontology.

5.3 Pizza Ontology

Pizza Ontology is created to choose from a restaurant menu what kind of Pizza options are there. It classifies Pizza regarding its Topping, Base and Spiciness etc. This example is already implemented at the following link (<http://www.co-ode.org/ontologies/pizza/2007/02/12/>).

5.3.1 Classes and Subclasses

Pizza Ontology contains main classes such as Pizza, PizzaTopping, PizzaBase etc. each class contains subclasses. Figure 5.28 shows classes of Pizza Ontology.

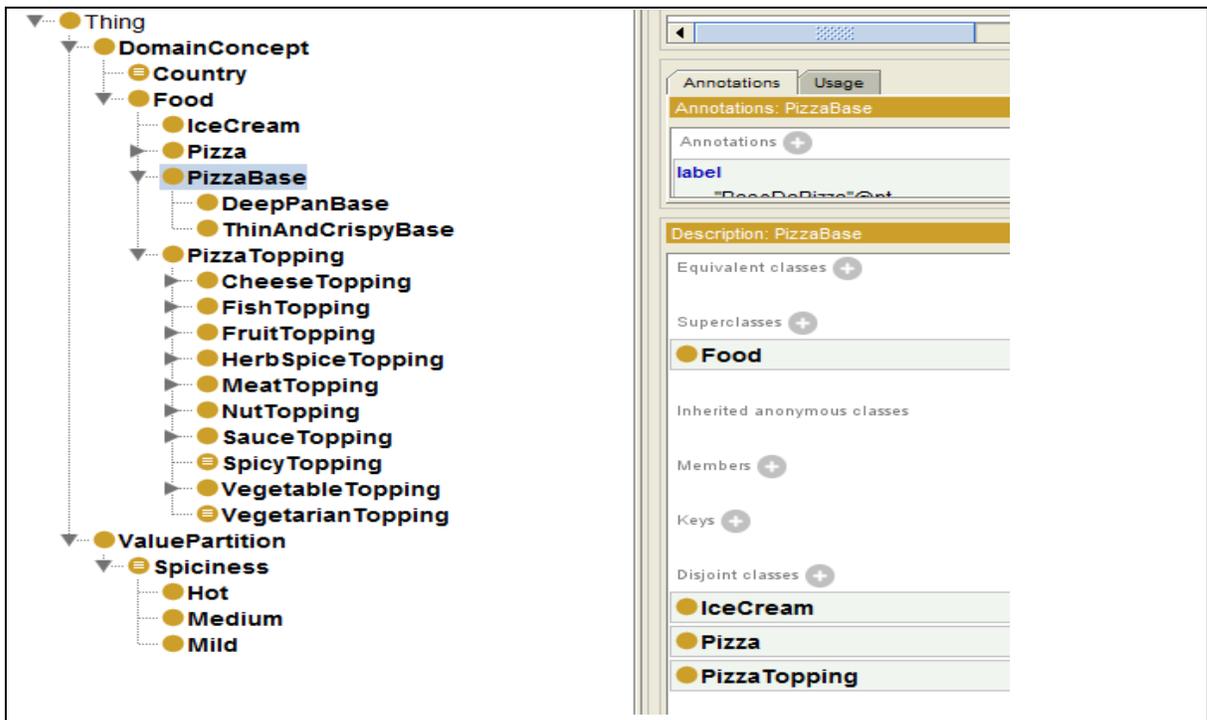


Figure 5.28 Pizza Ontology Classes

5.3.2 Object Properties

Object properties which represent relationships in Ontology are represented in Pizza Ontology. For example, all kinds of Pizzas has base, so all subclasses of Pizza inherit the object Property (hasBase some PizzaBase). Figure 5.29 shows “Cheesey Pizza hasBase some PizzaBase” which is inherited from class Pizza.

Also Figure 5.29 shows necessary and sufficient conditions for Cheesey Pizza class that it (hasTopping some CheeseTopping) that is why these conditions are made in Equivalent class part.

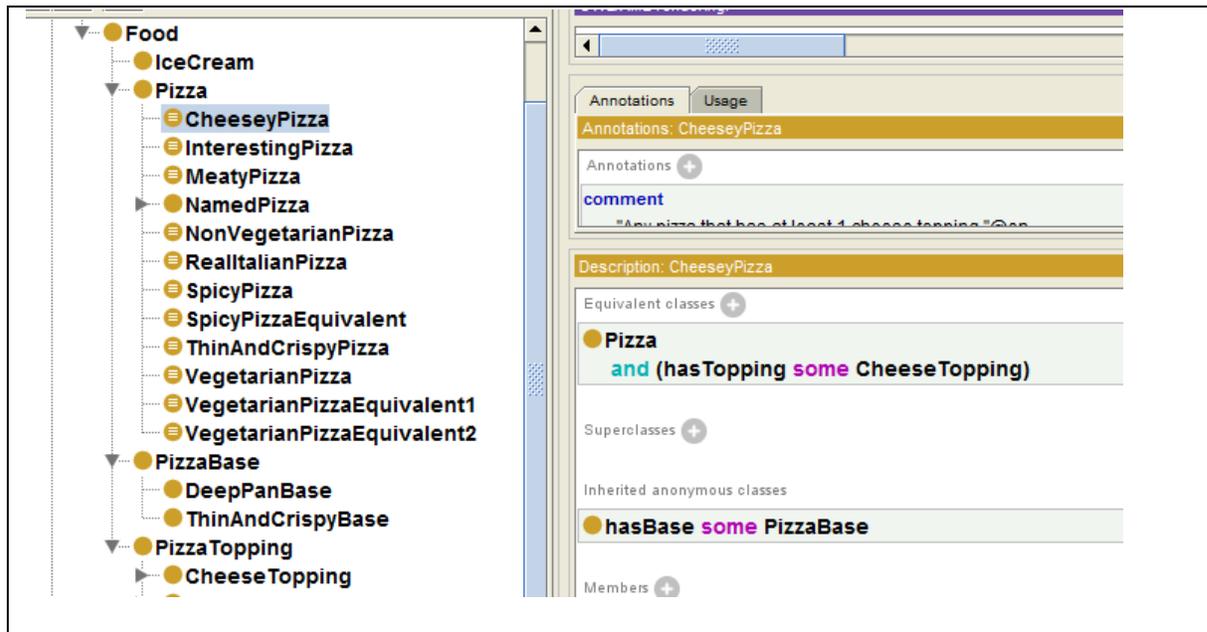


Figure 5.29 Object Properties for CheeseyPizza Class

5.3.3 Datatype Properties

Datatype Properties describe relationships between an individual and data values. And they can be done at class level and at individual level. To determine Pizza calories Datatype Property 'hasCalorificContentValue' was created. Figure 5.30 shows Datatype Property also called Data Property at individual level.

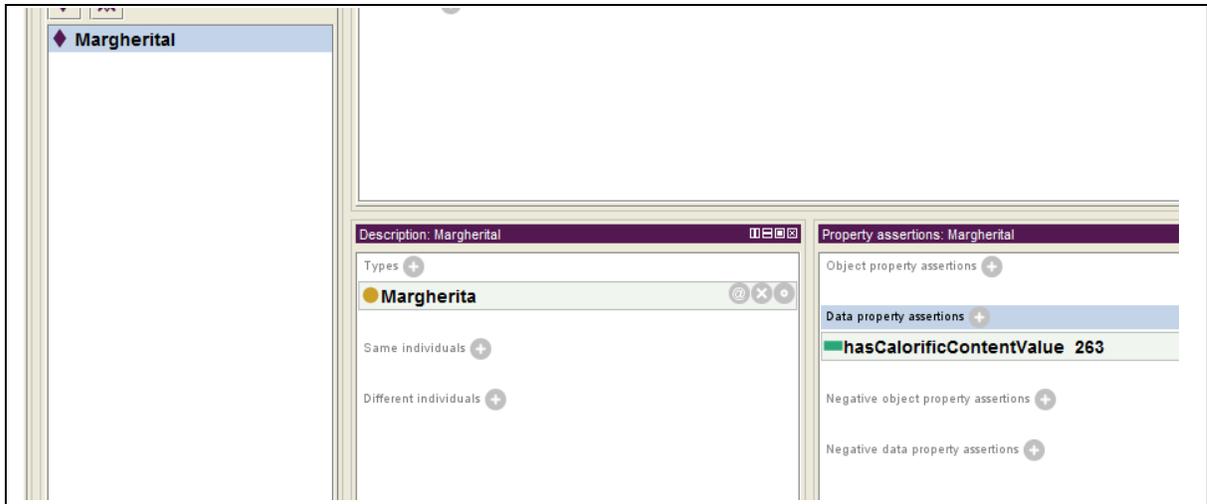


Figure 5.30 Data Properties at Individual Level

Figure 5.31 shows Data Property at class level 'hasCalorificContentValue some integer'.

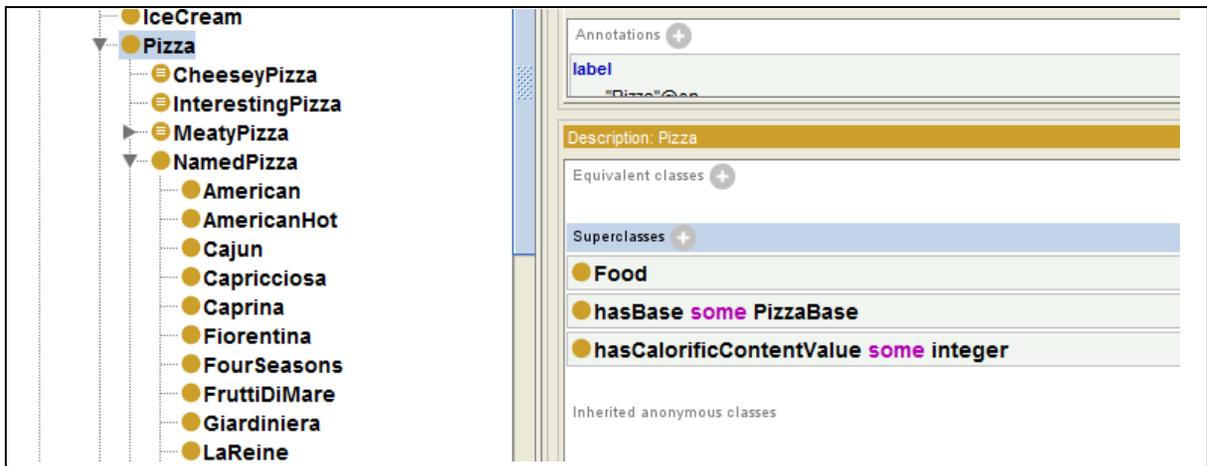


Figure 5.31 Data Properties at Class Level

Data Property can be used to specify restrictions on the possible values as shown in

Figure 5.32 'hasCalorificContentValue' is used to restrict the range for HighCaloriePizza to have more than 400 calories.

