AN IMPROVED SEMANTIC WEB-BASED SEARCH MODEL FOR E-COMMERCE USING OWL2

BY

PAM, BULUS DUNG

DEPARTMENT OF COMPUTER SCIENCE

FACULTY OF PHYSICAL SCIENCES

AHMADU BELLO UNIVERSITY, ZARIA

NIGERIA

MAY, 2017

AN IMPROVED SEMANTIC WEB-BASED SEARCH MODEL FOR E-COMMERCE USING OWL2

BY

PAM, BULUS DUNG

B.Sc. (ATBU, Bauchi, 2000)

P16PSCS8107

A DESEERTATION SUBMITTED TO THE SCHOOL OF POSTGRADUATE STUDIES

AHMADU BELLO UNIVERSITY (ABU), ZARIA

IN PARTIAL FULLFILMENT OF THE REQUIREMENTS FOR THE AWARD OF A

MASTER OF SCIENCE (M.Sc.) DEGREE IN COMPUTER SCIENCE

DEPARTMENT OF COMPUTER SCIENCE

AHMADU BELLO UNIVERSITY (ABU), ZARIA

MAY,2017

DECLARATION

I declare that the work in this dissertation titled "AN IMPROVED SEMANTIC WEB-BASED SEARCH MODEL FOR E-COMMERCE USING OWL2" has been written by me in the Department of Computer Science under the supervision of Prof. A.A. Obiniyi and Dr. S.E Abdullah.

The information contained in the literature has been duly acknowledged in the text and a list of references provided. No part of this dissertation was previously presented for another degree or diploma at any university.

Pam Bulus Dung

…… ……

Name of Student Signature/Date

CERTIFICATION

This dissertation titled "AN IMPROVED SEMANTIC WEB-BASED SEARCH MODEL FOR E-COMMERCE USING OWL2" by Pam Bulus Dung (M.Sc./Sci./1562/2011-2012) meets the regulations governing the award of the degree of M.Sc. Computer Science of Ahmadu Bello University (ABU), Zaria, and it is approved for its contribution to knowledge and literary presentation.

Prof. A. A. Obiniyi

……. …… ……

Chairman, Supervisory Committee Signature Date

Dr. S.E. Abdullahi

……. …… ……

Member, Supervisory Committee Signature Date

……. …… ……

External Examiner Signature Date

Prof. S.B Junaidu

……. …… ……

Head of Department Signature Date

Prof. Abba Ali Tijjani

……. …… ……

Dean Faculty of Physical Sciences Signature Date

Prof. Kabir Bala

……. …… ……

Dean School of Post Graduate Studies Signature Date

DEDICATION

This dissertation is dedicated to the glory of God for his perfect assurance that my work will be rewarded in ABU Zaria.

ACKNOWLEDGEMENT

Foremost, I would like to express my sincere gratitude to God and to my Major and Minor supervisors. Prof. A.A Obiniyi and Dr. S.E Abdullahi for the continuous support they rendered me in the course of the work, their patience, motivation, and enthusiasm kept me going. My sincere thanks also goes to my wife, Mrs Kangyang Pam Bulus and Children, Nta Comforted Bulus, Netsitseng Judah Bulus and Nken'rat Victory Bulus for their continuous love . My appreciation goes to my biological and extended family, also to my family friends Dr. Hosea Finangwai and Family, Mrs. Beatrice Nenpan Dikko , with her lovely kids Petnan Nenpan Dikko and Primpinan Nenpan Dikko, Engineer Joseph Dalyop and family, Mr. John Chaka and family, Dr. Ishaya Longdet and family, Dr. Udoh and family and Mrs. Dada and family who stood by me especially in prayers during the course of this study. I thank my fellow course mates Mr Oyelade, Olaide Nathaniel, Mr. Francis Nyeribe, Mr. Ishaya Nubunga and Mr. Bimbo, they were mates of honour indeed. Appreciation goes to my employers in FCE Pankshin, my Dean, and colleagues in the Department for their support and understanding. Finally, I would like to thank all my Spiritual fathers and friends in COCIN Vel, FCS, CAPRO, Gospel Truth, and Truth Embassy for standing by me and praying for me at all times.

ABSTRACT

Information sharing across independent web applications seems impossible with just the static world wide web applications. Meanwhile, with the advent of the semantic web, web applications interoperability through information sharing and Linked Data have been achieved. This interoperability is usually presented as a decentralized system that allows agents to crawl over them for information sharing or gathering, and is employed in developing web applications like E-Commerce. However, the existinge-Commerce model is being limited by its knowledge representation language. This model employs ontological approach using OWL for knowledge modeling and representation. Hence, this research seeks to create an improved e-commerce ontological base model which has a more rich and expressive knowledge base with OWL2 for knowledge modeling and representation.This knowledgebase will be compared against the e-Commerce knowledgebase that was modeled with OWL. Search results from the new knowledge base by the user yielded a more specific and needful outcome as compared to search on the knowledge base created with OWL.

TABLE OF CONTENTS

TITLEPAGE……..……...……...…..……..i
DECLARATION iv
CERTIFICATION v
DEDICATION vi
ACKNOWLEDGEMENT vii
ABSTRACT viii
TABLE OF CONTENTS ix
LIST OF FIGURES xi
LIST OF TABLES xii
ABREVIATIONS xiii
CHAPTER ONE 1
INTRODUCTION 1
1.1 Background to the Study 1
1.2 Statement of Research Problem 5
1.3 Motivation for the Dissertation. 6
1.4 Aim and Objectives of the Dissertation 6
1.5 Research Methodology 6
1.6 Dissertation Organization 7
CHAPTER TWO 9
LITERATURE REVIEW 9
2.1 Introduction to Semantic Web 9
2.2 The Vision of the Semantic Web 10
2.3 Data Storage in Semantic Web 11
2.4 Ontology 13
2.5 Technologies Stack in Semantic Web 15
2.5.1 RDF 15
2.5.2 RDFS 16
2.5.3 OWL 18
2.5.4 OWL2 20
2.6. OWL 2 Web Ontology Language New Features and Rationale 21
2.6.1 Syntactic sugar 21
2.6.2 New constructs for properties 22
2.6.3 Extended datatype capabilities. 25
2.6.4 Minor features 26
2.7 Query Languages in Semantic Web 27
2.7.1 SPARQL 27

2.7.2 Semantic web rule language (SWRL) 29
2.7.3 Semantic query-enhanced web rule language (SQWRL) 30
2.8 Intelligent Agents on Semantic Web 31
2.9 Rule Composition for the Ecommerce System 33
2.10 Tools, Reasoner and Inference Engine 33
2.11 Related Works 36
CHAPTER THREE 44
ARCHITECTURE AND ONTOLOGY MODEL OF THE E-COMMERCE SYSTEM 44
3.1 Introduction 44
3.2 Overview of the Proposed System 44
3.2.2 Components of the System 45
3.3 Ecommerce Ontology Development 47
3.3.1 The Core Ontology and Retailer's Registry 47
3.3.2 Distributed Retailer Knowledgebase 48
3.3.3 Algorithm of the proposed system 49
CHAPTER FOUR 50
THE IMPLEMENTATION OF THE COMPARISON MODEL AND E-COMMERCE APPLICATION 50
4.1 Introduction 50
4.2 A Comparison Model for Obiniyi *et al*, (2014) and the New System 50
4.2.1 Comparing the Two Models. 52
4.3 The E-Commerce Application. 55
4.4 Results and Discussion 57
Table 4.1: Generated queries and their descriptions 57
Table 4.2: Generated queries and their performances on two differently modeled ontologies 58
CHAPTER FIVE: 60
SUMMARY, CONCLUSION AND RECOMMENDATION 60
5.1 Summary 60
5.2 Conclusion 61
5.3 Recommendations 62
REFERENCES 63
APPENDIXES A: Ontology File 66

LIST OF FIGURES

Figure 2.3 Graphic Representation of a Triple (Antoniou and Frank, 2004) 16
Figure 2.4: RDF Representation of a Statement (Antoniou and Frank, 2004) 16
Figure 2.5: Graphical Representation of Classes and Properties in Hierarchy (Antoniou and Frank, 2004) 17
Figure 2.6: RDFS Illustration of Classes and Properties(Antoniou and Frank, 2004) 18
Figure 2.7: A Typical OWL File(Antoniou and Frank, 2004) 19
Figure 2.9: SPARQL Graph Pattern(Obiniyi *et al*, 2014) 28
Figure 2.10: SPARQL with restrictions(Obiniyi *et al*, 2014) 28
Figure 2.11: Agents in Semantic Web(Obiniyi *et al*, 2014) 32
Figure 2.12 Architecture of the Product IR Framework. (Zhang *et al.*, 2009) 37
Figure2.13: The Block Diagram Showing The Downloading unstructured text using 38
online search *(Jayaprabha and Saradha, 2010).* 38
Figure 2.14: The Design Methodology of Customer Process Variant (Yao *et al.*, 2012) 39
Figure 3.1: Architecture of the proposed Ecommerce System 45
Figure 3.2: Core Ontology – Retailer's Registry 48
Figure 3.3: Retailer's Ontology 49
Figure 4.1: Main.java 51
Figure 4.2 A Comparison Interface for Obiniyi *et al,* (2014) and the New System 52
Figure 4.3a: Ontology Constitutuents Selection for the Two Ontologies 53
Figure 4.3b: Ontology loading interface 53
Figure 4.4a: Variable and Query Entry Interface 53
Figure 4.4c: Jess Rule Engine Style of Query with SWRL 54
Figure 4.5 Results of the Queries Displayed 55
Figure 4.6 The E-Commerce Graphical User Interface. 55
Figure 4.7a The server side querying the user needs on retailer ontology. 56
Figure 4.7b The client side sending user query to server side. 56
Figure 4.8 Entering Server URL and Search Word 57
Figure 4.9 Result of Product Search on the E-commerce Application. 57

LIST OF TABLES

Table 4.1: Generated queries and their descriptions……..70

Table 4.2: Generated queries and their performances on two differently modeled ontologies……71

ABREVIATIONS

XML:	eXtensible Markup Language
DTD:	Document Type Definition
SPARQL:	SPARQL Protocol and RDF Query Language
SQWRL:	Semantic Web-enhanced Query Language
RDF:	Resource Description Framework
RDFS:	Resource Description Framework Schema
W3C:	World Wide Web Consortium
API:	Application Programming Interface
SWRL:	Semantic Web Rule Language
DL:	Defeasible Logic
RuleML:	Rule Markup Language
B2B:	Business to Business
B2C:	Business to Consumer
JADE:	Java Agent Development Environment
HTML:	Hyper Text Markup Language
XSLT:	Extensible Stylesheet Language Transformations
XPATH:	XML Path Language
IDE:	Integrated Development Environment
URI:	Universal Resource Identifier
URLs:	Universal Resource Locators
DARPA:	Defense Advanced Research Projects Agency
DAML+OIL:	DARPA Markup Language and Ontology Inference Layer
OWL:	Web Ontology Language

CHAPTER ONE

INTRODUCTION

1.1 Background to the Study

The history of bartering dates all the way back to 6000 BC, introduced by Mesopotamia tribes, bartering was adopted by Phoenicians. Phoenicians bartered goods to those located in various cities across the oceans.(BSH, 2016). That time, the exchange of goods or services was on the grounds of giving out a product or service so as to own another product, good or service as may be the case. With time, the idea of having a measurable medium that will be best used for exchange of goods or services came into trading. Commerce which relates to all economic activities resulting in production, exchange, distribution and consumption of commodities and services, dates back to the history of man (Dhanapal *et al.*, 2004). The commercial activities involve a seller (the owner of the product or service meant for exchange) and a buyer (the person who needs the product or service presented for exchange). The buyer bargains for the good or service, when an agreement is reached, then he makes the payment. When the world wide web(WWW) was developed in 1993 by Tim Berners-Lee, the director of world Wide Web Consortium(W3C) (Gruber, 1993), attention was not given to it, but when it became a medium for e-commerce and social networking all over the globe, it was amazingly patronise. The web has the following categories: Web 1.0, Web 2.0, Web 3.0 , 4.0 and Web 5.0. Web 1.0 could be considered as the "read-only web."(WEB1.0,2016). In other words, the early web allowed users to search for information and read it. There was very little in the way of user interaction or content contribution. The first shopping cart applications, which most e-commerce website owners use in some shape or form, basically fall under the category of Web 1.0. The overall goal was to present products to potential customers, much as a catalog or a

brochure does — only through a website retailers could also provide a method for anyone (anywhere in the world) to purchase (their) products. The year 1999 marked the beginning of a Read-Write-Publish era with notable contributions from LiveJournal (Launched in April, 1999) and Blogger (Launched in August, 1999). Now even a non-technical user can actively interact and contribute to the web using different blog platforms. Considering Berners-Lee's method of describing it, the Web 2.0, or the "read-write" web has the ability to contribute content and interact with other web users. This interaction and contribution has dramatically changed the landscape of the web. The Web 2.0 appears to be a welcome response to a web users demand to be more involved in what information is available to them.This era empowered the common user with a few new concepts like Blogs, Social-Media & Video-Streaming. Publishing your content is only a few clicks away! Few remarkable developments of Web 2.0 are Twitter, YouTube, eZineArticles, Flickr and Facebook. Web 3.0 is the "read-write-execute" web. Semantic markup and web services forms it's basis, the context of Web 3.0, take center stage by combining a semantic markup and web services, which promises the potential for applications that can speak to each other directly, and for broader searches for information through simpler interfaces. Web 4.0also known as the "Mobile Web" connects all devices in the real and virtual world in real-time. Web 5.0 is about the (emotional) interaction between humans and computers. The interaction will become a daily habit for a lot of people based on neurotechnology. For the moment web is "emotionally" neutral, which means web does not perceive the users feel and emotions. This will change with web 5.0 – emotional web. One example of this is www.wefeelfine.org, which maps emotions of people. With headphones on, users will interact with content that interacts with their emotions or changes in facial recognition,(WEB1.0, 2016). With the coming of these web advancements, organizations and enterprises are taking the advantage by using them to increase their sales and access the world. Several developments in online trading or e-commerce have emerged as a

result of these advances in web technology. Notable among these is the internet shopping robots, sometimes called shopbots**.** Shopbots is an 'automated tool that permits customers to easily search for prices and product characteristics from different online retailers (Smith, 2002). The growth of this Internet price search tools, notably the shopbots, has reduced consumers' search costs for price and some product characteristics. While a variety of analytic models predict that increased consumer search through shopbots will lower price levels among competing retailers, though there is no consensus in the empirical literature as to whether price dispersion will increase or decrease in response to increased consumer search through shopbots (Tang, *et al.*, 2007). Again, it was shown that market observers predicted benefits consumers stand to enjoy at the expense of retailers. Hence, shopbots will radically reduce consumer search costs and as well reduce retailer's opportunity to differentiate their products. This will in turn lower down the margins of retailers to zero. However, it was suggested that retailers still have several opportunity to differentiate their products, leverage brand names, peg strategic prices, and reduce the effectiveness of consumers search af shopbots (Smith, 2002). The applications do not just stop at listing products, they also enable the prospective consumer to buy the required product and pay for them.Again, the semantic web has technologies such as eXtensible Markup Language (XML), Resource Description Framework (RDF), Resource Description Framework Schema (RDFS), Ontology Web Language (OWL), and Ontology Web Language 2 (OWL2) which gives datasets of specific ontology that can be queried across the net by machines or intelligent agents, as such giving the necessary information to its assessors. Applications developers for e-commerce or electronic markets have been making use of these different patterns of representing datasets that has to do with the information about the goods or services of which they seek to convey and as well sell to their prospective consumers. Among all these technologies, XML was the first on line that enabled developers to organize data around tags that are well formed or well nested based on a

rule written in Document Type Definitions (DTDs) or XML Schema. This feature of the XML marks it out as the main catalyst of e-Business as rightly noted in a dissertation work (Willaert, 2001). Then in order of advancement the RDF is next after XML. Resource Description Framework (RDF) is often seen as a data model in which data is represented in object-attribute-value pattern called a statement. RDF has been given XML syntax and it is domain independent- that is, its applicability covers any real world domain. But users of RDF may choose to define their own terminology by using a schema called RDF Schema (RDFS). RDF/RDFS enabled us to model particular domains such as the products/services a store is willing to put online for sale. Another language for modeling concepts is the Ontology Web Language (OWL).

The semantic web has very great benefits for business, though these benefits have some weaknesses that are defined in the Linked Data Design Noted by Tim Berners-Lee in 2006 (Hart and Dolber, 2013). By making public business data as Linked Data, business organizations do not only make their products or services more findable in an objective fashion, but they provide a platform for other computer systems to read the exposed data. Exposing data as linked data becomes an exciting marketing technique (Lewis, 2008). Agents are now able to follow one node to another across the global giant graph. This change led to the Semantic web being 'revisited' (Shadbolt *et al.*, 2006) and 'rebooted' by Cyganiak in 2008 (Smith, 2002). Some of these agents are termed as web crawlers. These agents can traverse several datasets retrieving data and putting data into a piece of document for a user.

