University of Bradford eThesis

A Framework for Semantic Web Implementation based on Context-Oriented Controlled Automatic Annotation

Muna Salman HATEM

submitted for the degree
of Doctor of Philosophy

Department of Computer Science

University of Bradford

2009

Abstract

The Semantic Web is the vision of the future Web. Its aim is to enable machines to process Web documents in a way that makes it possible for the computer software to "understand" the meaning of the document contents. Each document on the Semantic Web is to be enriched with meta-data that express the semantics of its contents. Many infrastructures, technologies and standards have been developed and have proven their theoretical use for the Semantic Web, yet very few applications have been created. Most of the current Semantic Web applications were developed for research purposes.

This project investigates the major factors restricting the wide spread of Semantic Web applications. We identify the two most important requirements for a successful implementation as the **automatic** production of the semantically **annotated document**, and the creation and maintenance of **semantic based knowledge base.**

This research proposes a framework for Semantic Web implementation based on context-oriented controlled automatic Annotation; for short, we called the framework the Semantic Web Implementation Framework **(SWIF)** and the system that implements this framework the Semantic Web Implementation System (**SWIS**). The proposed architecture provides for a Semantic Web implementation of stand-alone websites that automatically annotates Web pages before being uploaded to the Intranet or Internet, and maintains persistent storage of Resource Description Framework (RDF) data for both the domain memory, denoted by Control Knowledge, and the meta-data of the Web site's pages. We believe that the presented implementation of the major parts of SWIS introduce a competitive system with current state of art Annotation tools and knowledge management systems; this is because it handles input documents in the

context in which they are created in addition to the automatic learning and verification of knowledge using only the available computerized corporate databases.

In this work, we introduce the concept of **Control Knowledge (CK)** that represents the application's domain memory and use it to verify the extracted knowledge. **Learning** is based on the number of occurrences of the same piece of information in different documents. We introduce the concept of **Verifiability** in the context of Annotation by comparing the extracted text's meaning with the information in the CK and the use of the proposed database table **Verifiability_Tab**.

We use the linguistic concept **Thematic Role** in investigating and identifying the correct meaning of words in text documents, this helps correct relation extraction. The verb lexicon used contains the argument structure of each verb together with the thematic structure of the arguments.

We also introduce a new method to chunk conjoined statements and identify the missing subject of the produced clauses. We use the **semantic class of verbs** that relates a list of verbs to a single property in the ontology, which helps in disambiguating the verb in the input text to enable better information extraction and Annotation.

Consequently we propose the following definition for the annotated document or what is sometimes called the **"Intelligent Document"**

"The Intelligent Document is the document that clearly expresses its syntax
and semantics for human use and software automation".

This work introduces a promising improvement to the quality of the automatically generated annotated document and the quality of the automatically extracted information in the knowledge base. Our approach in the area of using Semantic Web

technology opens new opportunities for diverse areas of applications. E-Learning applications can be greatly improved and become more effective.

Acknowledgement

I am grateful to many people who helped me with their comments during the past four years. I would like to especially thank my supervisor Daniel Neagu and my external supervisor Haider Ramadan for their valuable guidance, support and encouragement.

I would also like to thank Alaa Al-Mussali, Ahmed Al-Rawi, Adel Abu-Radwan and James Moody for their advice and comments on the linguistic concepts used in this thesis.

Many thanks to Sultan Qaboos University for providing the external supervisor. Thanks to my colleagues at the Department of Computer Science. Special thanks to Professor Khaled Day, Dr. Swamy Kutti, and Professor Ali Yousif, the Dean of the College of Science for their continuous support and encouragement.

Above all, I am grateful to God for blessing me with good health to complete this research and for every thing that has been accomplished in this work.

Publications

Journal Papers

1. Hatem, M., Neagu, D., Ramadan, H.(2006) "Towards personalization and a Unique Uniform Resource Identifier for Semantic Web Users with in an Academic Environment", *The Journal of Instructional Technology and Distance learning*, Vol. 3. No. 6. ISSN 1550-6908 ed. by Donald G. Perrin, June 2006, USA.

2. Al-Khanjari, Z.A., Kutti, N.S., Hatem, M. (2006) "An Extended E-Learning Architecture: Integrating Software Tools Within The E-Learning Portal", *The International Arab Journal for Information Technology (IAJIT),* 3(1):75-81, ISSN: 1683-3198, ed by Abuelrub, E., Alhadithi, J., Zarqa Private University, January, 2006,Amman, Jordan.