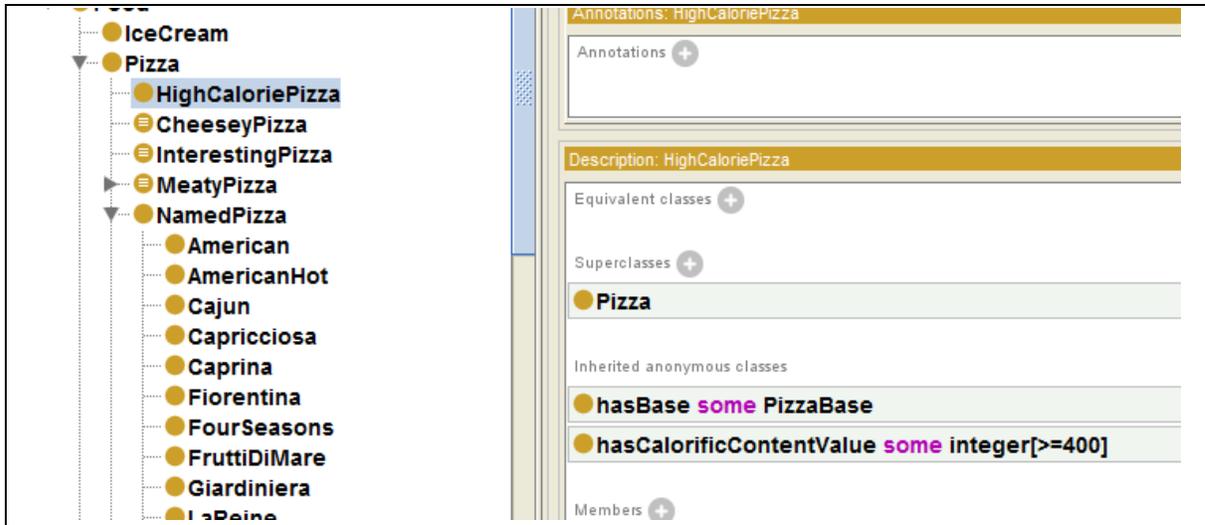


Figure 5.32 Data Property Restriction

5.3.4 has Value Restrictions

Has value restriction which is used to describe the set of individuals that have at least one relationship along a specified Object Property to a specific individual. Figure 5.33 shows the hasValue restriction (hasCountryOfOrigin value Italy) describes the set of individuals such as MozzarellaTopping that have at least one relationship along the hasCountryOfOrigin Object Property to the specific individual Italy (Horridge, et al, 2009).

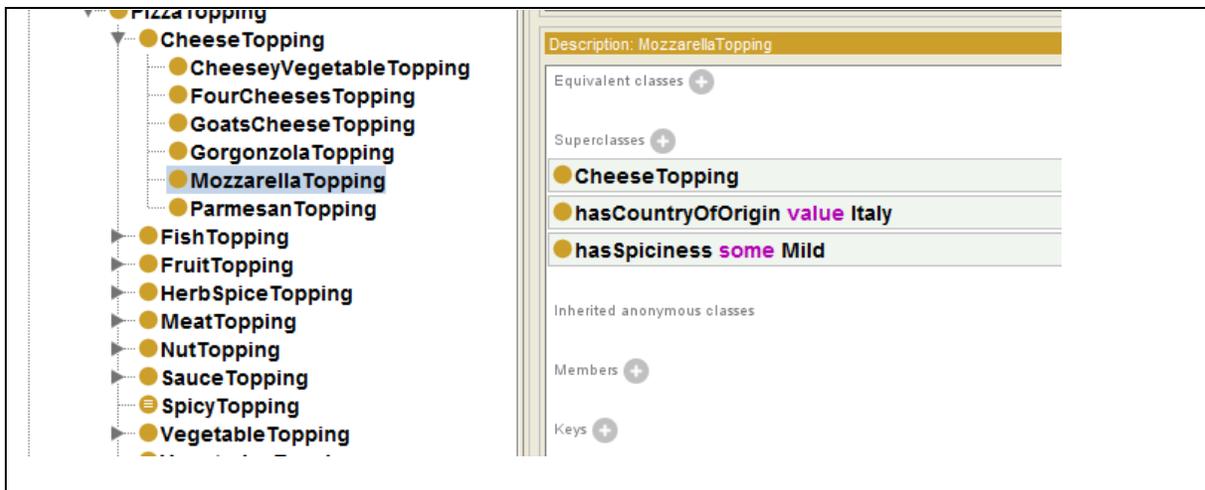


Figure 5.33 has Value Restriction

5.3.5 Cardinality Restrictions

Cardinality Restrictions can be at least, at most or exactly a specified number of relationships between individuals of different classes or between class of individuals and datatype value.

Figure 5.34 shows InterestingPizza has at least three toppings. The minimum Cardinality Restriction is used.

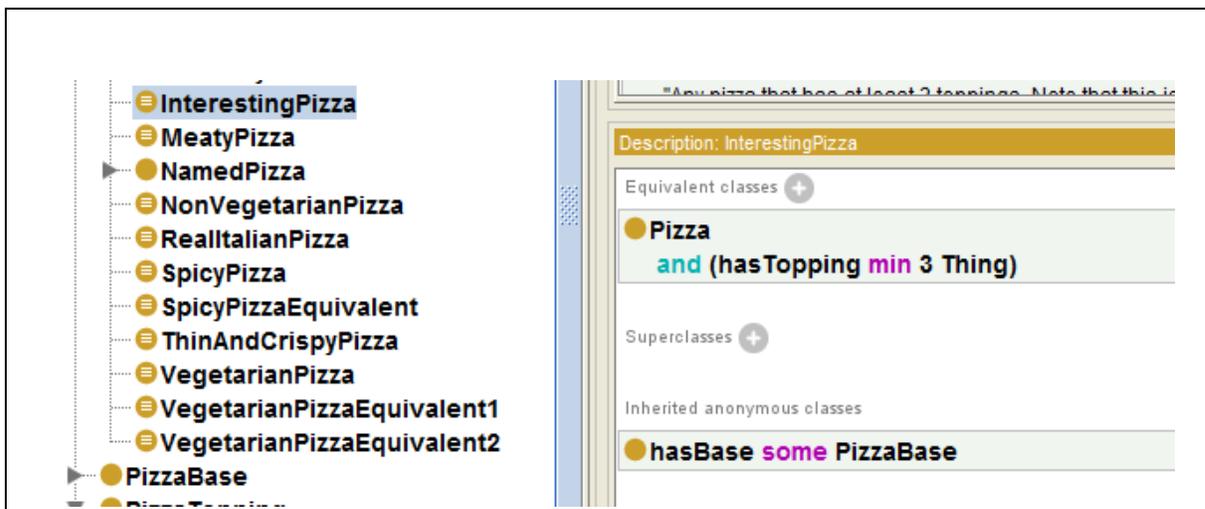


Figure 5.34 Minimum Cardinality Restriction

5.3.6 Quantifier Restrictions

Quantifier Restrictions include Universal Restriction that expressed by the keyword 'only' and Existential Restriction that expressed by 'some' keyword.

Figure 5.35 shows some examples for Quantifier Restrictions.

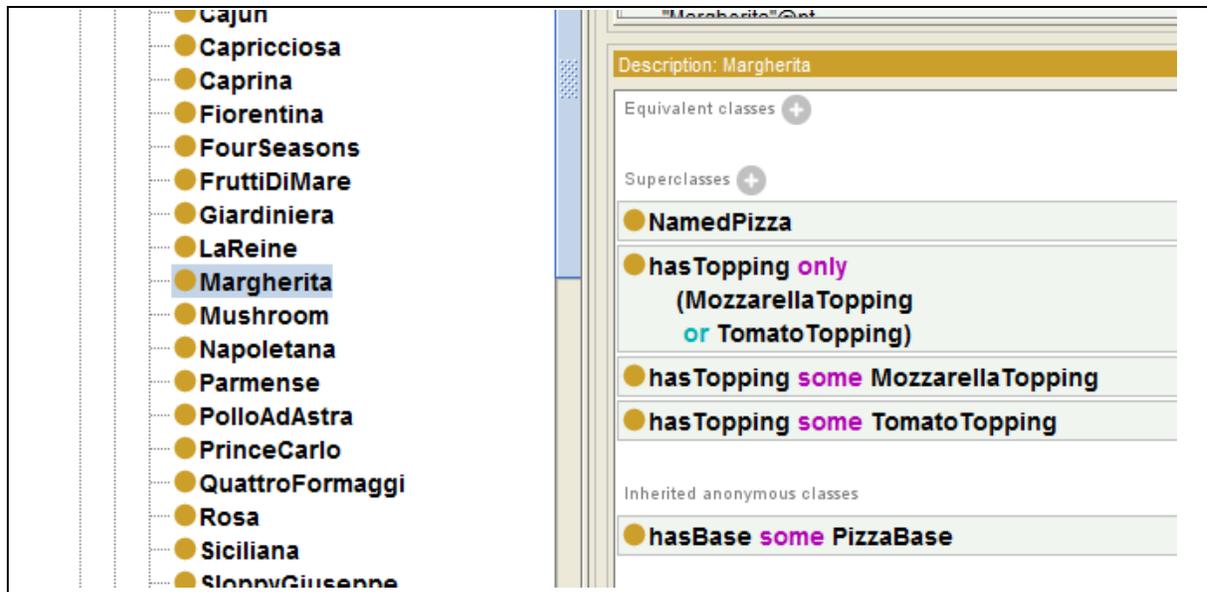


Figure 5.35 Quantifier Restrictions

5.3.7 Value Partitions

Value Partitions can be created to refine class descriptions. For example, a Value Partition called ‘Spiciness’ Value Partition is created to describe the spiciness of PizzaToppings. Value partitions is used to restrict the range to Mild, Medium, and Hot.(Horridge, et al, 2009). Figure 5.36 shows ‘Spiciness’ ValuePartition.



Figure 5.36 Spiciness Value Partition

5.3.8 Covering Axiom and Closure Axiom

Spiciness Value Partition has Covering Axiom to say that any member of Spiciness Value Partition must be a member of either Mild or Medium or hot as shown in Figure 5.37.

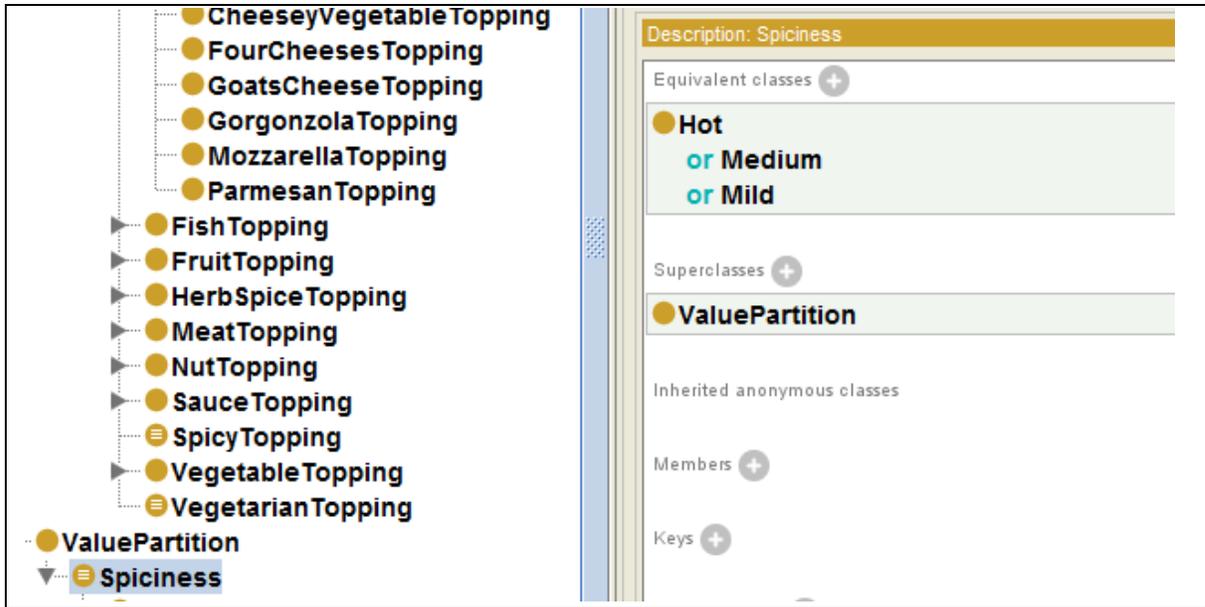


Figure 5.37 Spiciness Covering Axiom

A closure Axiom restrict Object property to have Universal Restriction with a specific filler. Figure 5.38 shows Margherita Pizza with a Closure Axiom for the hasTopping Property. It shows that the topping for Margherita Pizza must be Mozzarella or Tomato Topping only.