Some possible applications of semantic web and agents could be in the area of a personal agent that uses formal semantic web knowledge bases to book holidays or even doctor's appointment as thought of by Tim Berners-Lee (Antoniou and Frank, 2004) during their creation of semantic web, a multi-agent system that is capable of acting in this own community to build and maintain additional Linked Data sets.

1.2 Statement of Research Problem

The importance of the web has been pointed out in the previous section. Even though the Semantic web has done a lot for web application development, this dissertationhowever argues that such technologies as provided by the semantic web are not yet used to the fullest in the aspect of electronic market or ecommerce, especially in the case of Business to Commerce (B2C) applications.

Using OWL2 to implement ontology, one must choose the querying language carefully. SparqlProtocol and RDF Query Language (SPARQL) is a query language that is used to query ontology developed using RDF/RDFS. So many applications that use OWL/OWL2 to develop their ontology have made use of SPARQL as a querying language to query the OWL/OW2 ontology. But SPARQL as a query language was not actually meant to query OWL/OWL2; hence the ontology implemented in OWL/OWL2 must be serialized in RDF before it can be queried with SPARQL,(Obiniyi *et al*, 2014). Though often times, the semantics of the ontology may reduce/loss in the process of converting it from OWL/OW2 to RDF. This implies that there is bound to be loss of concepts and relations in the process of serializing OWL/OWL2 in RDF. Hence the inefficient or wrong query language on OWL/OWL2 results in an inefficient application specifically an e-commerce application also inefficient product search on e-Commerce applications negatively affects receptivity of the application by users.This usually results from a weakly modeled knowledge base.These knowledge bases are modeled as ontology file in semantic web applications.Various ontology languages can be used in modeling an ontology file. However, representing ontology with OWL limits expressivity, inference making and thereby adversely affecting product search. Therefore this research observes that querying OWL2 with SQWRL yields optimal output than OWL with SQWRL.

1.3 Motivation for the Dissertation Here in this section are the major technological motivation for the proposed solutions to the problems cited in section 1.2

a) OWL has some difficulty in concept expressivity, as such there was need to use a more expressive language: OWL2
b) With an OWL2 knowledge base, product search will be improved.
c) SQWRL and SWRL are primitive to OWL2. They provide a combined query and rule language fitting for OWL2.
d) Using a more primitive query and rule languages against a rightly modeled knowledgebase improves product search,Obiniyi *et al.,(*2014).

1.4 Aim and Objectives of the Dissertation

The aim of this research is to create improve ontological based search model for e-commerce using OWL2. The specific objectives are to:

a) Create a more expressive ontologies using some OWL2 constructs than OWL.
b) To implement the Ontology based Application for B2C as a case study.
c) Compare how the use of SQWRL to query our E-commerce model that was developed using OWL2 will generate more result than using SQWRL to queryObiniyi *et al.*,(2014) E-commerce model that was done using OWL
d) Enhance the efficiency of B2C by implementing the Ontology using OWL2 rather than OWL

1.5 Research Methodology

Below are the methods of achieving the objectives stated in section 1.4

a) Review related works in the use of Semantic web for deploying E-commerce system.

b) In this research, an application for showing the benefit of a knowledge base modeled in OWL2 as against OWL is created for comparison using Java

c) This application will receive two ontologies modeled in OWL2 and OWL.

d) Different queries will be constructed for the ontologies and there outputs will then be used to graphically observe their efficiency. This will be done by running different sample queries on (Obiniyi *et al.*,2014) model and our model and the result will be presented graphically.

e) On the other hand, an E-Commerce application will be created for product search using Java.

f) Two ontologies (core & retailer) will be developed for use in the E-commerce application using Protégé3.8 for OWL1 knowledge base and 4.2 for OWL2 knowledge base

g) Pellet will be used to reason over OWL2 to infer more results while JESS was used on OWL for more inference

1.6 Dissertation Organization

The rest of this dissertation work is structured as follows:

Chapter 2: Gives introductory notes on related researches in the direction that this dissertation concentrates on, and it also goes further to unveil relevant concepts behind the semantic web, its technologies, standards and its query languages –SPARQL and SQWRL. It also moves further to explain the semantic web framework for Java- that is Jena and OWL Application Programme Interface (API).

Chapter 3: Shows the architecture of the solution the dissertation proposes. It shows how to develop an ontology using Protégé and how to communicate with an OWL2-based ontology

using Jena and OWL2 API, and also, it shows how to use SQWRL to query ontology. Also, some OWL2 constructs that make it more expressive than OWL are considered in this chapter. Corresponding queries in SQWRL are also executed against the same ontology so as to demonstrate which one of such is more efficient.

Chapter 4: Here, both the server and client sides captured in the architecture are implemented and connected. Furthermore, the result of the implementation is discussed. Lastly, the Chapter carries out different testing on the developed application.

Chapter 5: In this chapter we summarized the work, drawing out conclusions that let to some recommendations. The limitations of the work were listed out as future works.

CHAPTER TWO

LITERATURE REVIEW

2.1 Introduction to Semantic Web

The Semantic Web is seen as an extension of the currentWorld Wide Web that catalogsinformation on a webpage and reprocesses it so that other machines including computers can understand the information. The semantic web alleviates the tasksrequired by humans. The conversion of the dataprovides a commonframework for multiple systems tounderstand and processthe content.(SW,2016).

The concept of the Semantic Web is the brainchild of the original creator of the World-Wide Web, Tim Berners-Lee. The idea behind the Semantic Web is 'to weave a Web that not only links documents to each other but also recognises the meaning of the information in those documents.(Frauenfelder, 2001), In other words, it can transform the current Web from a series of interconnected, but ultimately semantically isolated data islands into one gigantic, personal information storage, manipulation and retrieval database. According to Berners-Lee, (Hendler and Lissila, 2001), 'most of the Web's contentis designed for humans to read, not for computer programs to manipulate meaningfully'. Computers can adeptly parse Web pages for layout and routine processing – here a header, there a link to another page but, in general, computers have no reliable way to process the Semantics, or the meaning of the content of the page. Tim Berners-Lee sees it as being an extension of the current World-Wide Web that willbring a common structure to the content of Web pages, thereby providing such content withmeaning which will allow external software agents to carry out sophisticated tasks on behalfof the reader or user and, as such, promote a greater degree of cooperation between humansand computers. In so doing, a new age of computing will be ushered in where

machines arebetter able to 'process and "understand" the data that they merely display atpresent' (Berners-Lee *et al*. 2001).

2.2 The Vision of the Semantic Web

The vision of semantic web was stated in a statement as follows:

> "I have a dream for the Web [in which computers] become capable of analyzing all the data on the Web – the content, links, and transactions between people and computers. A 'Semantic Web', which should make this possible, has yet to emerge, but when it does, the day-to-day mechanisms of trade, bureaucracy and our daily lives will be handled by machines talking to machines. The 'intelligent agents' people have touted for ages will finally materialize."(Peter, 2015)

So, semantic web seeks to change the Internet by implementing the following concepts:

a) To open up the web of data to artificial intelligence- Getting the web to do a bit of thinking for users.
b) Encouraging enterprises, companies and organizations to publish their data freely in an open and standard format.
c) Encouraging businesses to utilize already available data on the web- Data give and take.

This actually means that it is taking the data embedded in the web pages of different companies and then harnessing them together as though it were a data of a particular database belonging to an organization, even though, this data includes other data belonging to several organizations, enterprises and industries. Going by the purpose of the semantic web, the concept of datasets now comes into play. Datasets entail a knowledge base usually holding a

very large data which may be related in some form (Freitas *et al.*, 2012). They are mostly used to actualize the concept of Linked Data using the various semantic web technologies (RDF, RDFS, OWL and OWL2). A dataset may contain a link within it that points to another dataset and this could generate a large global dataset that we invariably refer to as Linked Data (Bizer *et al.*, 2009). This Linked Data is very great in size, enabling networks that enhance intelligent agents (machines with reasoning capabilities) to 'crawl' over them for the sake of information retrieval. The vision of a Semantic Web can therefore be viewed from three different perspectives: (Kuck, 2004)

a) A type of universal library which can readily be accessed and used by humans in their day-to-day information acquisition.
b) The backbone for software or computational agents to utilize autonomously in order to perform particular activities on behalf of their human counterparts; and
c) A 'method for federating particular knowledge bases and databases to perform anticipated tasks for humans and their agents.

2.3 Data Storage in Semantic Web

There have seen several ways in which data is modeled and classified, some of the classification include:(Oyelade,*et al*., 2014)

a) Relational Database: in this format of data representation, data is collected together in related entities or relations in the form of entity sets. Each entity is related with another entity, such that entity sets in a particular entity could be seen as relating to the entity sets of another entity. Relationship among data is the focus of this type of data model. Relational databases have been used to model this data format and then persistently stored on Relational Database Management Systems.

b) Hierarchical Data Model: XML, models its data using syntactically correlating open and close tags. It has been used for data exchange between legacy systems and new modern data storage systems. XML is used mostly to store data that could be exported across different platforms. Using XML enables users to store data in a hierarchical format and then thereafter saved on files in the system.

Semantic web represents its data in the form of graph database. Resource Description Framework (RDF) acts as a basic building block that enables users to store semantic data. When using XML to represent data, the data seems to be represented in the form of an order or a data having precedence over another. This is from the concept of putting data round tags or elements nested in a specific order starting from the root element. In semantic web, data representation is done by showing that a resource is only linked or related to another resource without any intrinsic precedence over another resource. The diagram in Figure 2.1 clearly illustrates these differences.

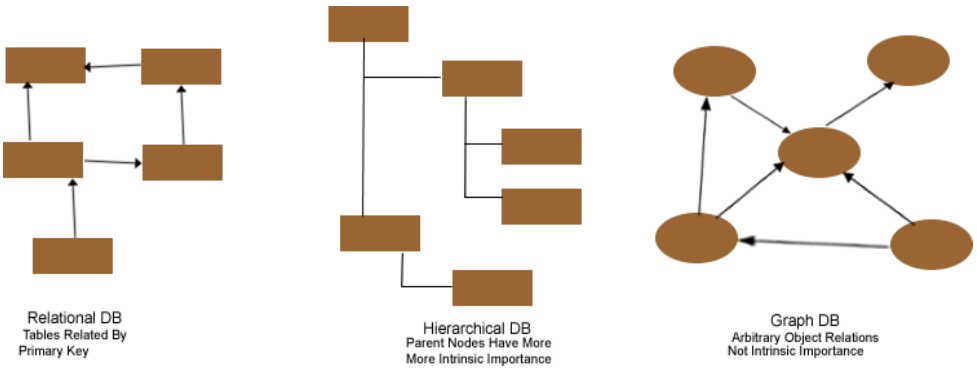


Figure 2.1 Different Data Representations(Obiniyi *et al*,2014*)*

The primary focus of semantic web as envisioned was that it should be able to understand the meaning of data, that is, the semantic web would be able to receive inputs (probably for search) from humans and then be able to retrieve corresponding results from a semantically structured data using RDF or OWL.

The semantic web is sometimes used to indicate the various technologies that it supports. The diagram in Figure 2.2illustrates how these technologies relate with each other. This is well known as the Semantic Web cake.

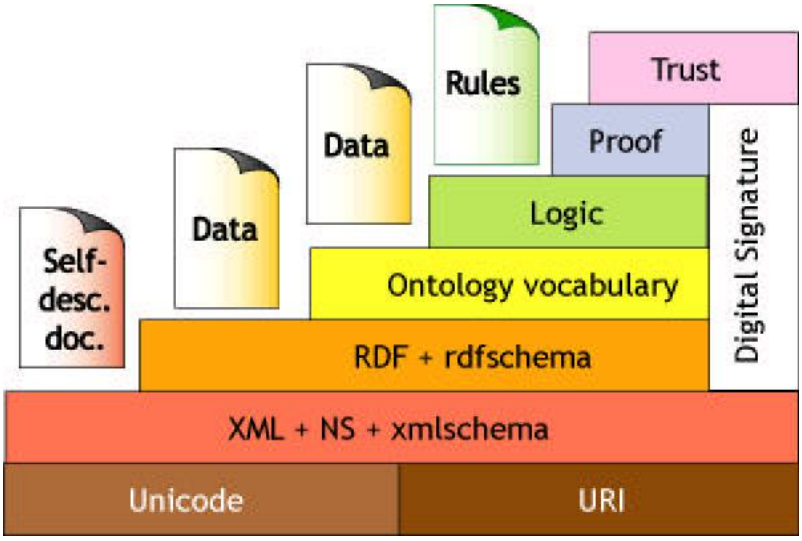


Figure 2.2 Semantic Web Technology Stack. (Ian and Peter, 2003)

2.4 Ontology

Other important components of the Semantic Web are Ontologies. In 1993, Gruber originally defined the notion of an ontology as an explicit specification of a conceptualization. The most typical kind of ontology for the Web has a taxonomy and a set of inference rules. The taxonomy defines classes of objects and relations among them. Classes, subclasses and relations among entities are a very powerful tool for Web use. A large number of relations can be expressed among entities by assigning properties to classes and allowing subclasses to inherit such properties. Inference rules in ontologies supply further power. Reasoning support is important because it allows to check the consistency of ontologies and the knowledge to check for unintended relationships between classes and automatically classify instances in classes. Automated reasoning support allows one to check more cases than could be checked

manually. Checks like the preceding ones are valuable for designing large ontologies, where multiple authors are involved, and for integrating and sharing ontologies from various sources. The computer doesn't truly understand any of this information, but it can now manipulate the terms much more effectively in ways that are useful and meaningful to the human user. With ontology pages on the Web, solutions to terminology (and other) problems begin to emerge. The meaning of terms or XML codes used on a Web page can be defined by pointers from the page to an ontology. Of course, the same problems as before can arise if one person points to an ontology that defines addresses as containing, for example, a zip code and another person points to one that uses a postal code instead. This kind of confusion can be resolved if ontologies (or other Web services) provide equivalent relations: one or both of the ontologies used may contain information that the zip code is equivalent to the postal code. Ontologies can enhance the functioning of the Web in many ways, and can be used in a simple fashion to improve the accuracy of Web searches. The search program can look for only those pages that refer to a precise concept instead of all the ones using ambiguous keywords. More advanced applications will use ontologies to relate the information on a page to the associated knowledge structures and inference rules. Ontologies enable knowledge sharing and reuse. There are many different ontologies already in existence and can be useful to an already existent ontology, depending on the application in the Web, depending on the application it can be useful to use an already existent ontology. Sometimes, however, it is necessary to modify one of those available ontologies or to create a new one, more adapted to the concerned domain.(Pomponio L. and Viale P, 2013).

2.5 Technologies Stack in Semantic Web

This technology stack helps one to be able to describe the ontology process. Figure 2.2 shows the ranges of technologies in this stack. Reading from the bottom of the figure, XML and its schema come first, then RDF and its schema, RDFS. Moving up is the web ontology language (OWL) and SPARQL- a query language- and then the rules specification language. This research shall make use of some of these technologies at this level. This research work shall take few sections to give an overview of the listed technologies in Figure 2.2 from the work of (Oyelade, *et al*, 2014).

2.5.1 RDF

Resource Description Framework (RDF) is a data model that takes up the syntax of XML. RDF may be used to model data just like XML is being used to model data. RDF has more powerful query languages that can enhance inference and it also possesses more advanced features than XML. Information or data is represented in RDF in the form of statements. The component of a statement in RDF could be broken into three. These are:

a) **Resource**: A resource is used to represent a 'thing' or an object to be discussed using a universal resource identifier (URI). This URI could be a universal resource locator (URL) or web address. For example a resource could be written as *http://www.ourproducts.org/hp*.

b) **Properties:** A property is used to show the relationship between resources. They are special kind of resources. They can also be identified by using URL. For example, a property can be represented as *http://www.products_rel.org/product.*

c) **Property Value**: This is the value of the property specified in a statement. This is mostly resources in themselves describing another 'thing' or object. For example, we could have a property value of *http://www.ourproducts.org/prod/laptop*.

A statement is an “Object-property-value” triple. By statement form of data representation in RDF, it means that a statement like *Hp sells laptop computer* could be represented as shown by the graph in Figure 2.3.