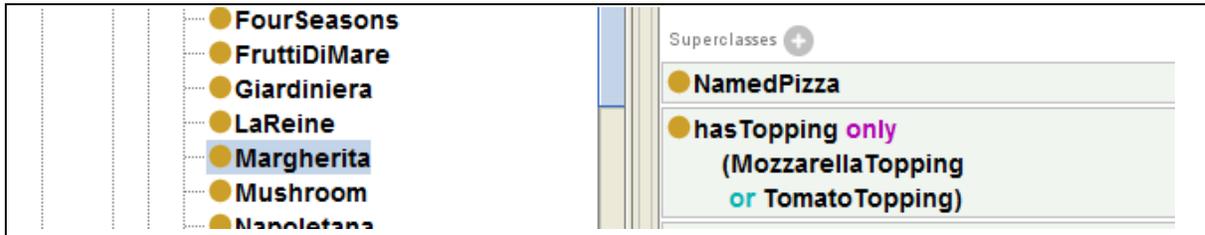


Figure 5.38 Margherita Closure Axiom

5.4 Family Ontology

Family Ontology is already implemented example found at:

(<http://protege.cim3.net/file/pub/ontologies/family.swrl.owl/family.swrl.owl>). Family

Ontology represents family members and how they are related to each other.

5.4.1 Classes and Subclasses

Family Ontology contains main classes and some subclasses as shown in Figure 5.39 such as Man, Woman, Father, Mother, Parent etc.

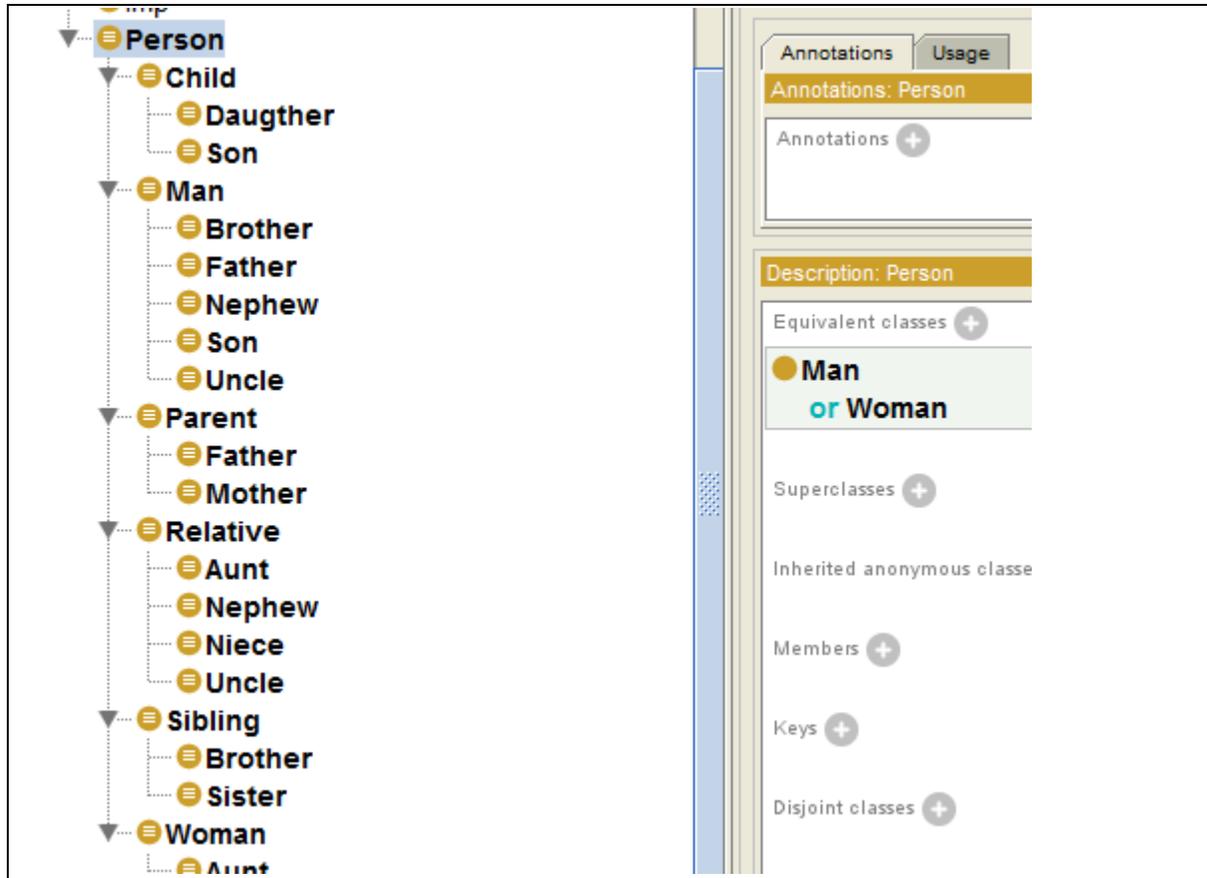


Figure 5.39 Classes and Subclasses of Family Ontology

5.4.2 Object Properties

Object properties in Ontology represent relationships between classes. Figure 5.40 shows some Object Properties in Family Ontology such as hasParent. For example, 'hasParent min 1 thing' explains that every child has parent. And this condition is done in equivalent class part to determine necessary and sufficient conditions to be a member of class child.



Figure 5.40 hasParent Object Property

5.4.3 Datatype properties

Datatype Properties represent attributes that describe classes and they can be represented in Ontology at class level and at individual level. There is not any Datatype Properties in Family Ontology.

5.4.4 hasValue Restrictions

Has value restriction which is used to describe the set of individuals that have at least one relationship along a specified Object Property to a specific individual. Figure 5.41 shows hasValue restriction specified a long hasSex Object Property 'hasSex value Female'.

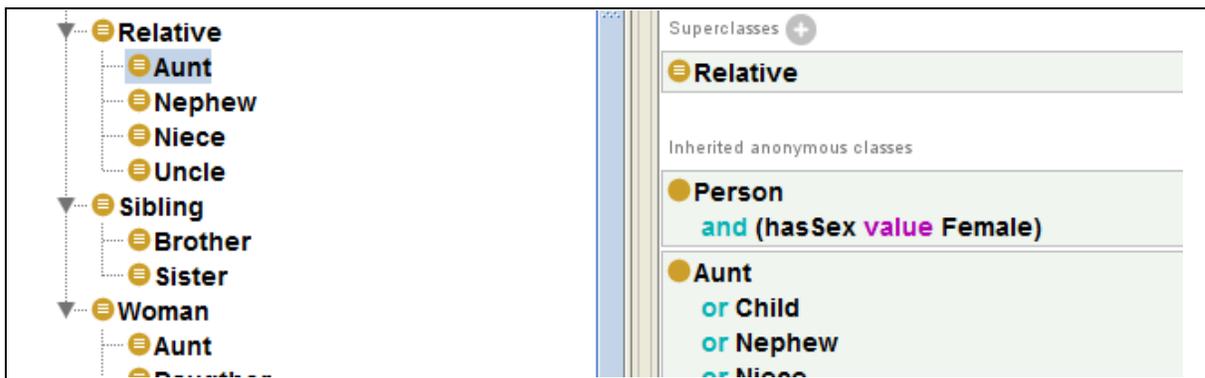


Figure 5.41 hasValue Restriction

5.4.5 Cardinality Restrictions

Cardinality Restrictions can be at least, at most or exactly a specified number of relationships between individuals of different classes or between class of individuals and datatype values.

Figure 5.42 shows minimum Cardinality Restriction ‘hasAunt min 1 Thing’ to explain that nephew has minimum one Aunt.

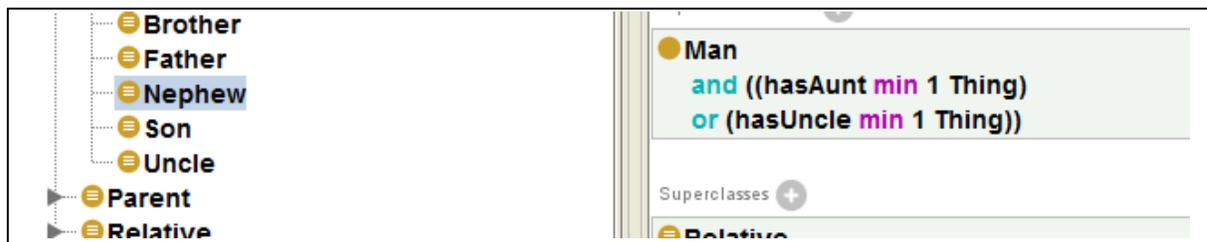


Figure 5.42 Minimum Cardinality Restriction

5.4.6 Quantifier Restrictions

Quantifier Restrictions include Universal Restriction that expressed by the keyword ‘only’ and Existential Restriction that expressed by ‘some’ keyword.

There is not any Quantifier Restrictions in Family Ontology.

5.4.7 Value Partitions

Value Partitions can be created to refine class descriptions. There is not any Value Partitions in Family Ontology.

5.4.8 Covering Axiom and Closure Axiom

A closure Axiom restrict Object property to have Universal Restriction with a specific filler. There is not any Closure Axiom in Family Ontology.

Covering Axiom is used by class Person which is covered to be superclass for the union of classes Man and Women as shown in Figure 5.43.

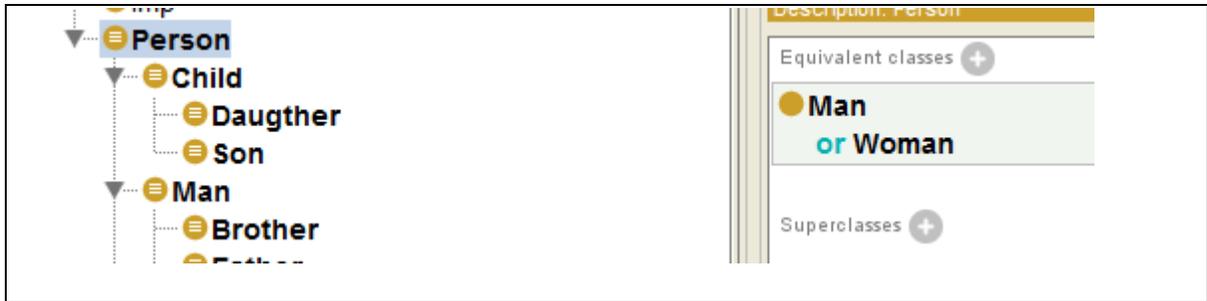


Figure 5.43 Covering Axiom

5.5 Conclusion

In this chapter, we implemented our case study using Protégé tool. We also explained a few already implemented case studies by others to understand how Business Rules are represented in Ontology and to implement our mapping between Business Rules categories and Ontology concepts. These case studies helped us in determining how Ontology concepts are related to each other. The case studies together with mapping will help us in concluding our extraction model to get Business Rules from ontology.

Chapter 6

Ontology-Based Business Rules Extraction Model & Algorithm

6.1 Introduction

In this chapter, we summarize our results. We want to extract Business Rules from an Ontology. These Rules define the structure and control the operation of an enterprise (www.businessrulesgroup.org). In previous chapters, we made a one to one mapping between Ontology and Business Rules that helped us in determining what Ontology concepts must be extracted to get Business Rules. It was clear that Business Rules are captured in ontology by axioms and relationships-constraints between terms. And we implemented case studies that helped us in determining relationships between Ontology concepts. In this chapter, we will provide Ontology Based Business Rules Extraction Model (OBBREM) which is inferred from mapping and the case studies. Since Business Rules are represented clearly and mapped one to one in Ontology, it is easy to extract these Business Rules from Ontology. Then we will propose Ontology Based Business Rules Extraction Algorithm (OBBREA) for extraction Business Rules from Ontology. Finally we will suggest some directions for future work.

6.2 Ontology Based Business Rules Extraction Model (OBBREM)

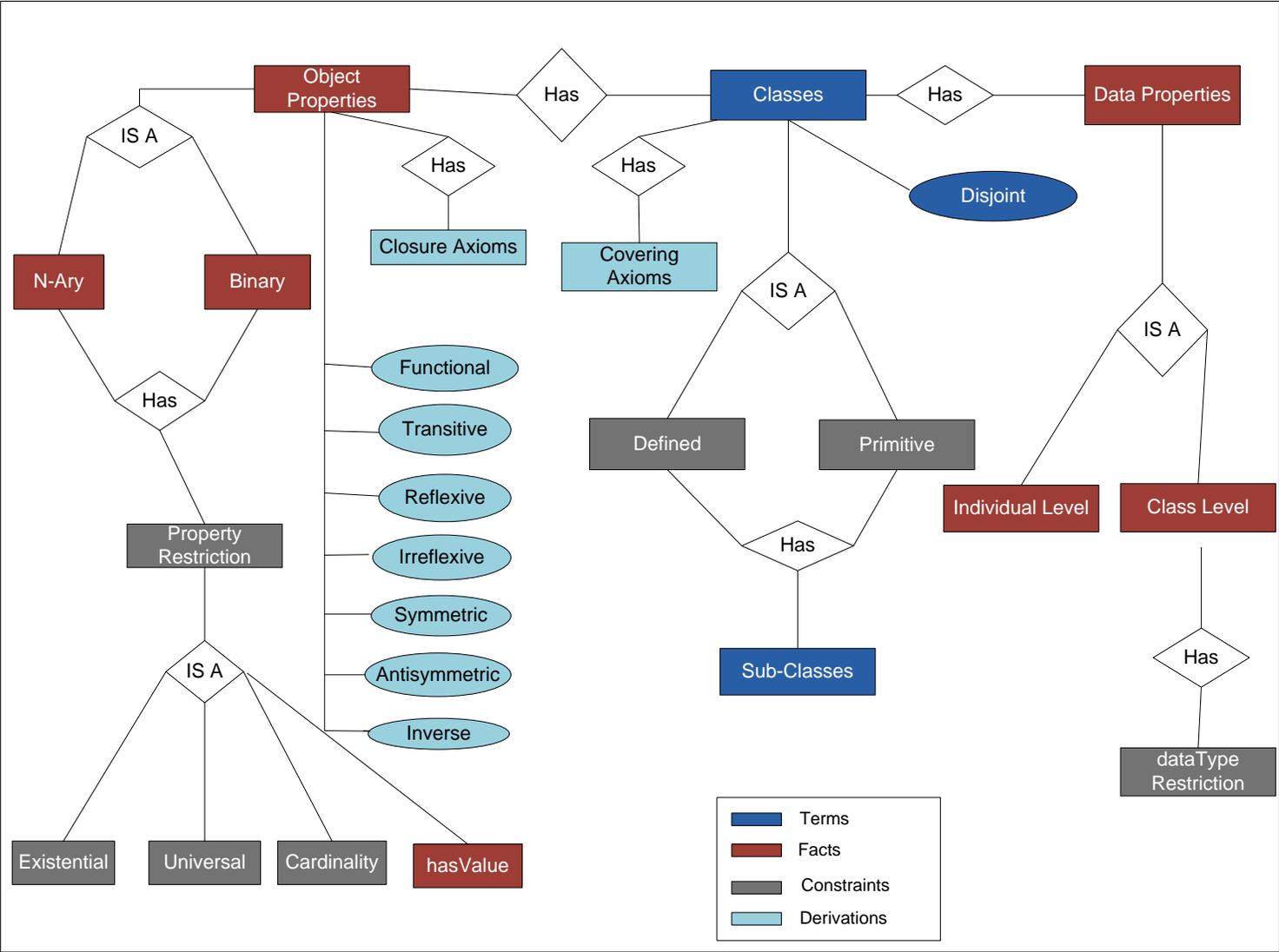


Figure 6.1 Ontology Based Business Rules Extraction Model (OBBREM)

Figure 6.1 shows our suggested model for extraction of Business Rules. Ontology Based Business Rules Extraction Model (OBBREM) represented a road for parsing Ontology code to extract Business Rules. We draw this model depending on our mapping process between Business Rules categories and Ontology concepts, which helped us in determining what do we need to extract from Ontology to get Business Rules. Also our extraction model depend on case studies which helped us in determining the relations between different Ontology concepts, as a consequence we determine how Business Rules are related to each other. We used Entity Relational Model (ERM) conventions in modeling OBBREM with few modifications such as we do not need cardinality or ordinary relationships between entities, we eliminate some concepts such as primary key and foreign key. We used ERM to link concepts to each other. We inferred these concepts and relations between them depending on deep analysis for how Ontology represents Business Rules and how they are mapped to Ontology concepts.

OBBREM suggested extraction for classes of Ontology. Classes represent one of Business Rules Structural Assertions which is Terms. A term is a word or a phrase that has a specific meaning for the business (Business Rules Group, 2000). And since classes represent concepts in Ontology so we can extract Business Terms from them. As shown in Figure 6.1 class can have subclasses and we must extract subclasses to know the hierarchy in terms. Classes and subclasses can be Primitive or Defined. Primitive classes only have necessary conditions. That means members of these classes must satisfy these conditions. But Defined classes have in addition to necessary conditions sufficient conditions. This guaranteed any random member satisfies these necessary and sufficient conditions can be a member of the class. And this information helped in determining what are the exact

conditions and constraints to define Business Rules Terms and sub Terms. A class can have disjoint property. It determines the relation between sibling classes. and it means that the member of a class cannot be a member of another class. It determines multiple inheritance, since if class is determined not to be disjoint then the member of this class can be a member of more than one class. And from this information we can determine how terms have multiple inheritance.

As shown in Figure 6.1 classes have Object Properties which represent the second category of Business Rules Structural Assertions which is Facts. Facts related Terms to each other. “They assert an association between two or more Terms” (Business Rules Group, 2000). Facts expressed Binary and N-ary relationships between Terms. And as its shown in the OBBREM Object Properties (relationships) either Binary or N-ary has Property Restrictions. These Restrictions define the relationship as Existential, Universal, the cardinality of the relationship and hasValue Restrictions. “Existential Restrictions describe classes of individuals that participate in at least one relationship along a specified property to individuals that are members of a specified class” (Horridge, et al, 2009, page38). Existential Restriction is represented by “some” key word and it can express the words “Should be” in Business Rules. The other Property Restriction is Universal Restriction. “Universal Restrictions describe classes of individuals that for a given property only have relationships along this property to individuals that are members of a specified class” (Horridge, et al, 2009, page38). Universal Restriction is represented by “only” key word and it can express the words “Must be” in Business Rules. Another Property Restriction “Cardinality Restrictions which describe the class of individuals that have at least, at most or exactly a specified number of relationships with other individuals or datatype values”

(Horridge, et al, 2009, page73). Cardinality Restrictions specified the relationship cardinality that explained numbers of individuals of each class participated in the relationship. So we can extract relationships and their cardinality, in other way relationship constraints from Object Properties and their cardinality restrictions. The last kind of Property Restriction is hasValue Restriction which “describes the set of individuals that have at least one relationship along a specified property to a specific individual” (Horridge, et al, 2009, page 92) and this helps in extracting attributes that describe classes. Business Rules like $X \langle \text{connecting verb} \rangle Y$, $X \text{ contains } Y$ could be extracted from Object Properties.

Also Its shown in Figure 6.1 Object Properties have some characteristics such as Symmetric, Reflexive, Transitive, Functional, Inverse, Irreflexive or Antisymmetric. And we can extract Business Rules Derivations from these information. For example if Property P is reflexive we can derive information from that P can be related to itself. The same for other characteristics of object properties we can derive new information depending on property characteristic definition. We explained the all of these definitions in chapter 4.

We can extract another derived information i.e. Business Rules Derivations from Object Properties. Since Object Properties have Closure Axioms that help in deriving new information. “a closure axiom on a property consists of a universal restriction that acts along the property to say that it can only be filled by the specified fillers” (Horridge, et al, 2009, page 64). We can derive that the restriction has a class (filler) that is the union of the classes that occur in the existential restrictions for the property.

Also its shown in Figure 6.1 the class can have a Covering Axiom and we can derive a new information from that depending on the definition of Covering Axiom . Covering Axiom

means that it marked itself as a class that is the union of classes being covered. Put in another way if we have three classes A, B and C and classes B and C are sub classes of class A, and class A is covered by class B and C, then we can derive that a member of class A must be a member of either class B or class C. i.e. class A would be a super class of B U C. And this helps in knowing how Business Terms affected by each other.

Also as its shown in OBBREM classes have Datatype Properties(Attributes). “ Datatype Properties describe relationships between an individual and data values” (Horridge, et al, 2009, page77).Datatype Properties represented as Attributes in Business Rules. “ An attribute expresses a fact in which a term describes some aspect of another term” (Hay, et al,2000, page21). In other words attribute is “ a specialization of Fact that expresses a ‘has property of’ relationship between terms, specifically an association between an entity type and a domain/abstract data type.” (Business Rules Group, 2000, page23). So attributes that describe each class can be extracted from Datatype Properties. Any Business Rule like X is A type of Y could be extracted from Datatype Properties. Datatype Properties could be defined at class level for all individuals. Or it can be defined at individual level as explained in case studies.

Also Datatype Properties at class level has Datatype Restriction that is used to specify restrictions on possible values such as specifying a range of values for a number (Horridge, et al, 2009). We can extract constraints from Datatype Restrictions.