Figure 2.3 Graphic Representation of a Triple (Antoniou and Frank, 2004)

From this statement, *http://www.ourproducts.org/hp* is our resource which we are talking about and *http://www.products_rel.org/product* is the property of the statement and *http://www.ourproducts.org/prod/laptop*is the property value. Figure 2.4 shows how this statement may be represented in RDF syntax.

```
<rdf:RDF
    xmlns:rdf=
"http://ww.w3c.org/1999/02/22/-rdf-syntax-ns#"
    xmlns:prod="http://www.ourproducts.org/hp#"
    xmlns:item="http://www.ourproducts.org/prod/laptop"
>
<rdf:Description rdf:about="hp">
  <prod:produces rdf:resource="item:laptop"/>
</rdf:Description>
```

Figure 2.4: RDF Representation of a Statement (Antoniou and Frank, 2004)

2.5.2 RDFS

XML Schema is used to validate or prove the legality of XML file. Resource Description Framework Schema (RDFS) is not really used the same way XML schema is used. RDFS is used to specify the specific domain that RDF statements are describing. RDF themselves do not take assumptions about a particular domain. RDFS as universal language helps users to be able to describe their own vocabulary. RDFS also provide means for RDF specified objectsto

be classifiedinto objects seen as collected classes. For instance, having a class of online stores or retailers of which HP is an object or instance. Relationship between classes and objects is described using *rdf:type*. Classes are used to impose what can be stated in the RDF using the schema. For example, to say that "*laptop computer is sold by desktop computer"*is not only vague but does not make any sense. RDF Schema could have been used to limit what classeslaptop computer and desktop computerbelong to and what *range* and *domain* of values the property *isSold* will take. Hierarchical definition can be defined for both class and property in RDFS. For example, consider that there is a product called computer system while laptop and desktop could be seen as subclasses of computer. Figure 2.5 illustrates this hierarchical representation of classes and properties. Figure 2.6 shows how to state that all laptops are computer systems, whenever a laptop's model is stated, then it is a dual core and it is Literal.

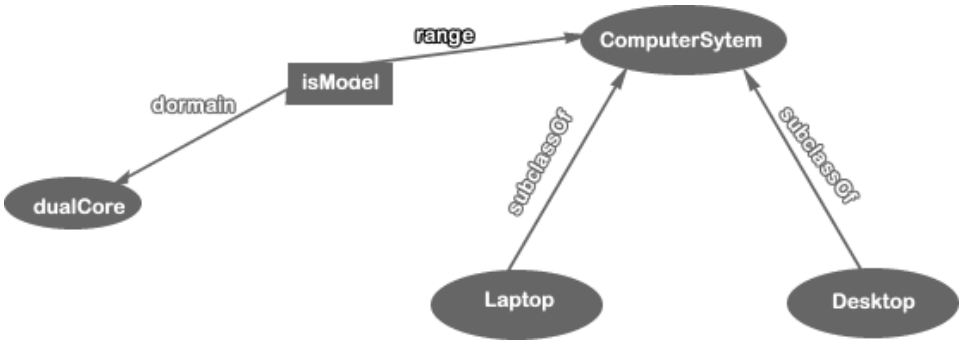


Figure 2.5: Graphical Representation of Classes and Properties in Hierarchy(Antoniou and Frank, 2004)

```
"http://ww.w3c.org/1999/02/22/-rdf-syntax-ns#"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
<rdfs:Class rdf:ID="ComputerSystem">
<rdfs:comment>The class of Computer System</
rdfs:comment>
</rdfs:Class>
<rdfs:Class rdf:ID="Laptop">
<rdfs:subClassOf rdf:resource="#ComputerSystem"/>
</rdfs:Class>
<rdfs:Class rdf:ID="Desktop">
<rdfs:subClassOf rdf:resource="#ComputerSystem"/>
</rdfs:Class>
<rdf:Property rdf:ID="isModel">
<rdfs:domain rdf:resource="#dualCore"/>
<rdfs:range rdf:resource="#computer"/>
</rdf:Property>
```

Figure 2.6: RDFS Illustration of Classes and Properties(Antoniou and Frank, 2004)

2.5.3 OWL

Web ontology language mostly called OWL is an extension of RDFS described in Section 2.5.2. OWL was designed as an improvement on RDFS in order to be able to increase expressivity, enhance reasoning over the ontology represented by the ontology language. OWL still leverages on the XML-based RDF syntax to express its concepts. RDFS generalizes instances modeled by RDF by using rdf:Class and other types of conceptualization that exist under the generalized concept using “rdf:subclass”. Similarly, RDFS uses rdf:onProperty and rdf:subProperty relations to convey the relationship that exists among classes or resources and their corresponding objects. However, OWL adds more features that take this ontology representation concept into a better expressive stage. Some of these added features include: (Obiniyi *et al.*, 2014)

a) Local scope of property: Even though it is right to use the rdf:range to define the range of a property for all classes, but with OWL feature owl:restriction, it will be possible to declare a range of restrictions that apply to a specific class.

b) Disjointness of Classes: Sometimes, to express the fact that even though two or more classes may be subclasses of the same class, which are not joined, OWL "owl:disjointWith" may be used to express this fact.

c) Boolean combination of classes: RDFS is limited in the expression of the combination of different classes using relative terms. OWL has owl:unionOf and owl:intersectionOf which are better solutions of this deficiency.

d) Special characteristics of properties: There are special characteristics of property of a class that may not easily represent using RDFS. With OWL, characteristics such as property transitivity, inverse and uniqueness may be achieved using owl:inverseOf, owl:transitivityProperty may be used for these purpose.

OWL constructors like owl:Class, owl:ObjectProperty and owl:DatatypeProperty are specializations of their corresponding RDFS features. OWL stem from DAML+OIL and has three flavors which are; OWL FULL, OWL Lite and OWL DL. OWL FULL has more expressive power compared to its counterparts. Figure 2.7 is an example of using OWL to express the fact that*product class is composed of Electronics and Automobile products* and that there is an inverse statement which may be expressed by saying *a retailer sells Electronics and Electronic products are sold by retailer.*

```
xmlns:owl ="http://www.w3.org/2002/07/owl#"
xmlns:rdf =
"http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:xsd ="http://www.w3.org/2001/XMLSchema#">
<owl:Class rdf:ID="products">
<owl:unionOf rdf:parseType="Collection">
<owl:Class rdf:about="#Electronics"/>
<owl:Class rdf:about="#Automobile"/>
</owl:unionOf>
</owl:Class>
<owl:ObjectProperty rdf:ID="sells">
<rdfs:range rdf:resource="#Electronics"/>
<rdfs:domain rdf:resource="#retailer"/>
<owl:inverseOf rdf:resource="#soldBy"/>
</owl:ObjectProperty>
```

Figure 2.7: A Typical OWL File(Antoniou and Frank, 2004)

2.5.4 OWL2

Despite the success story surrounding OWL, the numerous contexts in which the language has been applied have revealed some deficiencies in the original design such asExpressivity Limitations, Qualified Cardinality Restrictions, Relational Expressivity, Datatype Expressivity and Syntax Issues (Cuenca*et al*, 2008).

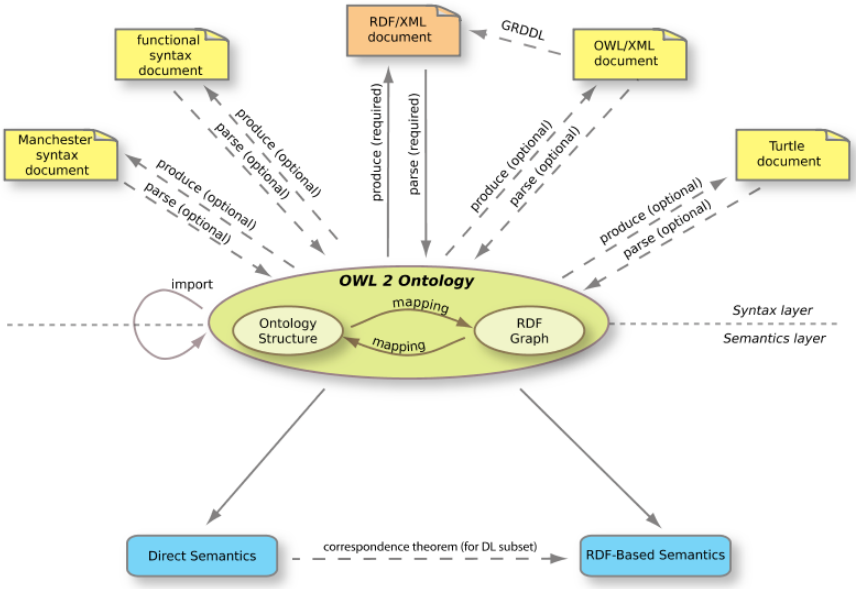


Figure2.8: Structure of OWL2(WOL,2016).

Figure 2.8 gives an overview of the OWL 2 language, showing its main building blocks and how they relate to each other. The ellipse in the center represents the abstract notion of an ontology, which can be thought of either as an abstract structure or as an RDF graph. At the top are various concrete syntaxes that can be used to serialize and exchange ontologies. At the bottom are the two semantic specifications that define the meaning of OWL 2 ontologies. Most users of OWL 2 will need only one syntax and one semantics; for them, this diagram would be much simpler, with only their one syntax at the top, their one semantics at the bottom, and rarely a need to see what's inside the ellipse in the center.(WOL,2016).

2.6. OWL 2 Web Ontology Language New Features and Rationale

OWL 2 adds several new features to OWL 1, including increased expressive power for properties, extended support for datatypes, simple metamodeling capabilities, extended annotation capabilities, and keys,(WOL,2016).Some of the new features of OWL 2 are presented here, organized in the following categories:

a) syntactic sugar,
b) new constructs that increase expressivity,
c) extended support for datatypes,
d) minor features.

2.6.1 Syntactic sugar

OWL 2 adds syntactic sugar to make some common patterns easier to write. Since all these constructs are simply shorthands, they do not change the expressiveness, semantics, or complexity of the language,(WOL,2016).Examples of the OWL2 syntactic sugar are:

a) **DisjointUnion:**DisjointUnion defines a class as the union of other classes, all of which are pairwise disjoint. Example:
   DisjointUnion(:CarDoor :FrontDoor :RearDoor :TrunkDoor). Which means A *:CarDoor* is exclusively either a *:FrontDoor,* a *:RearDoor* or a*:TrunkDoor* and not more than one of them.
b) **DisjointClasses:** DisjointClasses states that all classes from the set are pairwise disjoint. Example: DisjointClasses(:LeftLung:RightLung). Which means nothing can be both a :LeftLung and a :RightLung.
c) **NegativeObjectPropertyAssertion and NegativeDataPropertyAssertion:**While OWL 1 provides means to assert values of a property for an individual, it does not provide a

construct for directly asserting values that an individual does not have (negative facts). NegativeObjectPropertyAssertion and NegativeDataPropertyAssertion) states that a given property does not hold for the given individuals.

Example(i): NegativeObjectPropertyAssertion(*:livesIn:ThisPatient:IleDeFrance*). Which means *thisPatient* does not live in the *:IleDeFrance* region.

Example(ii):NegativeDataPropertyAssertion(*:hasAge:ThisPatient* 5^^*xsd:integer*). Which means *thisPatient* is not five years old.

2.6.2 New constructs for properties.

OWL 1 was mainly focused on constructs for expressing information about classes and individuals, and exhibited some weakness regarding expressiveness for properties,WOL,2016). OWL 2 offers new constructs for expressing additional restrictions on properties, new characteristics of properties, incompatibility of properties, property chains and keys. Examples of the new property construct in OWL2 are:

a) **SelfRestriction.**
   OWL 1 does not allow for the definition of classes of objects that are related to themselves by a given property, for example the class of processes that regulate themselves. This "local reflexivity" is useful in many applications, particularly when global reflexivity does not hold for a property in general, but local reflexivity holds for some classes of object. The OWL 2 construct **ObjectHasSelf** allows local reflexivity to be used in class descriptions. Example: SubClassOf(*:AutoRegulatingProcess* ObjectHasSelf(*:regulate*)). Which means Auto-regulating processes *regulate* themselves.

b) **Property Qualified Cardinality Restrictions.**
   While OWL 1 allows for restrictions on the number of instances of a property, e.g., for defining persons that have at least three children, it does not provide a means to

restrain the *class* or *data range* of the instances to be counted (*qualified* cardinality restrictions), e.g., for specifying the class of persons that have at least three children who are girls. In OWL 2, *both* qualified and unqualified cardinality restrictions are possible.ObjectMinCardinality, ObjectMaxCardinality, and ObjectExactCardinality (respectively, DataMinCardinality, DataMaxCardinality, and DataExactCardinality) allow for the assertion of minimum, maximum or exact qualified cardinality restrictions, object (respectively, data) properties.

Example(i): ObjectMinCardinality(5 *:hasDirectPartowl:Thing*). Which means a class of objects having at least 5 direct part.

Example(ii):ObjectExactCardinality(1 *:hasDirectPart:FrontalLobe*). Which meansa class of objects having exactly one *direct part* of type *frontal lobe.*

Example(iii): ObjectMaxCardinality(5 *:hasPart:Door*). Which means a class of objects having at most 5 *:Door.*

Example(iv): DataMaxCardinality(1 *:hasSSN*). Which means each individual has at most one Social Security Number.

c) **Reflexive, Irreflexive, and Asymmetric Object Properties.**

While OWL 1 allows assertions that an object property is symmetric or transitive, it is impossible to assert that the property is reflexive, irreflexive or asymmetric. The OWL 2 construct ReflexiveObjectProperty allows it to be asserted that an object property expression is globally reflexive - that is, the property holds for all individuals.

Example(i): ReflexiveObjectProperty(*:sameBloodGroup*). Which means everything has the same blood group as itself.

Example(ii): IrreflexiveObjectProperty(*:flowsInto*). Which means nothing can flow into itself.

Example(iii): AsymmetricObjectProperty(*:proper_part_of*). Which means the property *:proper_part_of* is asymmetric.

d) **Disjoint Properties.**

While OWL 1 provides means to state the disjointness of classes, it is impossible to state that properties are disjoint. The OWL 2 construct **DisjointObjectProperties** allows it to be asserted that several object properties are pairwise incompatible (exclusive); that is, two individuals cannot be connected by two different properties of the set, and the OWL2 DisjointDataProperties allows it to be asserted that several data properties are pairwise incompatible.

Example(i): DisjointObjectProperties(*:connectedTo:contiguousWith*). Which means *connectedTo* and *:contiguousWith* are exclusive properties.

Example(ii): DisjointDataProperties(*:startTime:endTime*). Which means start time of something, e.g., surgery, must be different from its end time

e) **Property Chain Inclusion.**

OWL 1 does not provide a means to define properties as a composition of other properties, as uncle could be defined; hence, it is not possible to propagate a property (e.g.; *locatedIn*) along another property (*e.g.; partOf*). The OWL 2 construct ObjectPropertyChain in a SubObjectPropertyOf axiom allows a property to be defined as the composition of several properties.

Example: SubPropertyOf(ObjectPropertyChain(*:locatedIn:partOf*) *:locatedIn*). Which means if x is *located in* y and y is *part of* z then x is *located in* z, for example a disease located in a part is located in the whole.

f) **Keys.**

OWL 1 does not provide a means to define keys. However, keys are clearly of vital importance to many applications in order to uniquely identify individuals of a given

class by values of (a set of) key properties. The OWL 2 construct HasKey allows keys to be defined for a given class.

Example(i): HasKey(*:RegisteredPatient:hasWaitingListN*).Which means each registered patient is uniquely identified by his waiting list number.

Example(ii): ClassAssertion(*:RegisteredPatient:ThisPatient*). Which means *ThisPatient* is a *:RegisteredPatient*.

Example(iii): DataPropertyAssertion(*:hasWaitingListN:ThisPatient* "123-45-6789"). Which means *ThisPatient* has the the waiting list number "123-45-6789".

Example(iv): HasKey(*:Transplantation* :donorId:recipientId:ofOrgan *)*. Which means each Transplantation is uniquely identified by a donor, a recipient, and an organ.

2.6.3 Extended datatype capabilities.

a) **Extra Datatypes and Datatype Restrictions.**

OWL 1 provides support for only integers and strings as datatypes and does notsupport any subsets of these datatypes,(WOL,2016). For example, one could state that every person has an age, which is an integer, but could not restrict the range of that datatype to say that adults have an age greater than 18. OWL 2 provides new capabilities for datatypes, supporting a richer set of datatypes and restrictions of datatypes by facets.Example: DatatypeRestriction(xsd:integer minInclusive 18). Which means new datatype with a lower bound of 18 on the XML Schema datatype xsd:integer.

b) **N-ary Datatypes.**

In OWL 1 it is not possible to represent relationships between values for one object, e.g., to represent that a square is a rectangle whose length equals its width. N-ary datatype support was **not** added to OWL 1 because there were issues regarding just what support should be added. However, OWL 2 includes syntactic constructs needed for n-ary datatypes, to provide a common basis for extensions.