Entities in Figure 6.1 can be considered as key words or clues for parsing Ontology to extract Business Rules without interfere of Business Experts. From this we can infer that Business Rules are not hidden in Ontology Code.

Ontology separates Business Rules from the data and presentation layers which helps in extracting these Business Rules. Business Rules are represented in Ontology in conceptual way. they are expressed in Ontology as they are expressed in real world. So extracting them from Ontology ease the process of maintaining and modifying them by analysts and Business Rules experts.

6.3 Ontology Based Business Rules Extraction Algorithm (OBBERA)

Our OBBREM model can be translated into algorithm that is used for extracting Business Rules from Ontology. This algorithm is inferred from the case studies and the mapping between Business Rules categories and Ontology concepts. Since we know how to represent each Business Rule in Ontology so using backtracking analysis (figgis, et al) we can infer how to extract Business Rules into algorithm. here is the algorithm:

```
Begin of algorithm
Parse OWL Code
Loop
    Look for classes and check for disjoint property to determine multiple inheritance
    Begin
        If defined class then
            Begin
                Print class name
                Print “ this class has necessary and sufficient conditions that means any random
```

individual satisfy these conditions would be a member of the class ”

End

Elseif Primitive Class then

Begin

Print class name

Print ”this class has necessary conditions only which means all members of the class

must satisfy these conditions”

End

/* looking for relationships*/

Look for Object Property that related to the class

If binary Object Property then

Begin

Look for characteristics of Object Property

Print derived business rules depending on characteristics Definitions

Look for Property Restriction

If Property Restriction is Existential then

Begin

Look for related filler class

Print class name "" Object Property "" 'some' "" filler class

End

Elseif Property Restriction is Universal then

Begin

Look for related filler class

Print class name "" Object Property "" 'Only' "" filler class

End

Else if Property Restriction is Cardinality

Look for related filler class

Print class name "" Object Property "" Cardinality Value "" filler class

End

Else if Property Restriction is hasValue

Look for related filler individual

Print class name "" Object Property "" 'Value' "" filler individual

Print attribute (filler individual) related to the class

```
End

Elseif N-ary Object Property

    Do the same as Binary Object Property just check number of Object Properties related to
the same class

end loop

/* End looking for relationships*/

/* looking for attributes*/

Begin

For each class do

Begin

    look for Data Properties at class level

    print Attributes

    look for Datatype (Data Property) Restriction

    print conditional constraints depending on Datatype Restriction

End for

For each individual do

Begin

    If attribute not Printed then
```

```
Begin

    look for Data Properties at individual level

    Print Attributes

End if

End for

End /* looking for attributes*/

Begin /* looking for derived Business Rules*/

    For each covered class do

        Find classes that covered

        Print derived Business Rule depending on covering axiom definition

    End for

    For each Universal Restriction of Object Properties do

        Find specified Fillers

        Print derived Business Rule depending on Universal Restriction definition

    End for

End /*looking for derived Business Rules*/

End algorithm
```

6.4 Future Work

In this thesis we proved that Ontology can be used as a tool to extract Business Rules easily. We suggested a model for extraction Business Rules that Ontology Based Business Rules Extraction Model (OBBREM), we concluded our model with a mapping between Business Rules categories and Ontology concepts and the case studies. Then using backtracking analysis we proposed an Algorithm for extraction Business Rules that is Ontology Based Business Rules Extraction Algorithm (OBBREA). In future we intend to provide a tool that could parse Ontology depending on our (OBBREA) to extract Business Rules. To help in the analysis process for any new application belongs to certain domain. So a new application will not be built from scratch. It will use the domain Business Rules.

References

1. Braun Simone, Schmidt Andreas and Walter Andreas. (2007). “Ontology Maturing: a Collaborative Web 2.0 Approach to Ontology Engineering”, Copyright is held by author/owner, Banff, Canada
2. Goudos Sotirios K., Peristeras Vassilios and Tarabanis Konstantinos. (2006). “Asemantic Web application for matching a Citizen’s Profile to Entitled Public Services”, Center for Research and Technology Hellas, Thessaloniki, Greece
3. Goudos Sotirios K., Peristeras Vassilios and Tarabanis Konstantinos. (2006). “Mapping Citizen Profiles to Public Administration Services Using Ontology Implementations of the Governance Enterprise Architecture (GEA) models”, Informatics and Telematics Institute, Center for Research and Technology Hellas, Thessaloniki, Greece and University of Macedonia, Information Systems Laboratory, Thessaloniki, Greece
4. Spyns Peter, Meersman Robert and Jarrar Mustafa. (2002). “Data Modelling Versus OntologyEngineering”, Vrije Universiteit Brussel-StarLab, Pleinlaan2, Building G-10 B-1050, Brussel, Belgium
5. Honour Eric C., Valerdi Ricardo. (2006). “Advancing an Ontology for Systems Engineering to allow consistent Measurement”, proceedings of the Conference on Systems Engineering Research, <http://www.hcode.com/seroi/documents/06-Honour-Advancing%20Ontology.pdf>
6. Chen Ruey-Shun, Chang Chan-Chine, Chen Chia-Chen and Chi Isabel. (2005). “Ontology-Based User Requirements analysis for Banking application”, 2-9525435-0 IEEE Sitis

7. Breaux Travis D. and Anton Annie I. (2006). "A Systematic Method for Acquiring Regulatory Requirements: A Frame-Based Approach", Department of Computer Science North Carolina State University
8. Noy Natalya F., McGuinness Deborah L.. "Ontology Development 101: A Guide to Creating Your First Ontology", Stanford University, StanfordCA, 94305
9. Saggion Horacio, Funk Adam, Maynard Diana, Bontcheva Kalina. (2007). "Ontology-based Information Extraction for Business Intelligence", Department of Computer Science, University of Sheffield, United Kingdom
10. The Business Rules Group. (2000). "Defining Business Rules ~ What Are They Really?", Guide Business Rules Project, Final Report revision 1.3
11. Baxter Ira, Hendryx Stan. (2005). "A Standard-Based Approach to Extracting Business Rules". Semantic Designs
12. Owen James. (2004). "Business Rules management systems", <http://www.infoworld.com/t/business/business-rules-management-systems-548>
13. Ulrich William M.. (2005). "Extracting Business Rules from Existing Systems", BPM Strategies Magazine, http://63.249.72.168/IT_Arch_Transformation_Articles/arch_extracting_business_rules.htm
14. Blog Eswaran's. (2010). "Extracting Business Rules from Legacy Code", <http://www.requirementsnetwork.com/node/2452>
15. Sneed Harry M. & Erdos Katalin. (1996). "Extracting Business Rules from Source Code", 0-8186-7283-8/96 IEEE
16. Wang Xinyu, Sun Jianling, Yang Xiaohu, He Zhijun, Maddineni Srimi. (2004). "Business Rules Extraction from Large Legacy Systems", 1534-5351/04 IEEE

17. Tobon Herbert Gomez and Franco Aldrin Fredy Jaramillo. (2010). “ Business Rules Extraction from Business Process Specifications Written in Natural Language”, Business Rules Journal, Vol.11, No.7,
URL:<http://www.BRCommunity.com/a2010/b543.html>
18. Rosenberg Florian and Dustdar Schahram. (2005). “Design and Implementation of a Service-Oriented Business Rules Broker”, 0-7695-2384-6/05, IEEE
19. Horridge Matthew, Knublauch Holger, Rector Alan, Stevens Robert, Wroe Chris, Jupp Simon, Moulton Georgina, Stevens Robert and Drummond Nick. (2009). “A Practical Guide To Building OWL Ontologies Using Protégé 4 and CO-ODE Tools Edition 1.2”, the university of Manchester
20. Bugaite Diana, Vasilecas Olegas. (2005). “Framework on application domain ontology transformation into set of Business Rules”, International Conference on Computer Systems and Technologies- CompSysTech,
<http://ecet.ecs.ru.acad.bg/cst05/Docs/cp/SIII/IIIB.8.pdf>
21. Vasilecas Olegas , Bugaite Diana. (2007). “An Algorithm for the Automatic Transformation of Ontology Axioms into a Rule Model”, International Conference on Computer Systems and Technologies CompSysTech’07 , ACM ISBN: 978-954-9641-50-9
22. Vasilecas Olegas, Būgaitė Diana. (2005). “The Method for Ontology Components Transformation into the Set of Business Rules”, ISSN 1392-0561.
INFORMACIJOS MOKSLAI
23. Hayes Pat, Welty Chris. (2006). “Defining N-ary Relations on the Semantic Web”,
<http://www.w3.org/TR/swbp-n-aryRelations>

24. Gottesdiener Ellen. (1997). "Business Rules Show Power, Promise", issue of Application Development Trends, vol 4., no. 3
25. Choi Ben. "Algorithms analysis and Design", CS 530 Algorithms Analysis and Design slides
26. Ankolekar Anupriya, Paolucci Massimo, and Sycara Katia. (2005). "Towards a Formal Verification of OWL-S Process Models",
<http://www.springerlink.com/content/p5q5535946648675/>
27. The Business Rules Group. (2001). "Defining Business Rules ` What Are They Really?", http://www.businessrulesgroup.org/first_paper/br01c5.htm
28. (2009). "DAML Services", <http://www.daml.org/services/owl-s/>
29. (2011). "Introduction to OWL", http://www.w3schools.com/RDF/rdf_owl.asp
30. Petal-Schneider Peter F.. (2004). "What is OWL (and why should I care?)", American Association for Artificial Intelligence, (www.aaai.org)
31. McGuinness Deborah. (2004). "OWL Web Ontology Language Overview",
<http://www.w3.org/TR/owl-features/>
32. Antoniou Grigoris and Harmelen Frank. "Web Ontology Language OWL", Department of Computer Science, University of Crete, Department of AI, Vrije Universiteit Amsterdam,
citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.6.9313&rep
33. Patel-Schneider and Horrocks Ian, et al. (2006-2007). "OWL 1.1 Web Ontology Language Overview", <http://www.webont.org/owl/1.1/overview.html#2.2>
34. (2007). "Web Ontology Language OWL",
<http://www.obitko.com/tutorials/ontologies-semantic-web/web-ontology-language-owl.html>

35. Bechhofer Sean, Harmelen Frank, Hendler Jim, Horrocks Ian, McGuinness Deborah, Patel-Schneider Peter, Stein Lynn. (2003). "OWL Web Ontology Language Reference", <http://www.daml.org/2002/06/webont/owl-ref-proposed>
36. (2007). "Web Ontology Language – OWL", <http://marinemetadata.org/references/owl>
37. Paulquek. "Ontology", Adapted from Barry Smith's draft http://queksiewkhoo.tripod.com/ontology_01.pdf
38. Shirky Clay. (2005). "Ontology is Overrated: Categories, Links and Tags", http://shirky.com/writings/ontology_overrated.html
39. Gurber Thomas. (1993). "A Translation Approach to Portable Ontology Specifications", Knoweldge Systems Laboratory Technical Report KSL 92-71
40. Guarino Nicola. (1998). "Formal Ontology and Information Systems", Proceedings of Fois98, IOS Press, pp 3-15
41. Gurber Thomas. (1991). "The Role Common Ontology in Achieving Sharable Reusable Knowledge Bases", Principles of Knowledge Representation and Reasoning, Proceedings of the second International Conference
42. Martin David, Burstein Mark, et al. "OWL-S: Semantic Markup for Web Services", <http://www.ai.sri.com/daml/services/owl-s/1.2/overview/>
43. Karthick R.(2009). "N-Queens Problem In Design And Analysis Of Algorithm Using Back Tracking Method", <http://www.articlesbase.com/computers-articles/nqueens-problem-in-design-and-analysis-of-algorithm-using-back-tracking-method-1306520.html>