Example: DataAllValuesFrom (*:admissionTemperature:currentTemperature* DataComparison(Arguments(x y) leq(x y)))). Which means an adult age is defined by using a lower bound of 18 with the XML Schema datatype xsd:integer.

c) **Data Range Combinations.**

While OWL 1 allows a new class to be constructed by combining classes, it does not provide means to construct a new datatype by combining other ones. In OWL 2 it is possible to define new datatypes in this way. In OWL 2, combinations of data ranges can be constructed using intersection (DataIntersectionOf), union (DataUnionOf), and complement (DataComplementOf) of data ranges. Example: DataComplementOf(*:adultAge*). Which means this data range contains all literals that are not a positive integer greater or equal to 18.

2.6.4 Minor features.

According to W3C, 2012 ,some of the minor features introduce in OWL2 are represented below.

a) **Inverse Properties.** In OWL 1, all properties are atomic, but it is possible to assert that some object property is the inverse of another property. In OWL 2, property expressions such as ObjectInverseOf(P) can be directly used in class expressions. This makes writing ontologies easier by avoiding the need to name an inverse. An inverse object property expression ObjectInverseOf(P) connects an individual a_1 with

a_2 if and only if the object property P connects a_2 with a_1. An inverse object properties axiom InverseObjectProperties(OPE_1 OPE_2) states that two properties are inverse.

Example(i): ObjectProperty(*:hasPart* inverse *:partOf*). Which means *hasPart* has an inverse property named *:part of.*

Example(ii): EquivalentProperties(*:hasPart* ObjectInverseOf(*:partOf*)). Which means *partOf* is the same as the inverse property of *:hasPart*.

Example(iii): InverseObjectProperties(*:hasPart:partOf*). Which means *hasPart* and *:partOf* are inverse properties.

2.7Query Languages in Semantic Web

According to Oyelade,*etal.,*(2014), Datahas been modeled using the semantic web technologies, however, this is not usually the final goal of owners of such datasets. Publishing such datasets so that it will be accessible by other users will actually be the goal of such data modeling. Some decide to expose their datasets to prospective users through end points that allows users to use specific query language to view the data. These query languages include SPARQL and SQWRL.

2.7.1 SPARQL

SPARQL Protocol and RDF Query Language (SPARQL) is a query language that was designed to query or retrieve information contained in an RDF data model. Though there are several other XML query languages used in querying XML, resting on the fact that XML is lower in the semantic web technology stack compared to RDF, it becomes pertinent to have different query language for RDF. With SPARQL, it is possible to write out a graph pattern that will be used against an RDF file and any matching graph pattern is retrieved from the RDF. Similar to RDF, SPARQL represents graph in the form of a triple. Variables are sometimes used in the positions of *subject, predicate* and *object.* Just like prefixed are used to

replace namespaces in RDF, SPARQL also provides prefix in order to shorten the use of namespaces. For example to write a SPARQL that will retrieve all triple patterns which display all products that are in a particular category, say automobile, the query in Figure 2.8 below shows how to structure out this SPARQL.

```
PREFIX rdf:
<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX prod: <http://www.mydomain.org/uni-ns#>
SELECT ?c
WHERE
{
     ?c rdf:type prod:Automobile .
}
```

Figure 2.9: SPARQL Graph Pattern(Obiniyi *et al*, 2014)

SPARQL also has some other features to refine a query to probably limit what result it retrieves. For example, filtering out the results retrieved from a query that list out available products under a category is possible. The query below filters out products that are only in the categories of wears.

```
PREFIX rdf:
<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX prod: <http://www.mydomain.org/uni-ns#>
SELECT ?c
WHERE
{
     ?c rdf:type prod:allCategroy .
     ?c prod:name prod:Wears .
     FILTER (?c = ?x).
}
```

Figure 2.10: SPARQL with restrictions(Obiniyi *et al*, 2014)

SPARQL can be used to describe the nature of an RDF graph, construct a new graph from an existing graph and to find out information about the nature of the graph. And SPARQL features such as CONSTRUCT and DESCRIBE respectively may be used for this purpose.

2.7.2 Semantic web rule language (SWRL)

Semantic Web Rule Language (SWRL) is a rule language that works in handy with query languagessuch as SQWRL (O'Connor *et al*, 2005). This rule language is an appropriate rule language for Semantic web. It adds simple Horn-style rules to OWL and OWL2Defeasible Logic DL in a syntactically and semantically coherent manner and all rules are expressed in terms of OWL and OWL2 concepts (classes, properties and individuals). Rules implemented with SWRL are saved in the same ontology in which they are being used and they can work with reasoners. It enables the combination of Horn-like rules and an OWL knowledge base. This combination implies the use of OWL2 and Rule Markup Language (RuleML). SWRL rules are of the form of antecedent (body) and consequent (head). The rule could be read as meaning that if the condition placed on the antecedent holds, then the condition placed on the consequent must hold. Both the antecedent and the consequent may be made of zero or more atoms. If the antecedent is empty, it is treated as trivially true, so the consequent must also be satisfied by every interpretation. Similarly, an empty consequent is treated as trivially false, so the antecedent must not be satisfied by any interpretation. Atoms in this rule can be in the form of A(x), P(x,y), sameAs(x,y) or differentFrom(x,y). Where A and P are representations of a given classes while sameAs and differentFrom are properties all in OWL. For example, if a given ontology has a class called Person, Adult and an object property called hasAge, then, it is correct to write a rule as shown in Rule 2.1

Person(?x) ^ hasAge(?x, 20) -> Adult(?x) Rule 2.1

This simply means that if a person's age is 20, then such a person must be an adult. SWRL have built ins that can help rule writers to further make their rules rich by using some basic functions in those built-ins. Consider Rule 2.2

Person(?x) ^ hasAge(?x, ?age) ^ swrl:greaterthan(?age, 25)-> Adult(?x) Rule 2.2

This rule implies that if a person is of age greater than 25, such a person is an Adult.

2.7.3 Semantic query-enhanced web rule language (SQWRL)

In order to be able to query ontology written in OWL, some query languages have been used. Some of these include SPARQL. Since SPARQL was actually meant for RDF, it implies that querying ontology written in OWL will demand that ontology be serialized in RDF. SPARQL does not understand the OWL language; hence retrieving information from an OWL file will not be as accurate as desired. A paper titled "SQWRL: a Query Language for OWL" (O'Connor and Amar, 2009) pointed out that SPARQL operates on its RDF serialization and has no knowledge of what language constructs that those serialization represents, therefore, cannot directly query the entailments made using those constructs. Moreover, there is no canonical RDF serialization of some OWL constructs. SQWRL was built upon SWRL and it replaces the consequent part of SWRL rule with a query. SQWRL uses SWRL's built in facility as an extension point. And with these built-ins, it defines operators that help to add more richness and more enhancements, compared to SPARQL, to the query language. The relevance of this is that no syntactic extensions are required to SWRL. Hence, standard SWRL serialization mechanism allows the result of the rule to be added to the underlying ontology. When executing SQWRL query, first SWRL inference rules are taken into account/executed. Running SQWRL requires a rule engine, currently Jess.

Some of the advantages of using SQWRL as a query language over SPARQL are; we do not need to invent new semantics, we have a standard serialization, it can use existing reasoning infrastructure, it is easier to learn when a knowledge of SWRL has been obtained and it is useful for debugging SWRL. We shall exemplify on how to use SQWRL to execute a query on an OWL file.

Let us still assume we have the same classes, and relations as specified in the previous section, then we could write a query as shown in Rule 2.3

Person(?x) ^ hasAge(?x, 20) -> sqwrl:select(?x) Rule 2.3

This query will retrieve or select all Persons whose age is 20. These results will be placed in the ?x variable. Since it was establish that SQWRL was built upon SWRL, we could write a rule and take the consequent part of the rule to be the antecedent part of the query, thereby shortening the query text. For instance, going by the rule written in Rule 2.2which reads thus;

Person(?x) ^ hasAge(?x, ?age) ^ swrl:greaterthan(?age, 25)-> Adult(?x)

It is possible to write a query like this;

Adult(?x) -> sqwrl:select(?x) Rule 2.4

This query will retrieve all instances of the class Adult which were generated by the rule that says that if a person‘s age is greater than 25, then such person is an adult.

2.8 Intelligent Agents on Semantic Web

According to Wooldridge and Jennings, 1995, an agent can be defined as an application situated in some environment and able to perform autonomous action in order to archive its designed objectives. An intelligent agent is also characterized by some degree of reactivity (ability to respond to changes in an environment), proactive (goal directed behavior) and social ability (capability of interacting with other agents). Intelligent agent can be defined as software programs extended to perform task more efficiently and with less human intervention.Combining the descriptions (Wooldridge *et al.*,1999) and (Russell and Norvig, 2003), it is clear that an agent is an encapsulated computer system made up of architecture and a program. Thanks for the idea of Linked Data. This has greatly enhanced the realization of implementing autonomous and trustworthiness into the semantic web. The semantic webs can

only bring meaning to data in situation whereby an intelligent agent is roaming from page to page on the internet. In today's world, semantic web needs to be autonomous, consistent and trustworthy. An intelligent approach will be required in all semantic web applications in order to realize these characteristics (autonomous, consistency and trustworthy). It is required that applicationss acting as intelligent agents should:

a) Be situated in a particular environment
b) Be able to perceive its environment.
c) Be capable of autonomous action in its environment.
d) Have some kind of design objectives.

These agents must also be made up of four parts:

a) A performance measure.
b) An environment.
c) Actuators
d) Sensors.

Figure 2.11 gives a pictorial view of intelligent agents on the semantic Web

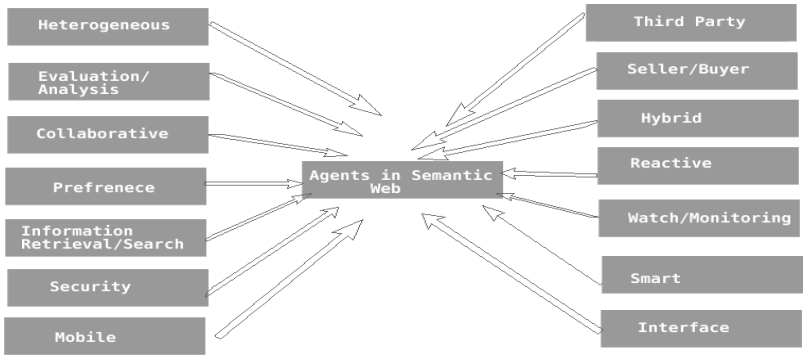


Figure 2.11: Agents in Semantic Web(Obiniyi *et al*, 2014)

2.9 Rule Composition for the Ecommerce System

We can add a SWRL rule saying that an individual X from the Dealers class, which has at the same time belongs to sellers of a category of product Y in a trade fair Z, such that Y was soldduring trade fair Z, belongs to a new class consistentOnlineComputerDealers. Such rule is best described by the following syntax:

> Dealers(?x), belongsTo(?x, ?y), belongsTo(?x, ?z), participateInFair(?y, ?z) ->
> consistentOnlineComputerDealers(?x)

2.10 Tools, Reasoner and Inference Engine

1. Protégé 4.3

Protégé is a free, open-source platform that provides a growing user community with a suite of tools to construct domain models and knowledge-based applications with ontologies. It is an open-source ontology editor and framework for building intelligent systems. http://protege.stanford.edu/products.php).

Why Protégé

Protégé's plug-in architecture can be adapted to build both simple and complex ontology-based applications. Developers can integrate the output of Protégé with rule systems or other problem solvers to construct a wide range of intelligent systems. Protege is actively supported by a strong community of users and developers that field questions, write documentation, and contribute plug-ins. For W3C Standards Support, Protege fully supports the latest OWL 2 Web Ontology Language and RDF specifications from the World Wide Web Consortium. And for the Extensible Open Source Environment, Protege is based on Java, is extensible, and provides a plug-and-play environment that makes it a flexible base for rapid prototyping and

application development. Protégé Desktop is a feature rich ontology editing environment with full support for the OWL 2 Web Ontology Language, and direct in-memory connections to description logic reasoners like HermiT and Pellet.

Protégé Desktop supports creation and editing of one or more ontologies in a single workspace via a completely customizable user interface. Visualization tools allow for interactive navigation of ontology relationships. Advanced explanation support aids in tracking down inconsistencies. Refactor operations available including ontology merging, moving axioms between ontologies, rename of multiple entities, and more. In summary Protégé 4.3 possess the following main characteristics.

a) It is W3C standards compliant
b) It has a customizable user interface
c) It supports visualizaton
d) It supports Ontology refactoring
e) It provides direct interface to reasoners
f) It has highly pluggable architecture
g) It is cross compatible with WebProtégé

2. Pellet 2.2

Pellet is an OWL 2 reasoner in Java; open source (AGPL) and commercially licensed, commercial support available. (https://github.com/complexible/pellet). Pellet implements various reasoner interfaces provided in the OWL- API. Pellet 1.5 uses a newer snapshot of OWL- API including support for OWL 1.1, built from development sources. The Pellet Reasoner Plug-in for Protégé 4 provides full integration of Pellet in the Protégé 4 environment. Pellet supports reasoning with SWRL rules. Pellet interprets SWRL using the DL-Safe Rules notion which means rules

will be applied to only named individuals in the ontology. No additional atoms are needed in the rules to enforce DL-safety. Pellet supports all SWRL atoms described in the specification, see below for details about which built-in functions are supported.You don't need to use any utility function to use SWRL in Pellet. You can directly load a file that contains SWRL rules into Pellet and rules will be parsed and processed. SWRL rules can be mixed with OWL axioms in an ontology, too. See the examples directory in the Pellet distribution for a sample program that shows how to use SWRL rules with Pellet. You can either use the OWLAPI or Jena interface; they both handle SWRL. Using the configuration optionDL_SAFE_RULES, the reasoner can be configured to ignore SWRL rules. Rules support is enabled by default and only the rules that use one of the mentioned unsupported features will be ignored (or an exception will be thrown based on the value of the configuration optionIGNORE_UNSUPPORTED_AXIOMS).

3. Netbeans 7.2

NetBeans IDE 7.2 provides significantly improved performance and an enhanced coding experience, with new static code analysis capabilities in the Java Editor and smarter project scanning. The release also includes notable features such as Scene Builder integration in JavaFX; support for multiple PHP frameworks; updated Groovy support; and many other enhancements in Java EE, Maven, C/C++ and the NetBeans Platform. It's release highlights include:

- Improved Performance
- FindBugs Integration
- JavaFX Scene Builder Integration
- Enhanced Java Editor Tools
- Enhanced Java EE Features

- PHP 5.4 Support and more
- Updated Groovy and Grails Support
- TestNGSupport (http://www.theserverside.com/news/thread.tss?thread_id=66742)

4. JDK.

Java Development Kit (**JDK**), which is a software development environment used for developing Java applications and applets. It includes the Java Runtime Environment (JRE), an interpreter/loader (java), a compiler (javac), an archiver (jar), a documentation generator (javadoc) and other tools needed in Java development.(https://dzone.com/articles/how-to-prepare-the-macbook-pro-for-java-developmen)

5. Jess Rule Engine

Jess is a tool for building a type of intelligent software called Expert Systems. An Expert System is a set of rules that can be repeatedly applied to a collection of facts about the world. Rules that apply are fired, or executed. Jess uses a special algorithm called Rete to match the rules to the facts. Rete makes Jess much faster than a simple set of cascading if.. then statements in a loop. Jess was originally conceived as a Java clone of CLIPS, but nowadays has many features that differentiate it from its parent. Often there is one "released" version and one "test" version available for download at any one time. As of this writing, version 7.1p2 is the stable version. Jess 8 is currently the test version. Jess 6.1, which works with JDK 1.2, Jess 5.2, an older release that works with JDK 1.1, and Jess version 4.5, an even older release that is compatible with JDK 1.0.2, are also available, but are not supported.

2.11 Related Works

In this section, the focus is to review four works that are related to this dissertation.

2.11.1 A Framework for an Ontology-based E-commerce Product Information Retrieval System Zhang *et al.* (2009)

This paper presents a framework for an ontology-based e-commerce product information retrieval system and proposes an ontology-based adaptation of the classical Vector Space Model with the consideration of the weight of product attribute.