44. Figgis Jane, Zubrick Ann, Butovac Anne and Alderson Anna, AAAJ Consulting Group. “Backtracking Practice and Policies to Research”,
<http://www.dest.gov.au/archive/highered/respubs/impact/pdf/3.pdf>
45. Hodrob Rami, Jarrar Mustafa. (2010). “ORM to OWL 2DL Mapping”, In Proceeding of the international Conference on Intelligent Semantic Web Applications and Services. Pages 131-137. ACMISBN9781450304757
46. Demey Jan, Jarrar Mustafa, Meersman Robert. (2002). “Interoperability between Rule Modeling Systems”, early version of this paper was presented in Rule Markup Languages for Business Rules on the Semantic Web Workshop
47. Joukhadar Ammar, Al-Maghout Hala. (2005). “ Improving Agility in Business Applications using Ontology Based Multilingual Understanding of Natural Business Rules”, Information Technology Faculty, Damascus University
48. Koschmider Agnes and Oberweis Andreas. (2005). “Ontology Based Business Process Description”, Institute of Applied Informatics and Formal, Proceedings of the CAiSE’05 workshops
49. Li Zhanjun, Ramani Karthik. (2007). “Ontology Based Information Extraction and Retrieval”, Cambridge University Press 0890-0604/07
50. Ceravolo Paolo, Fugazza Cristiano, Leida Marcello. (2004). “Modeling Semantics of Business Rules”, Universit a degli Studi di Milano, Dipartimento di Tecnologie dell'Informazione via Bramante, 65 26013 Crema (CR), Italy
http://www.crema.unimi.it/Biblioteca/Note_pdf/118.pdf
51. Jenz Dieter E., Gmbh Partner. (2003). “Ontology-Based Business Process Management”, http://www.bpiresearch.com/Resources/WP_BPMVision.pdf

52. Demey Jan, Jarrar Mustafa and Meersman Robert. (2000). “A Markup Language for ORM Business Rules”, VUB STARLab Vrije Universiteit Brussel, Pleinlaan 2, 1050 Brussels – Belgium, <http://xml.coverpages.org/ORM-200206.pdf>
53. Business Rules Group. (2001), “Defining Business Rules ~ What Are They Really?”, Chapter 1, http://www.businessrulesgroup.org/first_paper/br01c1.htm
54. “Logic, Proofs”, chapter 1, <http://www.math.northwestern.edu/~mlerma/courses/cs310-05s/notes/dm-logic.pdf>
55. Drummond Nick, Horridge Matthew, Stevens Robert, Wroe Chris, Sampaio Sandra. (2007). “Pizza Ontology v1.5 (2007/02/12)”, The University of Manchester, <http://www.co-ode.org/ontologies/pizza/2007/02/12/>
56. Kalibatiene Diana. (2010). “Protégé-OWL Problems with Launching Family Ontology”, <http://protege.cim3.net/file/pub/ontologies/family.swrl.owl/family.swrl.owl>
57. (2006). “Food Ontology”, <http://www.w3.org/TR/owl-guide/food.rdf>
58. Hobbs Jerry, Pan Feng. (2006). “Time Ontology in OWL”, W3C Working draft, <http://www.w3.org/TR/2006/WD-owl-time-20060927/>
59. Ministry of Education and Higher Education. (2008). “Teacher Education Strategy in Palestine”, Ministry of Education and Higher Education

Appendices

Appendix 1: OWL Rendering XML Code For Qualified Teacher Ontology

```
<?xml version="1.0"?>
<!DOCTYPE Ontology [
  <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
  <!ENTITY xml "http://www.w3.org/XML/1998/namespace" >
  <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
  <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
]>

<Ontology xmlns="http://www.w3.org/2002/07/owl#"
  xml:base="http://www.teachers.edu"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:xml="http://www.w3.org/XML/1998/namespace"
  ontologyIRI="http://www.teachers.edu">
  <Prefix name="xsd" IRI="http://www.w3.org/2001/XMLSchema#"/>
  <Prefix name="owl" IRI="http://www.w3.org/2002/07/owl#"/>
  <Prefix name="" IRI="http://www.w3.org/2002/07/owl#"/>
  <Prefix name="rdf" IRI="http://www.w3.org/1999/02/22-rdf-syntax-ns#"/>
  <Prefix name="rdfs" IRI="http://www.w3.org/2000/01/rdf-schema#"/>
```

<Annotation>

<AnnotationProperty abbreviatedIRI="rdfs:comment"/>

<Literal datatypeIRI="&rdf;PlainLiteral">This Ontology built for teacheres to determine who is qualified and unqualified teacher regarding their specilastes and teaching grades(levels).</Literal>

</Annotation>

<Declaration>

<Class IRI="#AfterSchoolTeacher"/>

</Declaration>

<Declaration>

<Class IRI="#Arabic"/>

</Declaration>

<Declaration>

<Class IRI="#BAinArts"/>

</Declaration>

<Declaration>

<Class IRI="#BAinEducationwithMajorinPreschoolEducation"/>

</Declaration>

<Declaration>

<Class IRI="#BAinEducationwithMajorinTeachingHighBasicLevel"/>

</Declaration>

<Declaration>

<Class IRI="#BAinEducationwithMajorinTeachingLowBasicLevel"/>

</Declaration>

```
<Declaration>
  <Class IRI="#BAinSciences"/>
</Declaration>
<Declaration>
  <Class IRI="#Certificate"/>
</Declaration>
<Declaration>
  <Class IRI="#Chemistry"/>
</Declaration>
<Declaration>
  <Class IRI="#ChristianEducation"/>
</Declaration>
<Declaration>
  <Class IRI="#DiplomainEducationHighBasicLevel"/>
</Declaration>
<Declaration>
  <Class IRI="#DiplomainEducationSecondaryLevel"/>
</Declaration>
<Declaration>
  <Class IRI="#DiplomainTeacherEducation"/>
</Declaration>
<Declaration>
  <Class IRI="#EighthGrade"/>
</Declaration>
```

```
<Declaration>
  <Class IRI="#EleventhGrade"/>
</Declaration>
<Declaration>
  <Class IRI="#EnglishLanguage"/>
</Declaration>
<Declaration>
  <Class IRI="#ExpertTeacher"/>
</Declaration>
<Declaration>
  <Class IRI="#FifthGrade"/>
</Declaration>
<Declaration>
  <Class IRI="#FirstGrade"/>
</Declaration>
<Declaration>
  <Class IRI="#ForthGrade"/>
</Declaration>
<Declaration>
  <Class IRI="#FrenchLanguage"/>
</Declaration>
<Declaration>
  <Class IRI="#Geography"/>
</Declaration>
```

```
<Declaration>
  <Class IRI="#Geology"/>
</Declaration>
<Declaration>
  <Class IRI="#Grade"/>
</Declaration>
<Declaration>
  <Class IRI="#HighBasicLevelTeacher"/>
</Declaration>
<Declaration>
  <Class IRI="#HighBasicLevelTeacherProgramB"/>
</Declaration>
<Declaration>
  <Class IRI="#HighBasiclevelTeacherProgramA"/>
</Declaration>
<Declaration>
  <Class IRI="#History"/>
</Declaration>
<Declaration>
  <Class IRI="#IslamicEducation"/>
</Declaration>
<Declaration>
  <Class IRI="#KindergardenGrade"/>
</Declaration>
```

```
<Declaration>
  <Class IRI="#KindergardenLevel"/>
</Declaration>
<Declaration>
  <Class IRI="#KindergardenTeacher"/>
</Declaration>
<Declaration>
  <Class IRI="#Level"/>
</Declaration>
<Declaration>
  <Class IRI="#Level1-10"/>
</Declaration>
<Declaration>
  <Class IRI="#Level1-12"/>
</Declaration>
<Declaration>
  <Class IRI="#Level1-4"/>
</Declaration>
<Declaration>
  <Class IRI="#Level11-12"/>
</Declaration>
<Declaration>
  <Class IRI="#Level5-10"/>
</Declaration>
```

```
<Declaration>
  <Class IRI="#Level5-12"/>
</Declaration>
<Declaration>
  <Class IRI="#Level8-12"/>
</Declaration>
<Declaration>
  <Class IRI="#LowBasicLevelTeacher"/>
</Declaration>
<Declaration>
  <Class IRI="#Math"/>
</Declaration>
<Declaration>
  <Class IRI="#NinthGrade"/>
</Declaration>
<Declaration>
  <Class IRI="#NoviceTeacher"/>
</Declaration>
<Declaration>
  <Class IRI="#Physics"/>
</Declaration>
<Declaration>
  <Class IRI="#PromotionValue"/>
</Declaration>
```

```
<Declaration>
  <Class IRI="#Science"/>
</Declaration>
<Declaration>
  <Class IRI="#SecondGrade"/>
</Declaration>
<Declaration>
  <Class IRI="#SecondaryTeacher"/>
</Declaration>
<Declaration>
  <Class IRI="#SeventhGrade"/>
</Declaration>
<Declaration>
  <Class IRI="#SixthGrade"/>
</Declaration>
<Declaration>
  <Class IRI="#SocialSciences"/>
</Declaration>
<Declaration>
  <Class IRI="#Subject"/>
</Declaration>
<Declaration>
  <Class IRI="#Teacher"/>
</Declaration>
```

```
<Declaration>
  <Class IRI="#TeachingaSubject"/>
</Declaration>
<Declaration>
  <Class IRI="#Technology"/>
</Declaration>
<Declaration>
  <Class IRI="#TenthGrade"/>
</Declaration>
<Declaration>
  <Class IRI="#ThirdGrade"/>
</Declaration>
<Declaration>
  <Class IRI="#TwelveGrade"/>
</Declaration>
<Declaration>
  <Class IRI="#TwoYearDiploma"/>
</Declaration>
<Declaration>
  <Class IRI="#University"/>
</Declaration>
<Declaration>
  <Class IRI="#ValuePartition"/>
</Declaration>
```

```
<Declaration>
  <ObjectProperty IRI="#hascertificate"/>
</Declaration>
<Declaration>
  <ObjectProperty IRI="#hasminexperience"/>
</Declaration>
<Declaration>
  <ObjectProperty IRI="#haspromotionvalue"/>
</Declaration>
<Declaration>
  <ObjectProperty IRI="#specialistin"/>
</Declaration>
<Declaration>
  <ObjectProperty IRI="#teachesgrades"/>
</Declaration>
<Declaration>
  <ObjectProperty IRI="#teacheslevels"/>
</Declaration>
<Declaration>
  <ObjectProperty IRI="#teachessubjects"/>
</Declaration>
<Declaration>
  <DataProperty IRI="#hasage"/>
</Declaration>
```

```
<Declaration>
  <DataProperty IRI="#hasname"/>
</Declaration>
<Declaration>
  <DataProperty IRI="#hasyearsofexperience"/>
</Declaration>
<Declaration>
  <NamedIndividual IRI="#Ahmad"/>
</Declaration>
<Declaration>
  <NamedIndividual IRI="#AlNajah"/>
</Declaration>
<Declaration>
  <NamedIndividual IRI="#AlQuds"/>
</Declaration>
<Declaration>
  <NamedIndividual IRI="#Birzeit"/>
</Declaration>
<Declaration>
  <NamedIndividual IRI="#OpenAlQuds"/>
</Declaration>
<EquivalentClasses>
  <Class IRI="#AfterSchoolTeacher"/>
  <ObjectIntersectionOf>
```

```

<Class IRI="#Teacher"/>
<ObjectSomeValuesFrom>
  <ObjectProperty IRI="#hascertificate"/>
  <Class IRI="#DiplomainTeacherEducation"/>
</ObjectSomeValuesFrom>
</ObjectIntersectionOf>
</EquivalentClasses>
<EquivalentClasses>
  <Class IRI="#ExpertTeacher"/>
  <ObjectIntersectionOf>
    <Class IRI="#PromotionValue"/>
    <DataSomeValuesFrom>
      <DataProperty IRI="#hasyearsofexperience"/>
      <DatatypeRestriction>
        <Datatype abbreviatedIRI="xsd:integer"/>
        <FacetRestriction facet="&xsd;minInclusive">
          <Literal datatypeIRI="&xsd:string">5</Literal>
        </FacetRestriction>
      </DatatypeRestriction>
    </DataSomeValuesFrom>
  </ObjectIntersectionOf>
</EquivalentClasses>
<EquivalentClasses>
  <Class IRI="#HighBasicLevelTeacher"/>