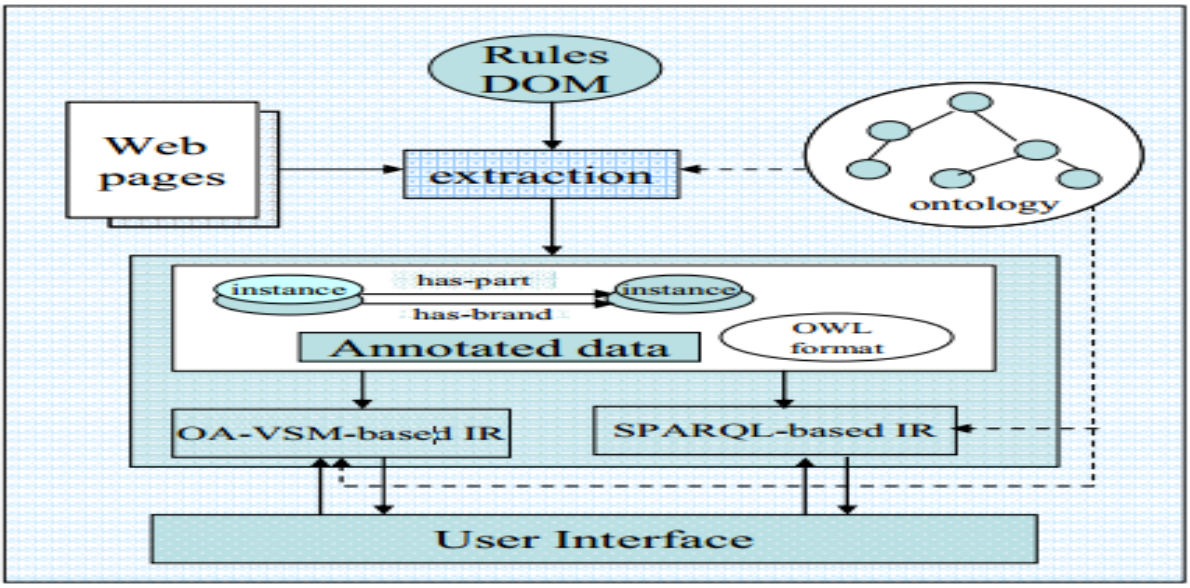


Figure2.12Architecture of the Product IR Framework. (Zhang*et al.*, 2009)

A computer and components related ontology was built, which is adopted to annotate the html documents and construct concept vectors of the documents. Then the system test is done and the experimental result indicates that our proposal is better than the traditional keywords based search, but OWL was used for the ontology and SPARQL was adapted for the query.

2.11.2 Semantic Integration Of Heterogeneous Web Data For Tourism Domain Using Ontology Based Resource Description Language (*Jayaprabha and Saradha ,2010)*.

Proposed a generic approach to transform the existing web contents related to tourism domain into semantic web format using RDF with ontology.

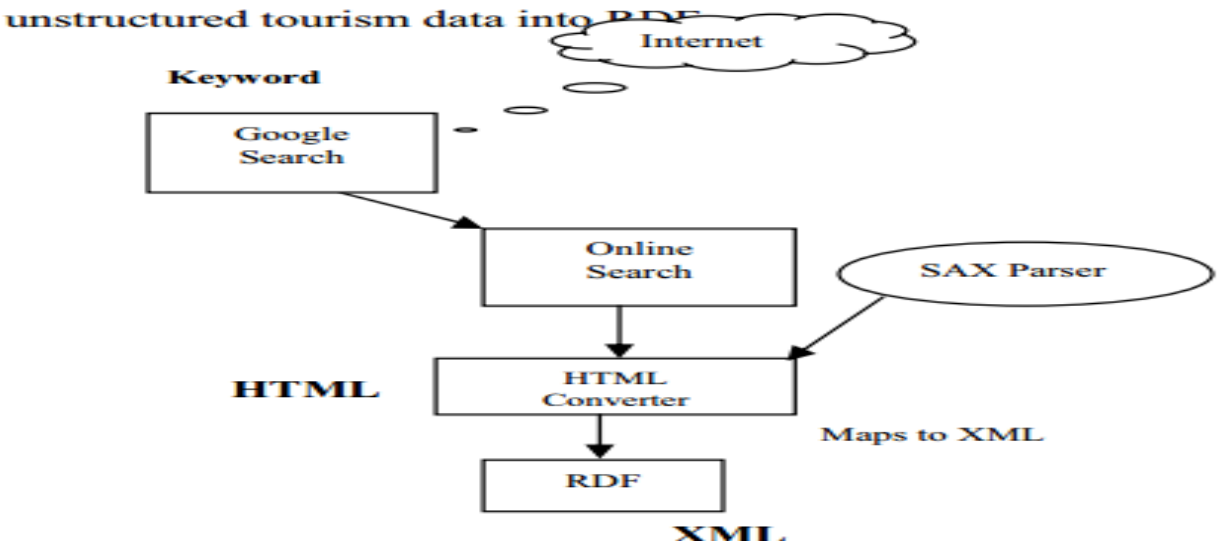


Figure2.13: The Block Diagram Showing The Downloading unstructured text using online search *(Jayaprabha and Saradha, 2010).*

The system has three modules for converting unstructured tourism data into RDF as seen in Figure 2.13, the first module handles the downloading of unstructured text using online search. It extracts heterogeneous tourism related text based on the keyword specified in the online search dialogue by linking the Google search engine to get the documents in HTML format. The second module Converting HTML into XML, by using a HtmlConvertor which in turn uses SAX parser to read HTML data and converts to XML. The next module is the Mapping Process. The XML documents are mapped an ontological based RDF file called Travel2.owl which contains the travel related information in turns of classes, instances, sub instances, object property, relation, data property, general axioms etc. When the travel related heterogeneous unstructured data from various web sites are extracted and converted into semantic web format called RDF. They are implemented in java. And the data can be made available for web using semantic search engine, however, based on the fact that RDF ontology language that was used has limited expressive power compared to OWL2, it strengthen our motivation to this work.

2.11.3 Modeling and Configuration of Process Variants for On-boarding Customers to IT Outsourcing (Yao*et al,* 2012).

Yao *et al,(*2012), worked on models and rules for customer on-boarding – a move the customer's IT environment into the service provider's mode of operation. The alluded that in shifting the customer's IT environment to the service provider's side, there are bound to be different scenarios from varying customers that are going to arrive at the service provider's side. Hence, there arise a need for automating the process variant configuration of each customer on the service provider's side.

As a result, they adopted the use of ontology and rules to both model the standard on-boarding process and configure process variants based on the business context that characterizes various scenarios, and the semantic rules model adaptation policies and help generate a customized process variant schema on the fly. These two methods put together provided a flexible and cost-benefit system for decongesting the customer ends at the expense of ceding some tasks to the service provider end.

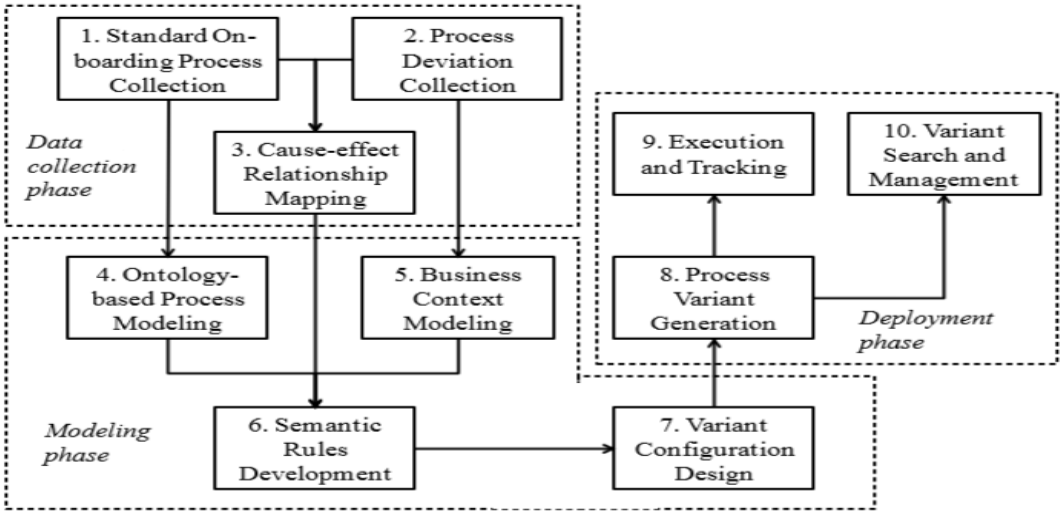


Figure 2.14: The Design Methodology of Customer Process Variant (Yao*et al.*, 2012)

Figure 2.14 shows the approach taken by Yao*et al.*(2012) in implementing their framework. First, they collated standardized on-boarding processes, then the studied and identified where there differences lies. This was to enable those spot out the point of intersection or relationship between them. Next, the ontology of these varying process of customers need was model in an ontology using OWL. Moreover, in other to contextualize the business environment in which this systems works, the business context was also modeled. Then, is the formulation of the semantic rules which models policies and generates a personalized process variant schema on the fly. Next is the design of the variant configuration pattern and its creation. Finally, they implemented the design to execution and tracking and as well maintain the entire process.

As earlier pointed out in the last paragraph, this research work modeled the ontologies used with OWL and implemented their rules using Jena. However, it must be noted that giving that RDF less expressive compared to OWL (recommended by W3 organization), OWL (also known as OWL 1) is now also considered less expressive compared to OWL 2 which was recommended by the same organization in 2009. OWL 2 adds features that don't change expressiveness, semantics, and complexity. However, it makes some patterns easier to write, allowing more efficient processing in reasoners. For instance, to show that some classes/instances are all possible in a situations but that not all of them can be obtainable at the same time, may not be model with OWL 1, but OWL 2 provides Disjoint Union construct for demonstrating this. Also, in a situation whereby they seek to model that instances of two different classes of the ontology cannot belong to more than one classs, may not be possible to model such in OWL 1. Hence OWL 2 provided Disjoint Classes construct for showing this. Furthermore, there are other scenarios that can be captured in modelling customers-business ontology as presented in this research work. But limitations of more expressive constructs in OWL 1 may hindered this. Considering the two constructs mentioned here and the improved cardinality constructs available in OWL 2. For example, OWL 2 qualifies the instances to be

counted, should we want to count them. Six varieties provided are: Data, Object, Min, Exact, Max, and Cardinality.

2.11.4 Engineering Semantic Web for E-Commerce Business Intelligence: ABilingual EEPS Ontology Model(Kerie, 2014).

Kerie (2014),developed a commerce related ontology. The basic focus of the research work was to improve the visibility of web, improve search engine optimization, and to realize semantic interoperability. He developed the ontology for Ethiopian Export Products and Services (EEPS) ontology. The work shows that the ontology is consistent and was derived from the existing heterogeneous products classification using GenTax algorithm, and was made to fall into the standard of GoodRelations - a lightweight, generic vocabulary for the Semantic Web that allows expressing all typical aspects of offers for goods and services on the Web, and was created by the E-business web science research group.

The ontology (EEPS) designed by this research work was well designed and implemented to constitute concepts, classes, object properties, data properties, relationships, instances and axioms in the domain. One thing that puts the EEPS ontology above related standard ecommerce ontology is that it is domain-specific, and tied to the products and services rendered within the domain of interest. Moreover, it is a bilingual ontology modeled in OWL2 DL. Kerie carried out inferencing with description logic (DL) in other to model a mechanism for a enabling a full-fledged semantic ecommerce framework. Meanwhile, SPARQL was adopted for querying to confirm whether the, competency questions are answered or not.

Though this research work presented a good framework for improving web visibility and search engine optimization, however, some limitations are identified with it. First, though a more expressive ontology language – OWL2 DL – was used in modelling the ontology, however, the use of SPARQL – originally tailored for querying ontology modeled with RDF –

was used for answering question from the underlying ontology. This is why this proposed research work argues that modelling OWL2 DL or any other more expressive ontology language is not sufficient. But being able to exercise the querying task with an equivalent query language would have helped in improving performance – improve web visibility and web search optimization. In this case of Kerie (2014) work, SQWRL would have being a suitable query language for answering the highlighted questions.

2.11.5 Enhancing Reasoning and Retrieval Performance on OWL using SQWRL (Obiniyi *et al.*, 2014).

They demonstrated that search on ecommerce website can be improved through the combine use of OWL and SQWRL in ontological modeling and query answering activities. The research work shows a client-server architecture (see Figure 2.15) that allows users or

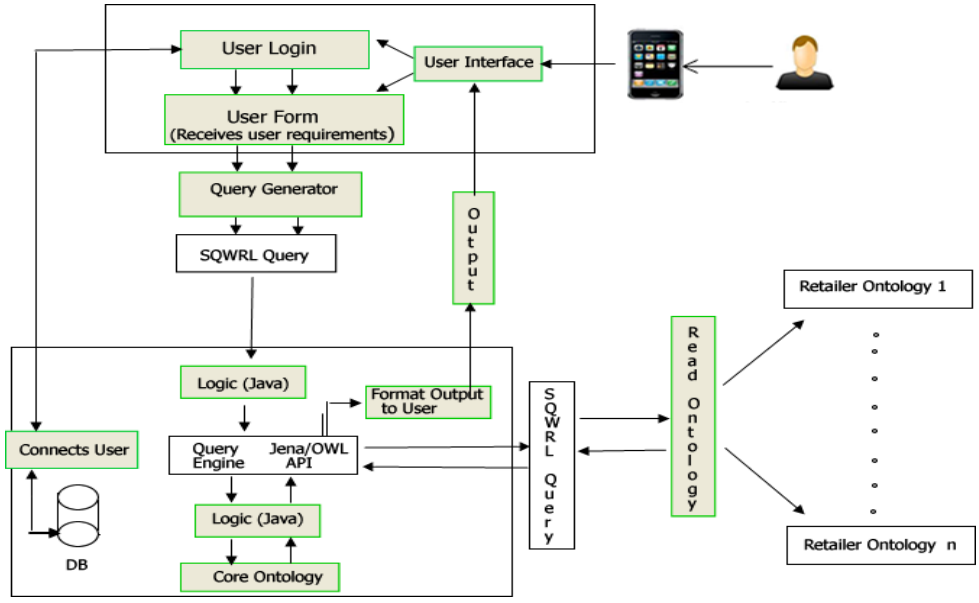


Figure 2.15: System Architecture of Obiniyi *et al. (*2014)(Obiniyi *et al,* 2014)

buyers to make their purchases the users access the server through their online purchase application deployed on their mobile phone. Two ontology were modeled. One for the server side, which serves as a registry for the registered retailers. The other ontology modeled was the retailer ontology; several instances of this ontology could exist on the self-managed online

repository of the retailers. However, it was observed that though the research argues that search optimization is their target, however, because of their choice of OWL as a modelling language, it was noted that this could have impaired the result they presented to be optimal. For instance, there was not means to model that a products that is being searched for by the user does not exist. This is because Obiniyi *et al*, (2014) knowledge base does not avail the ontology developer the power to show the negative of a statement. But in this work provides two important constructs to model the negation of a statement. These two constructs are: NegativeObjectPropertyAssertion and NegativeDataPropertyAssertion. With NegativeObjectPropertyAssertion, an object could be related with a negated predicate which in turns relates that object with a subject. For example, NegativeObjectPropertyAssertion is useful in showing that: Italian Shoe is not sold in Konga store, while the NegativeDataPropertyAssertion is usable in a situation whereby one seeks to model that: Introduction to Java Programming Book is not 60 in quantity on Amazon bookstore. All these are but unavoidable situations that must be molded in the ontology, moreover, it is not realizable in OWL 1 but with OWL 2.

CHAPTER THREE

ARCHITECTURE AND ONTOLOGY MODEL OF THE E-COMMERCE SYSTEM

3.1 Introduction

In this chapter, a theoretical implementation of the enhanced system that is being proposed by this research is hereby presented. First, the architecture of the proposed system is first presented. Subsequently, the comprising components of the entire system are outlined and further discussed. The ontology used by the framework in this system are categorized into two: the central registry ontology (also known as core ontology) and the retailer ontology. These two ontology are further modeled, explaining what influenced the choice of their comprising concepts, properties and individuals. Lastly, the business policy of online commerce activities are modeled using semantic rule language. Subsequently, the rules are plugged in to core ontology for entailment of new knowledge.

3.2 Overview of the Proposed System

The system architecture and the data flow are discussed in this section. The system architecture gives a pictorial view of the proposed system and how its components relates together. On the other hand, the data flow gives a scenario flow of event in the system. While section 3.2.1 focuses on the system architecture, section 3.2.2 demonstrates the typical flow of events and data in the system.