```

```

<ObjectIntersectionOf>
  <Class IRI="#Teacher"/>
  <ObjectAllValuesFrom>
    <ObjectProperty IRI="#teachesgrades"/>
    <ObjectIntersectionOf>
      <Class IRI="#EighthGrade"/>
      <Class IRI="#FifthGrade"/>
      <Class IRI="#NinthGrade"/>
      <Class IRI="#SeventhGrade"/>
      <Class IRI="#SixthGrade"/>
      <Class IRI="#TenthGrade"/>
    </ObjectIntersectionOf>
  </ObjectAllValuesFrom>
</ObjectIntersectionOf>
</EquivalentClasses>
<EquivalentClasses>
  <Class IRI="#HighBasicLevelTeacherProgramB"/>
  <ObjectIntersectionOf>
    <Class IRI="#HighBasicLevelTeacher"/>
    <ObjectSomeValuesFrom>
      <ObjectProperty IRI="#hascertificate"/>
      <Class IRI="#DiplomainEducationHighBasicLevel"/>
    </ObjectSomeValuesFrom>
  <ObjectSomeValuesFrom>

```

```

    <ObjectProperty IRI="#hascertificate"/>
    <ObjectUnionOf>
        <Class IRI="#BAinArts"/>
        <Class IRI="#BAinSciences"/>
    </ObjectUnionOf>
    </ObjectSomeValuesFrom>
</ObjectIntersectionOf>
</EquivalentClasses>
<EquivalentClasses>
    <Class IRI="#HighBasiclevelTeacherProgramA"/>
    <ObjectIntersectionOf>
        <Class IRI="#HighBasicLevelTeacher"/>
        <ObjectSomeValuesFrom>
            <ObjectProperty IRI="#hascertificate"/>
            <ObjectUnionOf>
                <Class IRI="#BAinEducationwithMajorinTeachingHighBasicLevel"/>
                <Class IRI="#TeachingaSubject"/>
            </ObjectUnionOf>
        </ObjectSomeValuesFrom>
    </ObjectIntersectionOf>
</EquivalentClasses>
<EquivalentClasses>
    <Class IRI="#KindergardenLevel"/>
    <ObjectIntersectionOf>

```

```

<Class IRI="#Level"/>
<ObjectAllValuesFrom>
  <ObjectProperty IRI="#teachesgrades"/>
  <Class IRI="#KindergardenGrade"/>
</ObjectAllValuesFrom>
</ObjectIntersectionOf>
</EquivalentClasses>
<EquivalentClasses>
  <Class IRI="#KindergardenTeacher"/>
  <ObjectIntersectionOf>
    <Class IRI="#Teacher"/>
    <ObjectSomeValuesFrom>
      <ObjectProperty IRI="#hascertificate"/>
    <ObjectUnionOf>
      <Class IRI="#BAinEducationwithMajorinPreschoolEducation"/>
      <Class IRI="#TwoYearDiploma"/>
    </ObjectUnionOf>
  </ObjectSomeValuesFrom>
</ObjectIntersectionOf>
</EquivalentClasses>
<EquivalentClasses>
  <Class IRI="#KindergardenTeacher"/>
  <ObjectExactCardinality cardinality="1">
    <ObjectProperty IRI="#teachesgrades"/>

```

```

    <Class IRI="#KindergardenGrade"/>
  </ObjectExactCardinality>
</EquivalentClasses>
<EquivalentClasses>
  <Class IRI="#Level1-4"/>
  <ObjectIntersectionOf>
    <Class IRI="#Level"/>
    <ObjectAllValuesFrom>
      <ObjectProperty IRI="#teachesgrades"/>
      <ObjectIntersectionOf>
        <Class IRI="#FirstGrade"/>
        <Class IRI="#ForthGrade"/>
        <Class IRI="#SecondGrade"/>
        <Class IRI="#ThirdGrade"/>
      </ObjectIntersectionOf>
    </ObjectAllValuesFrom>
  </ObjectIntersectionOf>
</EquivalentClasses>
<EquivalentClasses>
  <Class IRI="#LowBasicLevelTeacher"/>
  <ObjectIntersectionOf>
    <Class IRI="#Teacher"/>
    <ObjectSomeValuesFrom>
      <ObjectProperty IRI="#hascertificate"/>

```

```

    <Class IRI="#BAinEducationwithMajorinTeachingLowBasicLevel"/>
  </ObjectSomeValuesFrom>
</ObjectIntersectionOf>
</EquivalentClasses>
<EquivalentClasses>
  <Class IRI="#LowBasicLevelTeacher"/>
  <ObjectAllValuesFrom>
    <ObjectProperty IRI="#teachesgrades"/>
    <ObjectIntersectionOf>
      <Class IRI="#FirstGrade"/>
      <Class IRI="#ForthGrade"/>
      <Class IRI="#SecondGrade"/>
      <Class IRI="#ThirdGrade"/>
    </ObjectIntersectionOf>
  </ObjectAllValuesFrom>
</EquivalentClasses>
<EquivalentClasses>
  <Class IRI="#NoviceTeacher"/>
  <ObjectIntersectionOf>
    <Class IRI="#PromotionValue"/>
    <DataSomeValuesFrom>
      <DataProperty IRI="#hasyearsofexperience"/>
      <DatatypeRestriction>
        <Datatype abbreviatedIRI="xsd:integer"/>

```

```

    <FacetRestriction facet="&xsd:maxExclusive">
      <Literal datatypeIRI="&xsd:string">5</Literal>
    </FacetRestriction>
  </DatatypeRestriction>
</DataSomeValuesFrom>
</ObjectIntersectionOf>
</EquivalentClasses>
<EquivalentClasses>
  <Class IRI="#SecondaryTeacher"/>
  <ObjectIntersectionOf>
    <Class IRI="#Teacher"/>
    <ObjectSomeValuesFrom>
      <ObjectProperty IRI="#hascertificate"/>
      <Class IRI="#DiplomainEducationSecondaryLevel"/>
    </ObjectSomeValuesFrom>
    <ObjectSomeValuesFrom>
      <ObjectProperty IRI="#hascertificate"/>
      <ObjectUnionOf>
        <Class IRI="#BAinArts"/>
        <Class IRI="#BAinSciences"/>
      </ObjectUnionOf>
    </ObjectSomeValuesFrom>
  </ObjectIntersectionOf>
</EquivalentClasses>

```

```
<SubClassOf>
  <Class IRI="#Arabic"/>
  <Class IRI="#Subject"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#BAinArts"/>
  <Class IRI="#Certificate"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#BAinEducationwithMajorinPreschoolEducation"/>
  <Class IRI="#Certificate"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#BAinEducationwithMajorinTeachingHighBasicLevel"/>
  <Class IRI="#Certificate"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#BAinEducationwithMajorinTeachingLowBasicLevel"/>
  <Class IRI="#Certificate"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#BAinSciences"/>
  <Class IRI="#Certificate"/>
</SubClassOf>
```

```
<SubClassOf>
  <Class IRI="#Certificate"/>
  <Class abbreviatedIRI=":Thing"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#Chemistry"/>
  <Class IRI="#Subject"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#ChristianEducation"/>
  <Class IRI="#Subject"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#DiplomainEducationHighBasicLevel"/>
  <Class IRI="#Certificate"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#DiplomainEducationSecondaryLevel"/>
  <Class IRI="#Certificate"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#DiplomainTeacherEducation"/>
  <Class IRI="#Certificate"/>
</SubClassOf>
```

```
<SubClassOf>
  <Class IRI="#EighthGrade"/>
  <Class IRI="#Grade"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#EleventhGrade"/>
  <Class IRI="#Grade"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#EnglishLanguage"/>
  <Class IRI="#Subject"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#FifthGrade"/>
  <Class IRI="#Grade"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#FirstGrade"/>
  <Class IRI="#Grade"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#ForthGrade"/>
  <Class IRI="#Grade"/>
</SubClassOf>
```

```
<SubClassOf>
  <Class IRI="#FrenchLanguage"/>
  <Class IRI="#Subject"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#Geography"/>
  <Class IRI="#Subject"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#Geology"/>
  <Class IRI="#Subject"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#History"/>
  <Class IRI="#Subject"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#IslamicEducation"/>
  <Class IRI="#Subject"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#KindergardenGrade"/>
  <Class IRI="#Grade"/>
</SubClassOf>
```

```
<SubClassOf>
  <Class IRI="#Level1-10"/>
  <Class IRI="#Level"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#Level1-12"/>
  <Class IRI="#Level"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#Level11-12"/>
  <Class IRI="#Level"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#Level11-12"/>
  <ObjectAllValuesFrom>
    <ObjectProperty IRI="#teachesgrades"/>
    <ObjectUnionOf>
      <Class IRI="#EleventhGrade"/>
      <Class IRI="#TwelveGrade"/>
    </ObjectUnionOf>
  </ObjectAllValuesFrom>
</SubClassOf>
<SubClassOf>
  <Class IRI="#Level5-10"/>
```

```
<Class IRI="#Level"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#Level5-12"/>
  <Class IRI="#Level"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#Level8-12"/>
  <Class IRI="#Level"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#Math"/>
  <Class IRI="#Subject"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#NinthGrade"/>
  <Class IRI="#Grade"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#Physics"/>
  <Class IRI="#Subject"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#PromotionValue"/>
```

```
<Class IRI="#ValuePartition"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#PromotionValue"/>
  <ObjectUnionOf>
    <Class IRI="#ExpertTeacher"/>
    <Class IRI="#NoviceTeacher"/>
  </ObjectUnionOf>
</SubClassOf>
<SubClassOf>
  <Class IRI="#Science"/>
  <Class IRI="#Subject"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#SecondGrade"/>
  <Class IRI="#Grade"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#SecondaryTeacher"/>
  <ObjectAllValuesFrom>
    <ObjectProperty IRI="#teachesgrades"/>
  <ObjectIntersectionOf>
    <Class IRI="#EleventhGrade"/>
    <Class IRI="#TwelveGrade"/>
  </ObjectIntersectionOf>
</SubClassOf>
```

```
</ObjectIntersectionOf>
</ObjectAllValuesFrom>
</SubClassOf>
<SubClassOf>
  <Class IRI="#SeventhGrade"/>
  <Class IRI="#Grade"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#SixthGrade"/>
  <Class IRI="#Grade"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#SocialSciences"/>
  <Class IRI="#Subject"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#Teacher"/>
  <ObjectSomeValuesFrom>
    <ObjectProperty IRI="#hascertificate"/>
    <Class IRI="#Certificate"/>
  </ObjectSomeValuesFrom>
</SubClassOf>
<SubClassOf>
  <Class IRI="#Teacher"/>
```

```
<ObjectSomeValuesFrom>
  <ObjectProperty IRI="#teachesgrades"/>
  <Class IRI="#Grade"/>
</ObjectSomeValuesFrom>
</SubClassOf>
<SubClassOf>
  <Class IRI="#Teacher"/>
  <ObjectSomeValuesFrom>
    <ObjectProperty IRI="#teachessubjects"/>
    <Class IRI="#Subject"/>
  </ObjectSomeValuesFrom>
</SubClassOf>
<SubClassOf>
  <Class IRI="#Teacher"/>
  <ObjectAllValuesFrom>
    <ObjectProperty IRI="#specialistin"/>
    <Class IRI="#Subject"/>
  </ObjectAllValuesFrom>
</SubClassOf>
<SubClassOf>
  <Class IRI="#Teacher"/>
  <ObjectAllValuesFrom>
    <ObjectProperty IRI="#teacheslevels"/>
    <Class IRI="#Level"/>
  </ObjectAllValuesFrom>
</SubClassOf>
```

```

</ObjectAllValuesFrom>

</SubClassOf>

<SubClassOf>

  <Class IRI="#Teacher"/>

  <DataSomeValuesFrom>

    <DataProperty IRI="#hasage"/>

    <Datatype abbreviatedIRI="xsd:integer"/>

  </DataSomeValuesFrom>

</SubClassOf>

<SubClassOf>

  <Class IRI="#Teacher"/>

  <DataSomeValuesFrom>

    <DataProperty IRI="#hasname"/>

    <Datatype abbreviatedIRI="xsd:string"/>

  </DataSomeValuesFrom>

</SubClassOf>

<SubClassOf>

  <Class IRI="#TeachingaSubject"/>

  <Class IRI="#Certificate"/>

</SubClassOf>

<SubClassOf>

  <Class IRI="#Technology"/>

  <Class IRI="#Subject"/>

</SubClassOf>