3.2.1 System Architecture

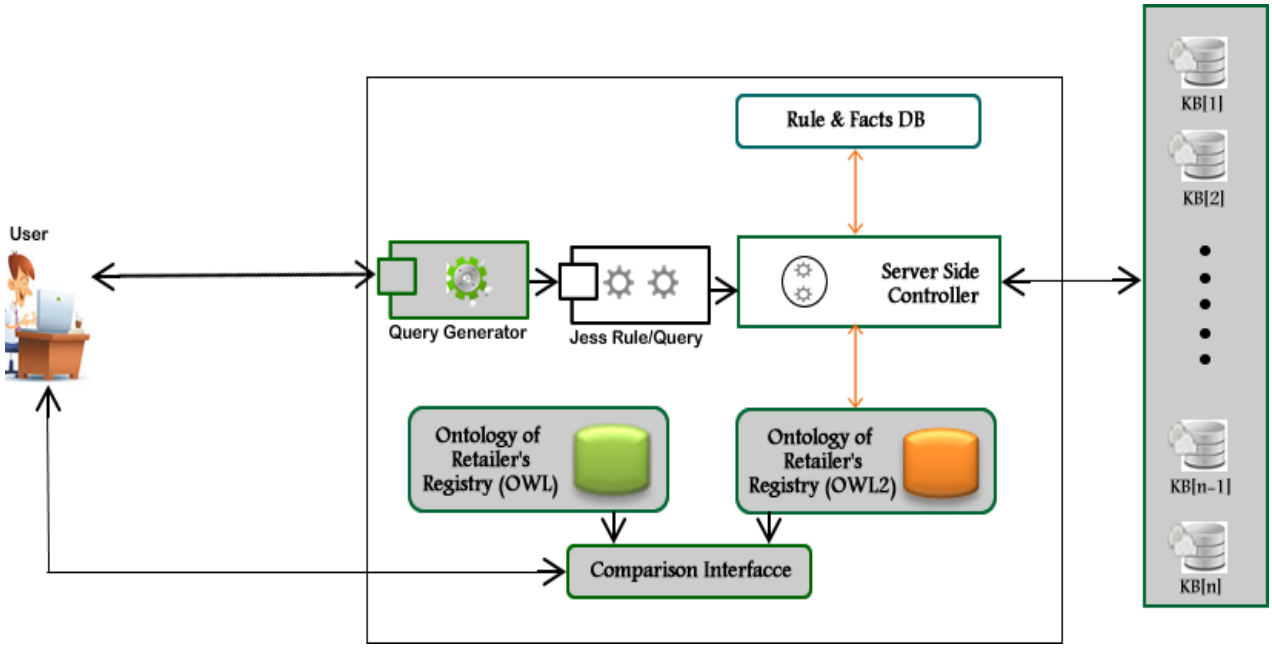


Figure 3.1: Architecture of the proposed Ecommerce System

Figure 3.1 is the model that captures the major components of the proposed ecommerce system. The system architecture consists of a query parser, a query generator, a controller that handles the flow of events and data, ontology of registered retailers' and dealers, a database of rules and facts, and finally a distributed retailer's ontologies. Section 3.2.2. lists out the components and their functions.

3.2.2 Components of the System

The components of the system architecture are categorized into the client side, server layer, and the distributed retailer knowledgebase. These are further discussed in the following items.

1.Client Layer (User end)

The client layer serves as front-end for the users of the system which is the online buyers. The client layer provides the users with two user interactive interface. These are the query entry interface and the result display interface. Users or buyers are expected to search for their

desired product through the query entry. The query entry interface accepts inputs in an English-like manner. This query is comparable to that of typing search request into the Google search engine. However, this proposed system seeks to convert the English-like query into a semantic web understandable query. This will enable the efficient querying of the ontological knowledgebase.

2. Query Generator

The query generator automatically composes a semantic web query using the Semantic Query Web Rule Language (SQWRL) based on the semantic web tokens sent to it. The work of the query generator interfaces between the semantic web based ecommerce search engine and the users end. The query generated by this component is further sent to the controller.

3. Rule & Facts Database.

Rules are necessary in this system. The essence of the relevance of incorporating a rule set into this system is to easily entail supplementary information that is not originally modelled in the ontological knowledgebase. This rules are invoked against the knowledgebase. Note that this system does not depend only on the specified rule set. It also work with a reasoner. Reasoners like Pellet, Racer and Fact ++ are useful in inferring what is not readily available in the ontological database. In this research, Pellet 2.2 is being used for the reasoning task.

4. Ontology of Retailer's Registry.

This proposed system uses two ontologies. The core ontology and the retailer or dealer ontology. The core ontology serves as a registry for all registered retailers or dealers on the web page. This registry are usually looked up enroute in search of information about products online. Sections 3.3.1 and 3.3.2 deals with the ontologies of the retailer registry and retailer product respectively.

5. Server Side Controller.

This module handles the logical coordination of the entire system. All necessary call to other components and sub modules are being coordinated by this controller. The controller also control access to the retailer products ontology or database. This enables it to present the user or the online searcher with the result of the search being carried out.

3.3 Ecommerce Ontology Development

The ecommerce ontology consists of the central registry ontology (also known as core ontology) and the retailer registry ontology. They are discussed in details in this section, and are visually illustrated.

3.3.1 The Core Ontology and Retailer's Registry

In this section, the core ontology is presented and discussed. The logical implication of some of its most relevant modelling concepts are briefly and closely discussed. Figure 3.2 illustrates the ontological knowledge base designed for this dissertation. It consists of the following concepts: Dealers (which in turn consists of the following subclasses: ProducerDealer, RetailerDealer, and WholeSaleDealer), Location, Person (which also consists of the following: Buyer, Seller), Product, ProductCategory (Automobile, Electronics&Computer, and Wears), and TradesOrg. The visual display of the core ontology shown in Figure 3.2.

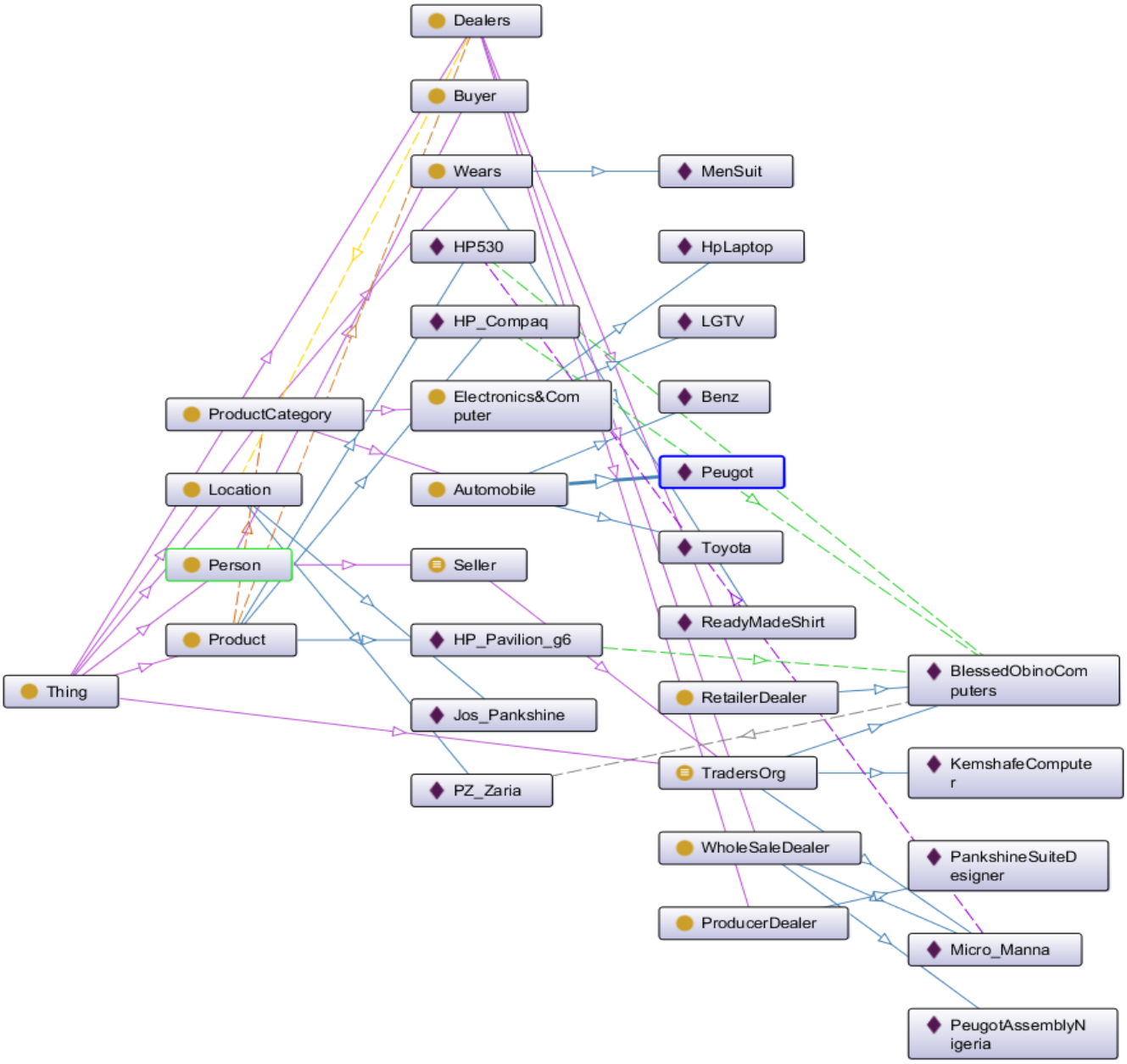


Figure 3.2: Core Ontology – Retailer's Registry

3.3.2 Distributed Retailer Knowledgebase

Figure 3.3 is a complete illustration of all the concepts and relations used to model the retailer ontology.

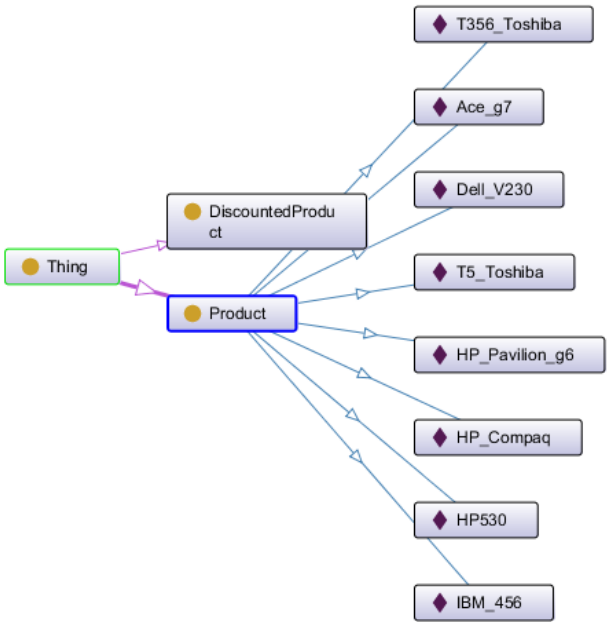


Figure 3.3: Retailer's Ontology

3.3.3 Algorithm of the proposed system

StepI: Let queryString be SWRL+SQWRL constructed

StepII: Connect to server from client side

StepIII: Let S = User search word

Step IV: Array W words = tokenized from users search word

StepV: For each W word as w

If w is in ontology

concatenate w to queryString

Else

Remove from search words S

StepVI: Construct SQWRL query from queryString

StepVII: Execute query

Step VIII: loop through result and display

CHAPTER FOUR

THE IMPLEMENTATION OF THE COMPARISON MODEL AND E-COMMERCE APPLICATION

4.1 Introduction

In this chapter, the model designed to map out the comparison between the two ontologies is presented. Thereafter, the E-Commerce application's implementation is discussed in detail, with the inner working structures clearly outlined for clarity. Lastly, to end this section, the result and discussion section clearly shows the argument of this research, hence contributing to knowledge in the development of semantic based E-Commerce system.

4.2 A Comparison Model for Obiniyi *et al,* (2014) and the New System

To implement the work, thenew model using OWL2 was compared with the model earlier used using OWL by Obiniyi *et al*, (2014). To run the comparison graphical user interface, the NetBeans file Main.java must first of all be run. See Figure 4.1

This model is designed to allow the two ontologies (modeled with OWL1 and OWL2) to be uploaded for onward comparison. The implication of this comparison is to enable same query be executed observe the two ontologies and observed the number of records that can be retrieved from the ontologies each and by so doing, modeling of domain knowledge would be preferred to OWL (or OWL1). Note that henceforth, OWL will be used to refer to OWL1.

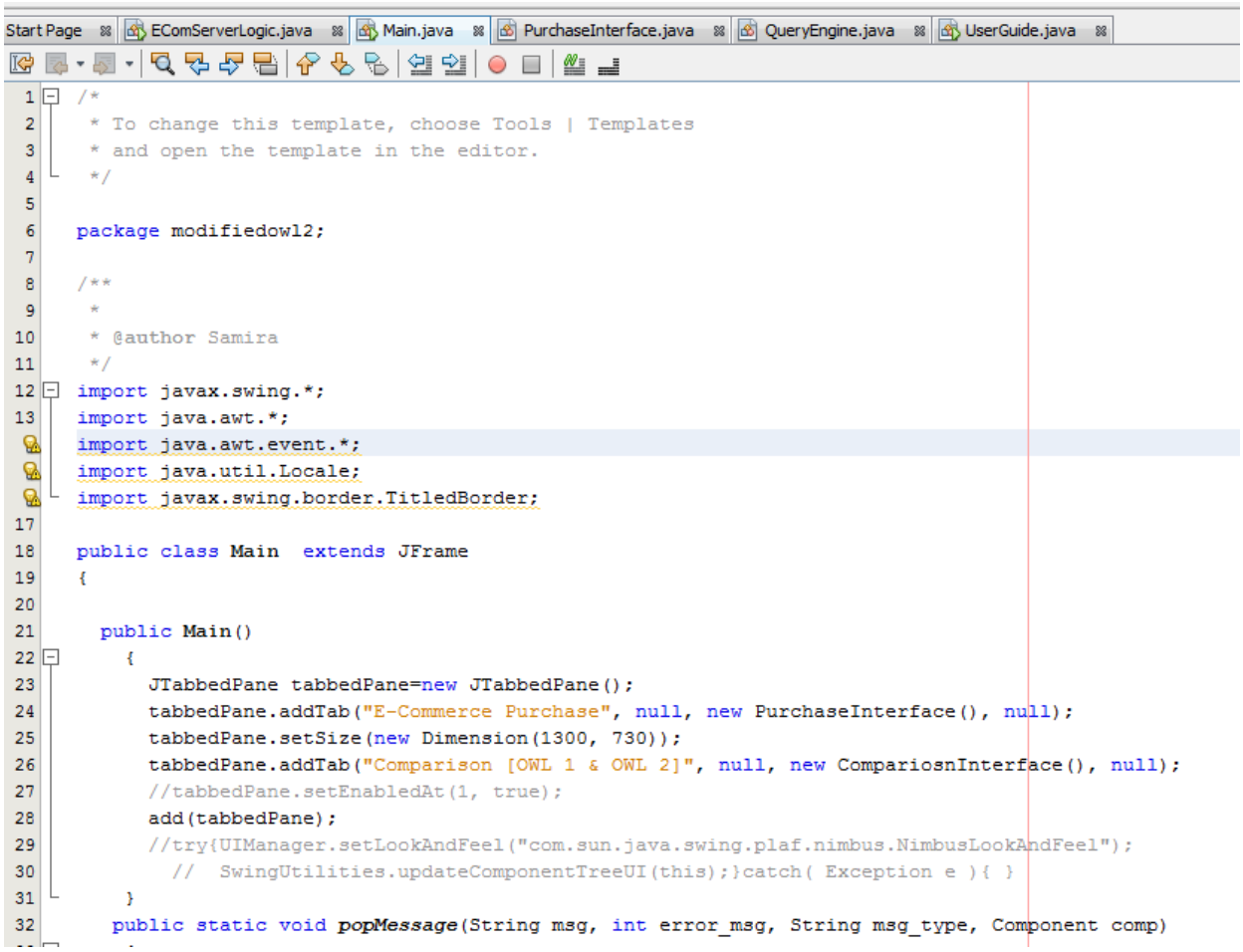


Figure 4.1: Main.java

After launching the Main.java file, the main application interface below appears as seen in Figure 4.2. This figure clearly shows two tabs: one for the comparison model and the other for the E-Commerce application. However, what is captured by Figure 4.2 is the comparisons model. This comparison model is segmented into the visualizer, the ontology loading section, the query entry and parsing section, the variable entry section, and lastly the result display section. Each of these sections are taken in turn and further explained the following subsections.

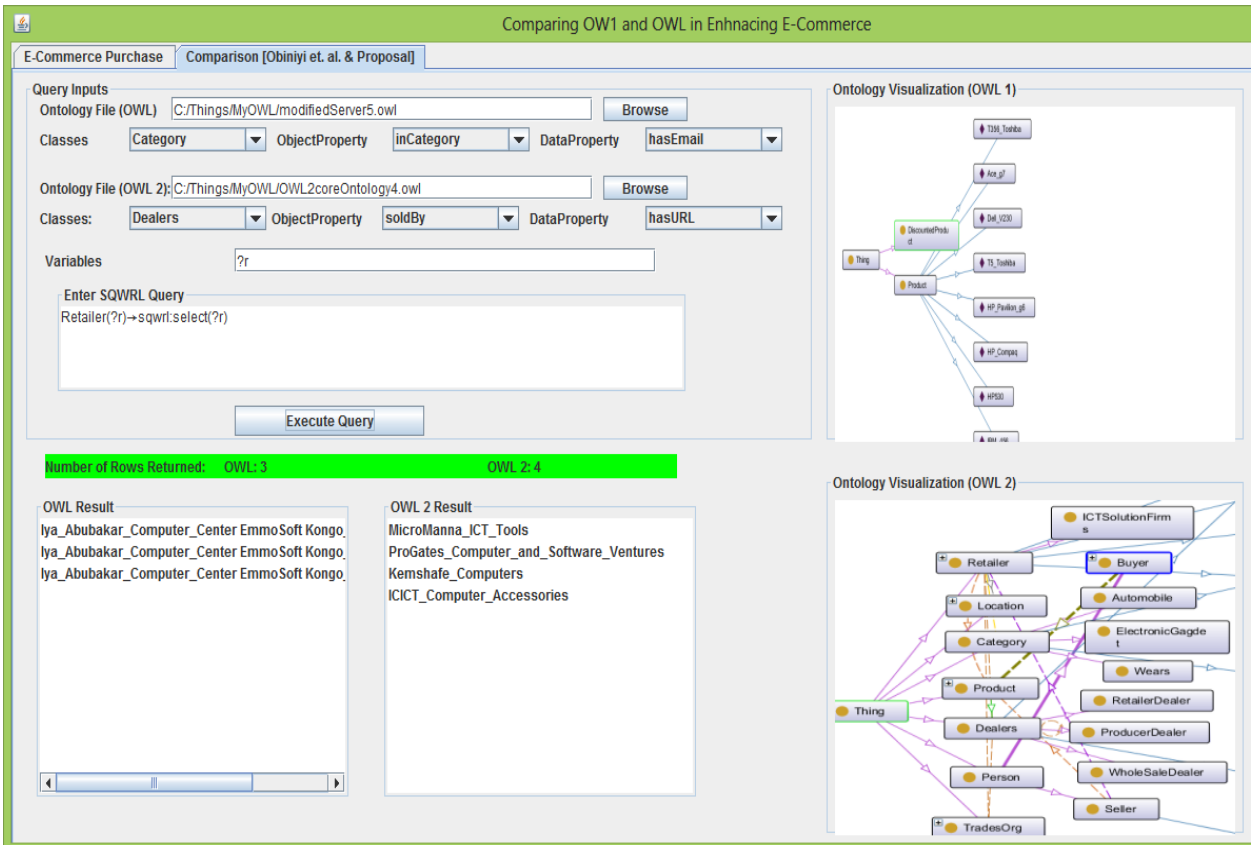


Figure 4.2 A Comparison Interface for Obiniyi *et al*, (2014) and the New System

4.2.1Comparing the Two Models.

Provision is given for loading of the OWL and OWL2 ontology files as seen in Figure 4.3a and 4.3b. Once the ontology files are loaded, the each ontology file components are listed as Classes, as object or data property as the case may be. Figure 4.3a is java code listing of how the classes, object and data properties combo boxes are populated. On the other hand, Figure 4.3b captures the graphical user interface for performing those tasks.

When this is done the user can then go ahead to execute the query on the two ontologies. Before the query execution, a query variable must be inserted, say r, then the query is typed in the "Enter SQWRL Query box" using the correct query syntax. See Figure 4.4a

```
    filePath.setText(dirfile); classNames1.removeAllItems();
    objPredicate1.removeAllItems(); dataPredicate1.removeAllItems();
    for(int i=0; i < qu.enumerateClasses().size(); i++)
        classNames1.addItem(qu.enumerateClasses().elementAt(i));
    for(int i=0; i < qu.enumerateObjectProperty().size(); i++)
       objPredicate1.addItem(qu.enumerateObjectProperty().elementAt(i));
    for(int i=0; i < qu.enumerateDataTypeProperty().size(); i++)
       dataPredicate1.addItem(qu.enumerateDataTypeProperty().elementAt(i));
}
else
 {
    filePath11.setText(dirfile); classNames.removeAllItems();
    objPredicate.removeAllItems(); dataPredicate.removeAllItems();
    qu.setFilewithPellet(owlFile);
    for(int i=0; i < qu.keywords.size(); i++)
       classNames.addItem(qu.keywords.elementAt(i));
    for(int i=0; i < qu.keywords3.size(); i++)
       objPredicate.addItem(qu.keywords3.elementAt(i));
    for(int i=0; i < qu.keywords2.size(); i++)
       dataPredicate.addItem(qu.keywords2.elementAt(i));
```

Figure 4.3a: Ontology Constitutuents Selection for the Two Ontologies

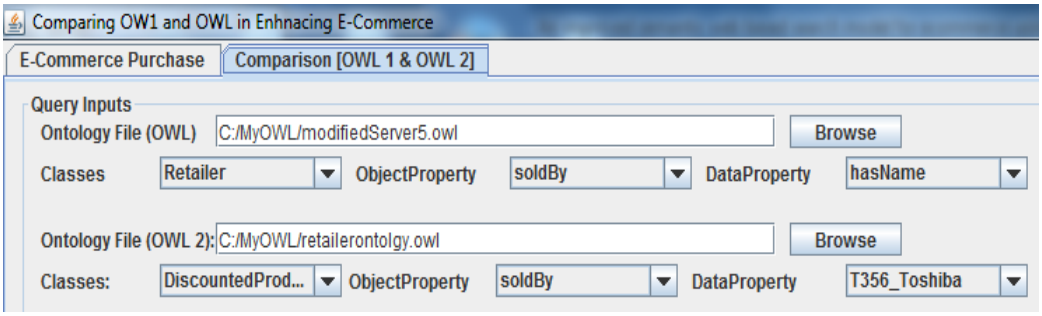


Figure 4.3b: Ontology loading interface

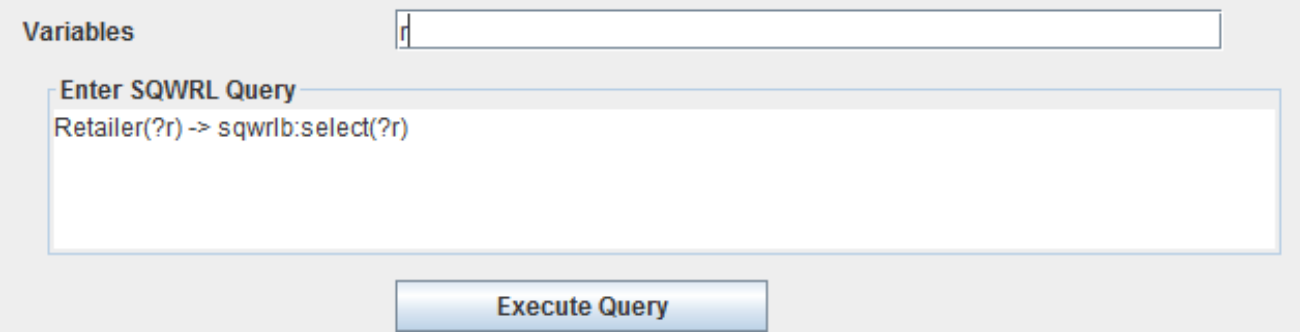


Figure 4.4a: Variable and Query Entry Interface

The variable (*r* for example) is used for storage of result during the retrieval by the query engines. For example, to execute the sample rule in Figure 4.4a, Jess rule engine is suitable for

implementing such query against OWL ontologies. while on the other hand, to implement the same query on the OWL2 ontology, Pellet2.2 reasoner is needful in archiving this. Hence, Figures 4.4b and c captures the process of executing the SWRL rule in Pellet and Jess Rule engines respectively.

```
antecedent.add(factory.getSWRLDataPropertyAtom(hasPrice, var, var1));
antecedent.add(factory.getSWRLDataPropertyAtom(hasQty, var, var2));
antecedent.add(factory.getSWRLObjectPropertyAtom(soldBy, var, var4));
antecedent.add(factory.getSWRLDataPropertyAtom(ownerURL, var4, var5));
//antecedent.add(factory.getSWRLDataPropertyAtom(hasQty, var2, factory
consequences.add(factory.getSWRLClassAtom(clsSelectedProds, var));
consequences.add(factory.getSWRLDataPropertyAtom(hasPrice, var, var1))
consequences.add(factory.getSWRLDataPropertyAtom(hasQty, var, var2));
consequences.add(factory.getSWRLObjectPropertyAtom(soldBy, var, var4))
consequences.add(factory.getSWRLDataPropertyAtom(ownerURL, var4, var5)
```

Figure 4.4b: Pellet Reasoner Style of query with SWRL

```
SQWRLQueryEngine queryEngine =SQWRLQueryEngineFactory.create(ontModel);
SWRLFactory fact = new SWRLFactory(ontModel);
SWRLRuleEngine ruleEngine = SWRLRuleEngineFactory.create(ontModel);
queryEngine.createSQWRLQuery("Query"+type, qry);
SQWRLResult result = queryEngine.runSQWRLQuery("Query"+type);
```

Figure 4.4c: Jess Rule Engine Style of Query with SWRL

The sample SWRL+SQWRL syntax for querying is Retail(?r)->sqwrl:select(?r). The meaning of this SWRL+SQWRL statement is to retrieve all instances or indiviudals of class Retailer, and store them in the variable r. This will then be iterated for listing. Once the query is typed then the “Execute Query” button is clicked. Provision is made for the result of the query to be displayed in the two windows below, see Figure 4.5

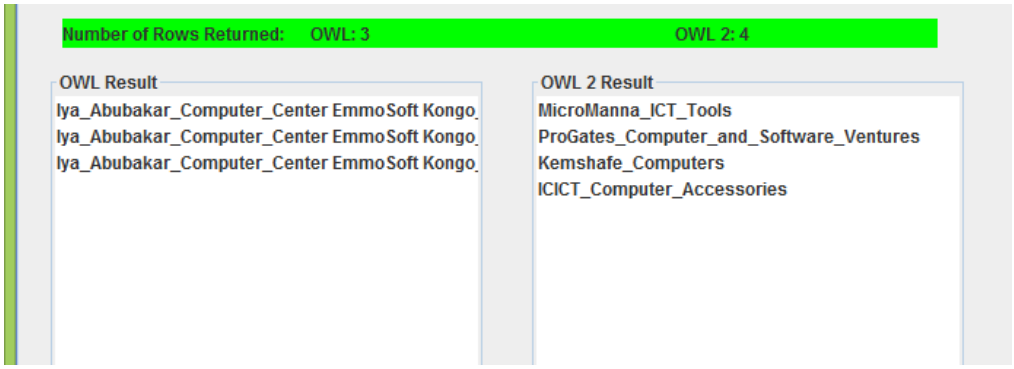


Figure 4.5 Results of the Queries Displayed

4.3The E-Commerce Application.

Again developing an e-commerce application was one of our principal objective in the entire work. This was achieve also here. Running the Main.java file launches the application for the e-commerce as well, see e-commerce graphical user interface below in Figure 4.6

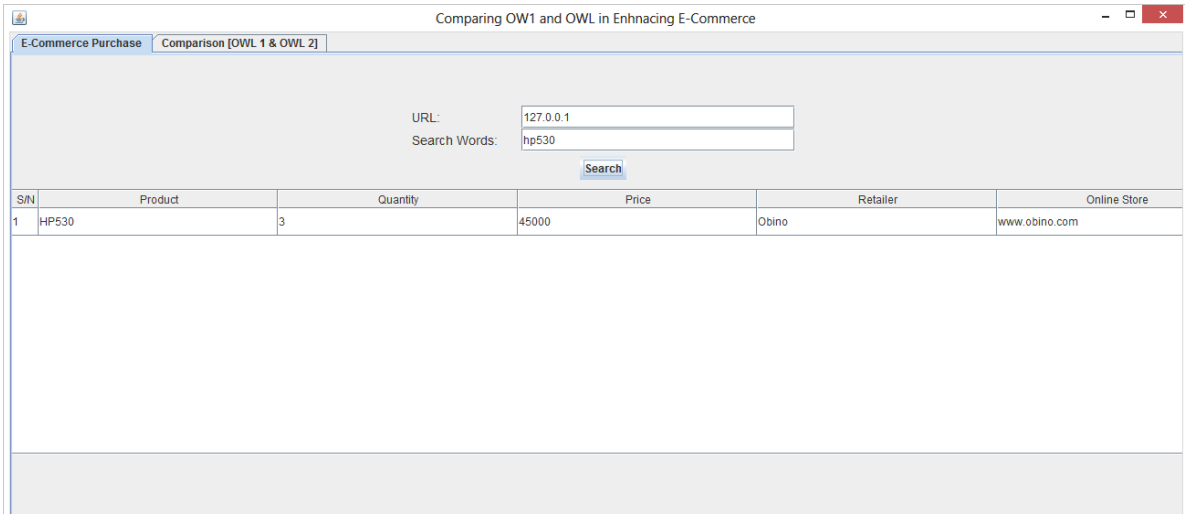


Figure 4.6 The E-Commerce Graphical User Interface.

Before using the e-commerce application for a product search, the server hosting the core ontologies for the each registered retailer must be put on, so a simulation of this was done here we have to run the EComServerLogic.java and PurchaseInterface.java file, see Figure 4.7a and b. Though just some snippets are captured and shown here,

```
bufRead=new BufferedReader(new InputStreamReader(clientCon.getInputStream())); Sys
searchWord=bufRead.readLine(); System.out.println("server 3...");
System.out.println("searchWord: "+searchWord);
String OWLfile="C:/MyOWL/retailerontolgy.owl";
QueryEngine qu=new QueryEngine();
UserGuide ug=new UserGuide(searchWord, OWLfile, qu);
String s=ug.formatOutput(ug.rows); System.out.println("server 4...");
bufWrite=new BufferedWriter(new OutputStreamWriter(clientCon.getOutputStream()));
bufWrite.write(s);
bufWrite.flush();System.out.println("server 5...");

bufWrite=new BufferedWriter(new OutputStreamWriter(clientCon.getOutputStream()));
ready4Com=false;
```

Figure 4.7a The server side querying the user needs on retailer ontology.

```
socket=new Socket(ip, 8888);
writeMe= new BufferedWriter( new OutputStreamWriter(socket.getOutputStream()));

System.out.println("Sending to retrieve from ontology...");
writeMe.write(searchWord); //"HP530\n"
writeMe.flush();

System.out.println(searchWord);
readMe= new BufferedReader( new InputStreamReader(socket.getInputStream()));
answer=readMe.readLine(); System.out.println("REPORT RETRIEVED: "+answer);
updateTable(answer);
System.out.println(answer);
writeMe.close();
readMe.close();
```

Figure 4.7b The client side sending user query to server side.

When this file is run, it pops up this dialogue box from the two taps one can start the server or stop the server. So before running the e-commerce application the server must be started in order to allow the search on them. On the search interface in Figure 4.8 the URL of the server we intent to conduct the search must be inserted, in this case 127.0.0.1 after which a search word can be inserted by the user say hp530 when done, the search button can then be clicked. See Figure 4.8

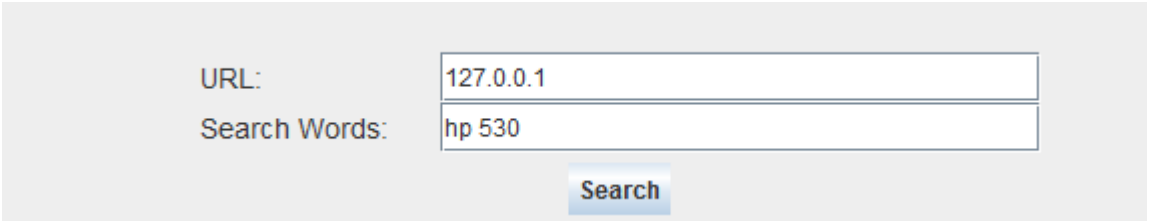


Figure 4.8 Entering Server URL and Search Word

The result of the product search is displayed as shown below, in Figure 4.9

S/N	Product	Price	Quantity	Retailer	Online Store
1	HP401	3	45000	Obino	www.obino.com
2	HP401	3	45000	Obino	www.obino.com

Figure 4.9 Result of Product Search on the E-commerce Application.

4.4 Results and Discussion

Few queries are generated and tested against the similar OWL and OWL2 ontologies so as to observe and compare the number of search the could make. Tables 4.1 and 4.2 give some comparison figures observed in this research.