```

```
<SubClassOf>
  <Class IRI="#TenthGrade"/>
  <Class IRI="#Grade"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#ThirdGrade"/>
  <Class IRI="#Grade"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#TwelveGrade"/>
  <Class IRI="#Grade"/>
</SubClassOf>
<SubClassOf>
  <Class IRI="#TwoYearDiploma"/>
  <Class IRI="#Certificate"/>
</SubClassOf>
<DisjointClasses>
  <Class IRI="#Certificate"/>
  <Class IRI="#Grade"/>
</DisjointClasses>
<DisjointClasses>
  <Class IRI="#Certificate"/>
  <Class IRI="#Level"/>
</DisjointClasses>
```

```
<DisjointClasses>
  <Class IRI="#Certificate"/>
  <Class IRI="#Subject"/>
</DisjointClasses>
<DisjointClasses>
  <Class IRI="#Certificate"/>
  <Class IRI="#Teacher"/>
</DisjointClasses>
<DisjointClasses>
  <Class IRI="#Certificate"/>
  <Class IRI="#University"/>
</DisjointClasses>
<DisjointClasses>
  <Class IRI="#ExpertTeacher"/>
  <Class IRI="#NoviceTeacher"/>
</DisjointClasses>
<DisjointClasses>
  <Class IRI="#Grade"/>
  <Class IRI="#Level"/>
</DisjointClasses>
<DisjointClasses>
  <Class IRI="#Grade"/>
  <Class IRI="#Subject"/>
</DisjointClasses>
```

```
<DisjointClasses>
  <Class IRI="#Grade"/>
  <Class IRI="#Teacher"/>
</DisjointClasses>
<DisjointClasses>
  <Class IRI="#Grade"/>
  <Class IRI="#University"/>
</DisjointClasses>
<DisjointClasses>
  <Class IRI="#Level"/>
  <Class IRI="#Subject"/>
</DisjointClasses>
<DisjointClasses>
  <Class IRI="#Level"/>
  <Class IRI="#Teacher"/>
</DisjointClasses>
<DisjointClasses>
  <Class IRI="#Level"/>
  <Class IRI="#University"/>
</DisjointClasses>
<DisjointClasses>
  <Class IRI="#Subject"/>
  <Class IRI="#Teacher"/>
</DisjointClasses>
```

```

<DisjointClasses>
  <Class IRI="#Subject"/>
  <Class IRI="#University"/>
</DisjointClasses>
<DisjointClasses>
  <Class IRI="#Teacher"/>
  <Class IRI="#ValuePartition"/>
</DisjointClasses>
<DisjointClasses>
  <Class IRI="#University"/>
  <Class IRI="#ValuePartition"/>
</DisjointClasses>
<ClassAssertion>
  <Class IRI="#Teacher"/>
  <NamedIndividual IRI="#Ahmad"/>
</ClassAssertion>
<ClassAssertion>
  <Class IRI="#University"/>
  <NamedIndividual IRI="#AlNajah"/>
</ClassAssertion>
<ClassAssertion>
  <Class IRI="#University"/>
  <NamedIndividual IRI="#AlQuds"/>
</ClassAssertion>

```

```

<ClassAssertion>
  <Class IRI="#University"/>
  <NamedIndividual IRI="#Birzeit"/>
</ClassAssertion>
<ClassAssertion>
  <Class IRI="#University"/>
  <NamedIndividual IRI="#OpenAIQuds"/>
</ClassAssertion>
<DataPropertyAssertion>
  <DataProperty IRI="#hasage"/>
  <NamedIndividual IRI="#Ahmad"/>
  <Literal datatypeIRI="&xsd;integer">25</Literal>
</DataPropertyAssertion>
<DataPropertyAssertion>
  <DataProperty IRI="#hasname"/>
  <NamedIndividual IRI="#Ahmad"/>
  <Literal datatypeIRI="&xsd;string">Ahmad Ali Othman</Literal>
</DataPropertyAssertion>
<SubObjectPropertyOf>
  <ObjectProperty IRI="#hascertificate"/>
  <ObjectProperty abbreviatedIRI=":topObjectProperty"/>
</SubObjectPropertyOf>
<SubObjectPropertyOf>
  <ObjectProperty IRI="#hasminexperience"/>

```

```

    <ObjectProperty abbreviatedIRI=":topObjectProperty"/>
  </SubObjectPropertyOf>
  <SubObjectPropertyOf>
    <ObjectProperty IRI="#haspromotionvalue"/>
    <ObjectProperty abbreviatedIRI=":topObjectProperty"/>
  </SubObjectPropertyOf>
  <SubObjectPropertyOf>
    <ObjectProperty IRI="#specialistin"/>
    <ObjectProperty abbreviatedIRI=":topObjectProperty"/>
  </SubObjectPropertyOf>
  <SubObjectPropertyOf>
    <ObjectProperty IRI="#teachesgrades"/>
    <ObjectProperty abbreviatedIRI=":topObjectProperty"/>
  </SubObjectPropertyOf>
  <SubObjectPropertyOf>
    <ObjectProperty IRI="#teacheslevels"/>
    <ObjectProperty abbreviatedIRI=":topObjectProperty"/>
  </SubObjectPropertyOf>
  <SubObjectPropertyOf>
    <ObjectProperty IRI="#teachessubjects"/>
    <ObjectProperty abbreviatedIRI=":topObjectProperty"/>
  </SubObjectPropertyOf>
  <FunctionalObjectProperty>
    <ObjectProperty IRI="#haspromotionvalue"/>

```

```

</FunctionalObjectProperty>
<ObjectPropertyDomain>
  <ObjectProperty IRI="#hascertificate"/>
  <Class IRI="#Teacher"/>
</ObjectPropertyDomain>
<ObjectPropertyDomain>
  <ObjectProperty IRI="#haspromotionvalue"/>
  <Class IRI="#PromotionValue"/>
</ObjectPropertyDomain>
<ObjectPropertyDomain>
  <ObjectProperty IRI="#specialistin"/>
  <Class IRI="#Teacher"/>
</ObjectPropertyDomain>
<ObjectPropertyDomain>
  <ObjectProperty IRI="#teachesgrades"/>
  <Class IRI="#Teacher"/>
</ObjectPropertyDomain>
<ObjectPropertyDomain>
  <ObjectProperty IRI="#teacheslevels"/>
  <Class IRI="#Teacher"/>
</ObjectPropertyDomain>
<ObjectPropertyDomain>
  <ObjectProperty IRI="#teachessubjects"/>
  <Class IRI="#Teacher"/>

```

```
</ObjectPropertyDomain>
<ObjectPropertyRange>
  <ObjectProperty IRI="#hascertificate"/>
  <Class IRI="#Certificate"/>
</ObjectPropertyRange>
<ObjectPropertyRange>
  <ObjectProperty IRI="#haspromotionvalue"/>
  <Class IRI="#ExpertTeacher"/>
</ObjectPropertyRange>
<ObjectPropertyRange>
  <ObjectProperty IRI="#haspromotionvalue"/>
  <Class IRI="#NoviceTeacher"/>
</ObjectPropertyRange>
<ObjectPropertyRange>
  <ObjectProperty IRI="#specialistin"/>
  <Class IRI="#Subject"/>
</ObjectPropertyRange>
<ObjectPropertyRange>
  <ObjectProperty IRI="#teachesgrades"/>
  <Class IRI="#Grade"/>
</ObjectPropertyRange>
<ObjectPropertyRange>
  <ObjectProperty IRI="#teacheslevels"/>
  <Class IRI="#Level"/>
```

```
</ObjectPropertyRange>
<ObjectPropertyRange>
  <ObjectProperty IRI="#teachessubjects"/>
  <Class IRI="#Subject"/>
</ObjectPropertyRange>
<SubDataPropertyOf>
  <DataProperty IRI="#hasyearsofexperience"/>
  <DataProperty abbreviatedIRI=":topDataProperty"/>
</SubDataPropertyOf>
<FunctionalDataProperty>
  <DataProperty IRI="#hasage"/>
</FunctionalDataProperty>
<FunctionalDataProperty>
  <DataProperty IRI="#hasname"/>
</FunctionalDataProperty>
<FunctionalDataProperty>
  <DataProperty IRI="#hasyearsofexperience"/>
</FunctionalDataProperty>
</Ontology>
<!-- Generated by the OWL API (version 3.1.0.1602) http://owlapi.sourceforge.net -->
```