Query ID	Query	Description
Query1	$Retailer(?r) \rightarrow sqwrl:select(?r)$	*Selects all available retailers from the ontology*
Query2	$Retailer(?r),\ ProductCategory(?p),\ belongsTo(?r, ?p) \rightarrow sqwrl:select(?r)$	*Selects retailers who belongs to a related group product dealers*
Query3	$Retailer(?r),\ soldBy(HP530,\ ?r) \rightarrow sqwrl:select(?r)$	*Selects retailers who sells a particular product*
Query4	$Retailer(?r),\ Location(?l),\ locatedAt(?r, ?l) \rightarrow sqwrl:select(?r)$	*How retailers are located in a given location*
Query5	$Retailer(?r),\ Seller(?s),\ sellsFor(?s, ?r) \rightarrow sqwrl:select(?r)$	*Does a retailer have known sellers?*

Table 4.1: Generated queries and their descriptions

Query ID	Obiniyi *et. al.* (2014)	New System
Query1	3	4
Query2	1	2
Query3	1	1
Query4	3	4
Query5	0	3

Table 4.2: Generated queries and their performances on two differently modeled ontologies

The graph shown by Figure 4.9 is a graphical illustration of how an E-Commerce application will greatly improve its product search should the knowledgebase be developed in an expressive way.



Figure 4.9: A Graphical Illustration of the Efficiency the New System over Obiniyi *et al,* (2014)

Objective number three of this work is to enhance the efficiency of B2C by implementing the Ontology using OWL2 rather than OWL saw the design of the ontology in chapter three and implements how these ontologies are being accessed. The implementation of this application was designed in such a way that when the user search request is being translated to SWRL, a corresponding SQWRL query is generated for the OWL1ontology. These two corresponding queries are then run to observe their various outputs and then evaluated if truly using SQWRL to query OWL2 file used to model information in B2C environment, is more efficient than SQWRL to query and ontology using OWL1. The two queries are channeled to different files

and then compared. Figure 4.4 captures the result after the corresponding query was run against the different ontologies.

Looking closely at Figure 4.5, SQWRL with OWL1 returns three results while SQWRL with OWL2 ontology returns four rows of result. It obvious that SQWRL querying OWL2 ontology is more efficient than SQWRL when used against OWL with such constructs described in chapter two.

CHAPTER FIVE:

SUMMARY, CONCLUSION AND RECOMMENDATION

5.1 Summary

In this concluding chapter, I close the work done in the course of the research. Justification was given to the choice of the different Semantic Web technologies that were used. For further research work, I also present the limitations we came across. Finally, the means or methods we drew to realize our objective was summarized

The second chapter of the dissertation presented the various Semantic Web technologies available that were considered. Some of which were RDF/RDFS, OWL,OWL2 and XML (these are used for ontology implementation) and SQWRL query language. SWRL rule language was similarly used as a means of deriving more facts in the underlying ontology. In the course of realizing objectives of thedissertation, SQWRL was used as the query language to be used against OWL1 and OWL2 – it was also adapted as the desired or more efficient ontology language to enhance reasoning due to the availability of constructs that make reasoning more easily done. When these Semantic Web technologies were used, it became clear that accessing information from an OWL2 based ontology using SQWRL as a query language results in more hits than using SQWRL on OWL. In the same manner, it was pointed out that a more inference supporting ontology can be developed when using OWL2 than using OWL1.

The main contribution of this dissertation is theimprovement of the results or information that would be retrieved from a B2C (an aspect of E-commerce) Web applications or databases only if a better data modeling language as well as when a better query language is chosen.

5.2 Conclusion

To conclude, I have shown that OWL2 constructs that has better expressive and intuitive power proves OWL2 to be more expressive than using OWL or RDF/RDFS to model or create ontologies. In a similar vein, we saw that SQWRL is more efficient when used with OWL2 than when used with OWL- especially when some more powerful OWL2 constructs are used. We showed that B2C search will be better if the aforementioned technologies are used.Finally an ontological Ecommerce application was developed to show the use of these Semantic Web technologies.

Semantic web technology provides different ontology languages which vary in their features. The difference in each ontology language has their advantages and disadvantage couple with their usage with respect to query language used. This dissertation shows the limitation of OWL as an ontology language. It is clear based on the variability of features of these ontology languages that modeling of data or information in a domain will be affected by their available syntax and vocabulary. Furthermore, querying or retrieval of information from this modeled data could be realized using SQWRL. The choice of these query languages with respect to the ontology language used, will improve the output of the query. Therefore, a major contribution to knowledge of this dissertation is the enhancement of the possible number of results or information that would be retrieved from a B2C (an aspect of E-commerce).Based on the sample query we ran on the comparison application, results show that the new model gave more results from table 4.2 in section 4, Results from query1 showed the new model with 4

results while the old had 3, query 5 results showed the new model with 3 results while the old with 0.

5.3 Recommendations

In the future, researchers seeking to improve this work may first consider to deploy it as a web application. This will make it convenient to call it a semantic web application rather than an ontological application. Secondly, few rules were written to demonstrate the workability of the Ecommerce application. An improvement on this wok might be to add more rules so as to promote autonomous environment. Considering the space of time, the ontology was developed just for a particular domain of Ecommerce (Computer Accessories). To further a wide range of search, other domain may be incorporated. The work just focuses on search for products, we recommend that the next line of work should go beyond just searching of products to buying as well where an online web payment platform is created.

REFERENCES

Antoniou, G., and Frank V. H. (2004). *Semantic Web Primer*, MIT Press, ambridge.

Barter System History (BSH) (2016): The Past and Present . Retrievedon 10-5-2016 from
https://www.mint.com/barter-system-history-the-past-and-present#sthash.CRhHQPY%20c.dpuf

Berners-Lee, T. (1994). *Weaving the Web*, The MIT Press, Cambridge

Berners-Lee, T., Hendler, J. and Lissila, O. (2001. *The Semantic Web. [Online]*. Retrievedon 10-5-2016from http://sciam.com/article.cfm?articleID=00048144- 10D2-1C70-84A9809EC588EF21.

Bizer, C., F., and Heath, T., T., Berners-Lee, T., (2009). Linked Data- the Story so Far,*International Journal on Semantic Web and Information System (IJSWIS,)* Vol. 5(3), pp. 1-23

Cuenca B., Horrocks I., Motik B., Parsia B., Patel-Schneider P., and Sattler U (2008).*OWL 2: The Next Step for OWL*

Dhanapal C.S, Mahesh M. P, Sivanantham S. and Premakumari M.(2004), *Commerce Higher Secondary - First Year*.Tamilnadu Text Book Corporation College Road, Chennai-600 006 Retrievd on 12-5-2016 from http://docplayer.net/8312352-Commerce-higher-secondary-first-year-untouchability-is-a-sin-untouchability-is-a-crime-untouchability-is-inhuman-tamilnadu-text-book-corporation.html

Frauenfelder, M. (2001). *A smarter Web*. [Online]. Retrieved on 13-5-2016 from http://www.technologyreview.com/articles/frauenfelder1101.asp.

Freitas, A., Curry E., and Oliveira, G. J., and O'Riain, S., (2012). *Querying Heterogeneous Datasets on the Web: Challenges, Approaches and Trends*, IEEE Internet Computing, Vol. 16, No.3, pp 24-33.

Gruber, T., (1993). A Translation Approach to Portable Ontologies, *Knowledge Acquisition, 5(* 2), pp. 199-220.

Hart G., DolbearC. (2013). *Linked Data: A Geographic Perspective,* CRC Press, Boca Raton.

Jayaprabha P, and Saradha A (2010),*Semantic Integration Of Heterogeneous Web Data For Tourism Domain Using Ontology Based Resource Description Language*. Journal of Computer Applications, 3(3),pp4-5

Kerie, A. (2014). *Engineering Semantic Web for E-Commerce Business Intelligence: ABilingual EEPS Ontology Model*. A master dissertation submitted to the school of Graduate studies of Arba Minch University.

Kück (2004). *Tim Berners-Lee's Semantic Web.* South African Journal of Information Vol.6(1) March 2004

Lewis, D., (2008). *Intelligent agents and the Semantic Web:* Developing an intelligent Web, *IBM Cooperation.*

Obiniyi A.A, Oyelade O.N and Junaidu S.B (2014). *Enhancing Reasoning and RetrievalPerformance on OWL using SQWRL*.International Journal of Engineering Research & Technology (IJERT), Vol. 3 Issue 2, February.

O'Connor, M., Knublauch, H., Tu, S., Grosof, B., Dean, M., Grosso W., and Musen, M.,(2005).*Supporting Rule System Interoperability on the Semantic Web with SWRL*. Retrieved from www.mit.edu/~bgrosof/paps/**swrl**-editor-swc2005.pdf*on* 25th July,2014.

Oyelade O., Junaidu S., and ObiniyiA.(2014).*Semantic Web Framework for E-Commerce Based on OWL*. IJCSI International Journal of Computer Science Issues, Vol. 11, Issue 3, No 2, May 2014 ISSN (Print): 1694-0814 | ISSN (Online): 1694-0784.www.IJCSI.org

OWL 2 Web Ontology Language Document Overview (Second Edition).Retrieved on 16-6-2016 from https://www.w3.org/2007/OWL/wiki/Document_Overview.

OWL 2 Web Ontology Language New Features and Rationale (Second Edition) W3CRecommendation 11 December 2012.(WOL)(2016): Retrieved on 17-6-2016fromhttps://www.w3.org/TR/owl2-new-features/

Peter B (2015). *The Future of Post-Human Meta-Data: Towards a New Theory of Structure and Process in information science.*Prim rose Hall Publication GroupBrisbane Australia. Retrieved on 17-6-2016 from https://books.google.com.ng/books?id=ES7NCQAAQBAJ&printsec=frontcover#v=onepage&q&f=false.

Pomponio L. and Viale P.(2013), *Semantic Web in Action*- Consumer Experience Studio Rosario, Argentina December 4, 2013 Retrieved on 16-6-2016 from https://www.globant.com/sites/default/files/semantic_web_in_action_0.pdf

Russell S. and NorvigP. (2003). *Artificial Intelligent: A Modern Approach.* Pearson Education, Inc. New Jersey.

Semantic Web.(SW)(2016):Retrieved on 16-6-2016 from http://www.businessdictionary.com/definition/semantic-web.html

Shadbolt, N., Hall, W. and Tim B. (2006). *The Semantic Web Revisited*, IEEE IntelligentSystems, 21(3),pp 96-101

Smith D. M., (2002). *The Impact of Shopbots on Electronic Markets*.*Journal of the Academy of Marketing Science*, Fall 2002, *Vol. 30(*4), pp. 442-450Text Book Corporation College Road, Chennai-600 006.

Tang Z.,Smith M., and Montgomery A,(2007).*The Impact of Shopbot Use on Prices and Prices Dispersion*: Evidence from Online Book Retailing. Heinz College Research. Retrieved on 16-6- 2016 fromhttp://ssrn.com/abstract=989084.

Yao W., Basu S., Li J., and Stephenson B. (2012).*Modeling and Configuration of Process Variants for On-boarding Customers to ITOutsourcing*. HP Laboratories HPL- 2012-157, Hewlett-Packard Development Company, L.P.

Web 1.0 vs Web 2.0 vs Web 3.0 vs Web 4.0 vs Web 5.0(WEB1.0)(2016): – A bird's eye on the evolution and definitionRetrieved on 16-6-2016from https://flatworldbusiness.wordpress.com/flat-education/previously/web-1-0-vs-web-2- 0-vs-web-3-0-a-bird-eye-on-the-definition/

What are Semantic Computing and Semantic Web? Retrieved on 16-6-2016 from https://www.freelancer.com/community/articles/semantic-web-using-html5

Willaert, F., (2001).*XML-Based Frameworks and Standards for B2B Ecommerce*, Unpublished Dissertation of Katholieke Universiteit Leuven

Wooldridge M., Jennings N., and KinnyD.(1999).*The Gaia Methodology for Agent-Oriented Analysis and Design*, Autonomous Agents and Multi-Agent Systems 3(3), pp.285-312

Zhang L. and Huang W. (2009). *A Framework for an Ontology-based E-commerce Product Information Retrieval System.* Journal Of Computers, Academy Publisher 4(6), pp 439-440

APPENDIXES A: Ontology File

Core ontology: retailer.owl

Prefix(owl:=<http://www.w3.org/2002/07/owl#>)
Prefix(rdf:=<http://www.w3.org/1999/02/22-rdf-syntax-ns#>)
Prefix(xml:=<http://www.w3.org/XML/1998/namespace>)
Prefix(xsd:=<http://www.w3.org/2001/XMLSchema#>)
Prefix(rdfs:=<http://www.w3.org/2000/01/rdf-schema#>)
Ontology(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5>
Declaration(Class(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#DiscountedProduct>))
Declaration(Class(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#Product>))
Declaration(Class(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#Retailer>))
Declaration(Class(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#SelectedProducts>))
Declaration(ObjectProperty(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#soldBy>))
ObjectPropertyDomain(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#soldBy><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#Product>)
ObjectPropertyDomain(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#soldBy><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#SelectedProducts>)
ObjectPropertyRange(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#soldBy><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#Retailer>)
Declaration(ObjectProperty(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#testing>))
Declaration(DataProperty(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#expiryDate>))
DataPropertyDomain(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#expiryDate><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#Product>)
DataPropertyRange(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#expiryDate> xsd:dateTime)
Declaration(DataProperty(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#hasId>))
Declaration(DataProperty(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#numAvailable>))
SubDataPropertyOf(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#numAvailable> owl:topDataProperty)
DataPropertyDomain(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#numAvailable><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#Product>)
DataPropertyDomain(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#numAvailable><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#SelectedProducts>)
DataPropertyRange(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#numAvailable> xsd:int)
Declaration(DataProperty(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#percentageDiscount>))
DataPropertyDomain(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#percentageDiscount><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#DiscountedProduct>)

DataPropertyRange(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#percentageDiscount> xsd:int)
Declaration(DataProperty(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#retailerURL>))
DataPropertyDomain(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#retailerURL><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#Retailer>)
DataPropertyRange(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#retailerURL> xsd:string)
Declaration(DataProperty(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#soldFor>))
DataPropertyDomain(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#soldFor><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#Product>)
DataPropertyDomain(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#soldFor><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#SelectedProducts>)
DataPropertyRange(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#soldFor> xsd:decimal)
Declaration(NamedIndividual(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#Ace_g7>))
ClassAssertion(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#Product><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#Ace_g7>)
ObjectPropertyAssertion(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#soldBy><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#Ace_g7><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#MicroManner_Computers>)
DataPropertyAssertion(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#numAvailable><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#Ace_g7> "5"^^xsd:int)
DataPropertyAssertion(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#soldFor><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#Ace_g7> "67800"^^xsd:decimal)
Declaration(NamedIndividual(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#CBR_Computers>))
ClassAssertion(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#Retailer><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#CBR_Computers>)
DataPropertyAssertion(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#retailerURL><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#CBR_Computers> "www.cbronlineshop.com")
Declaration(NamedIndividual(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#Dell_V230>))
ClassAssertion(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#Product><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#Dell_V230>)
ObjectPropertyAssertion(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#soldBy><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#Dell_V230><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#MicroManner_Computers>)
DataPropertyAssertion(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#numAvailable><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#Dell_V230> "9"^^xsd:int)
DataPropertyAssertion(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#soldFor><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#Dell_V230> "67000"^^xsd:decimal)

Declaration(NamedIndividual(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#HP530>))
ClassAssertion(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#Product><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#HP530>)
ObjectPropertyAssertion(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#soldBy><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#HP530><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#MicroManner_Computers>)
DataPropertyAssertion(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#numAvailable><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#HP530> "6"^^xsd:int)
DataPropertyAssertion(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#soldFor><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#HP530> "90000"^^xsd:decimal)
Declaration(NamedIndividual(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#HP_Compaq>))
ClassAssertion(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#Product><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#HP_Compaq>)
ObjectPropertyAssertion(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#soldBy><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#HP_Compaq><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#MicroManner_Computers>)
DataPropertyAssertion(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#numAvailable><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#HP_Compaq> "11"^^xsd:int)
DataPropertyAssertion(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#soldFor><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#HP_Compaq> "78900"^^xsd:decimal)
Declaration(NamedIndividual(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#HP_Pavilion_g6>))
ClassAssertion(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#Product><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#HP_Pavilion_g6>)
ObjectPropertyAssertion(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#soldBy><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#HP_Pavilion_g6><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#MicroManner_Computers>)
DataPropertyAssertion(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#numAvailable><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#HP_Pavilion_g6> "3"^^xsd:int)
DataPropertyAssertion(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#soldFor><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#HP_Pavilion_g6> "89700"^^xsd:decimal)
Declaration(NamedIndividual(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#IBM_456>))
ClassAssertion(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#Product><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#IBM_456>)
ObjectPropertyAssertion(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#soldBy><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#IBM_456><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#CBR_Computers>)

DataPropertyAssertion(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#numAvailable><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#IBM_456> "2"^^xsd:int)
DataPropertyAssertion(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#soldFor><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#IBM_456> "156000"^^xsd:decimal)
Declaration(NamedIndividual(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#MicroManner_Computers>))
ClassAssertion(<http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#Retailer><http://www.semanticweb.org/samira/ontologies/2015/10/untitled-ontology-5#MicroManner_Computers>)