3. Hatem, M., Ramadan, H., Neagu, D. (2005) "e-Learning based on Context Oriented Semantic Web", *Journal of Computer Science*, 1(4): 499-503, ISSN 1549-3636, Science Publications, 2005, New York, USA.

4. Ramadan, H., Hatem, M., Al-Khanjri, Z., Kutti, S.(2005) "A Classification of Techniques for Web Usage Analysis", *Journal of Computer Science*, ISSN: 1549-3636, 1(3): 413-418, Science Publications, 2005, New York, USA

Conference Papers

5. Hatem, M., Neagu, D., Ramadan, H. (2008) "RDF Repository of Experts based on Context Oriented Automatic Annotation Framework", *Second Asia International Conference on Modelling & Simulation AMS2008*. Kuala Lumpur, Malaysia 13 – 15 May 2008. Proceeding (ed. David Al-Dabass) IEEE Computer Society: ISBN: 978-0-7695-3136-6, pp. 29-34

6. Hatem, M., Neagu, D., Ramadan, H.,(2006) "Towards personalization and a Unique Uniform Resource Identifier for Semantic Web Users", *9th International Conference on Interactive Computer aided Learning ICL2006*, Villach, Austria, September 2006. Proceedings (ed. Michale E. Aure) KasseI university press: ISBN 3-89958-195-4

7. Hatem, M., Ramadan, H., Neagu, D. (2005)" e-Learning based on Context Oriented Semantic Web", *The first International Conference on E-Business and E-Learning EBL05* , Princess Sumaya University for Technology, May 23-24 2005 Amman, Jordan. Proceedings (ed. by Chapelet. P, Awajan, A.) Princess Sumaya University for Technology: ISBN 9957-8585-0-0, pp 51-56

8. Ramadan, H., Hatem, M. (2005), "A Classification of Techniques for Web Usage Analysis", *The first International Conference on E-Business and E-Learning*, Princess Sumaya University for Technology, May 23-24, 2005 Amman, Jordan. Proceedings (ed. by Chapelet. P, Awajan, A.) Princess Sumaya University for Technology: ISBN 9957-8585-0-0, pp 210-215

9. Hatem, M., Daniel Neagu, Ramadan, H. (2005) "Context Oriented RDF Repository for Semantic Web: Application to Sultan Qaboos University Web Services", *The 6th Informatics Workshop for Research Students*, University of Bradford, UK, March 2005. Proceedings (ed. by Rigas,D.) University of Bradford: ISBN 1851432205, pp 67-70

Contents

List of Figures

List of Tables

To my supportive family

To my son Alameen for being such understanding and loving child.

To my granddaughter Wejd for filling my life with joy and happiness again.

"Now, miraculously, we have the Web. For the documents in our lives, everything is simple and smooth. But for data, we are still pre- Web."

Tim Berners-Lee, Business Model for the Semantic Web[1]

---

[1] Business Model for the Semantic Web, http://www.w3.org/DesignIssues/Business

1. The Semantic Web

1.1 Introduction

The success of the World Wide Web (WWW) has made it the source of choice for information, and the entry point to business transactions. The amount of information on the Web is growing so fast that it has become increasingly difficult to locate the required information without having to spend a considerable amount of time searching for the relevant pieces of information. The number of Web pages and sites on the Internet is an important indicator of the size and the growth of the Internet. This growth has been studied by the Online Computer Library Corporation (OCLC) [84] and Netcraft [82]. OCLC estimated the number of websites on the Internet as 4,882,000 in June 1999 which included nearly 300 million pages and over 500 million files, while Netcraft estimated the number as 6,177,453 with nearly 800 million pages. This makes it obvious that the discrepancies in the numbers are due to the different definitions and methodologies used [71]. Netcraft estimated the number of websites in 2005 as 70.3 million website and this number increased to 158.2 million in February 2008 with growth of 2.6 million sites compared to January 2008. These statistics mean that the number of websites on the Internet is increasing rapidly and the Web is becoming big to the extent that current technologies can not cope with it; the use of keyword search is becoming inapt to cope with the users' demands. In response to the problems with the current Web, new methods and techniques are required. The Semantic Web road map was introduced by Tim Berners-Lee as the new vision of the Web [17] where information is expressed in a machine-understandable form. The Semantic Web is seen

by many researchers and commercial companies as the "Future Web" or the "Third Generation Web". Currently, the focus of Internet research and knowledge management research are targeted towards creating a global decentralized knowledge based system. Many different definitions have been proposed to describe the Semantic Web; we prefer the following as it is related to the objective of this work:

> *"The Semantic Web is a vision for a next-generation network that lets content publishers provide notations designed to express a crude meaning of the page instead of merely dumping arbitrary text onto a page. Autonomous Agent software can then use this information to organize and filter data to meet the user's needs"* [83].

Hence, one of the basic requirements of the Semantic Web is adding meta-data to Web pages; this meta-data is the machine-understandable semantics of the Web page contents.

Although the Semantic Web is attracting researchers' attention to solve the difficulties and problems of the current Web, there is not even one Semantic Web implementation currently available that fulfils the Semantic Web vision. The reason is the Knowledge Acquisition problem [98].

This research is directed towards solving some Semantic Web implementation problems. The implementation of Semantic Web application of an academic institution: namely Sultan Qaboos University (SQU), is used as a case study to demonstrate this work.

1.2 What is Expected from the Semantic Web

Current search engines are based on the traditional syntactic exploration of Web contents, there are obviously many significant problems with the current Web: search engines seem to be ineffective in collating information that provides a high degree of relevance, the use of different synonyms to represent the same idea excludes important Web pages from the search results, and many words have multiple meanings that causes an unmanageable number of irrelevant hits in search results [53, 87]. Semantic Web is not only needed to solve the current problems with today's Web, it is expected to provide for new facilities that are not feasible with the current Web; some of the expected facilities can be briefly presented as follows:

Information Retrieval

The documents retrieved with Semantic Web will not only include those documents that are relevant to the keywords used in the search query, but will also retrieve documents that use terms with the same meaning or that belong to a certain class, subclass of concept.

Information Extraction and Software Agents

Software agents are expected to play a key role with semantic retrieval. Web documents on the Semantic Web contain extra information that enables software agents to extract relevant information from the retrieved Web pages. The user of the Semantic Web does not browse retrieved documents searching for relevant information; extracted information is presented to the user enabling quick access to the required information.

Maintenance of Web Documents

One of the most demanding, tedious and time consuming activities with the current Web is keeping Web documents updated. The current status of many websites shows clear inabilities of website owners to keep their sites updated. The Semantic Web is expected to enable automatic generation of updated documents independent of the methodology used in storing and manipulating the data of the website.

At present the number of dynamically generated Web pages from databases is larger than the number of static pages [52]; this is true for many important sites and it will continue to be the case because data is currently handled in an efficient way in databases. Annotation such as the deep Annotation suggested by [52] may play a role in providing Annotation for the dynamically generated Web pages from current databases, but we expect that future implementations will make the operational data based on semantic representation and handling, such as the Jena [67] implementation backed by persistent storage like Oracle data base. Such an implementation is expected to provide for both static and dynamic Web pages and enables dynamic update for both types of Web pages. This will include pages which start static and end dynamic.

Adaptive Websites

Currently, data about user interests and intention are collected directly from user supplied information or from past user usage of the Internet like the user's previous searches, the time spent on a site, the links visited, etc. On the Semantic Web, such data is empowered by the fact that it is machine-processable. This data can be combined with the other machine-processable information available on the Semantic Web to provide for more efficient services offered to the user and tailored to the particular user needs and ability.

Web Services

Current Web services face many limitations and conflicting proprietary standards [7]. For example, current Web service techniques do not offer clear separation between syntax and semantics; when applications are integrated, the grand effort is spent in negotiating different formats [48].

Web services could benefit from the Semantic Web technologies and standards released by the World Wide Web Consortium (W3C) to build better Semantic Web services than the ones that are currently attempted. Semantic Web services are expected to offer frameworks that enable agent software to make use of machine-processable data on the Semantic Web, and to monitor and verify the Web services. This will eventually provide better services to the users.

1.3 Problems in Preparing Web Pages for the Semantic Web

Many academic and scientific research groups are working on developing new approaches and putting the current available approaches into practice. The World Wide Web Consortium (W3C) focuses its activities on developing standards and technologies for the Semantic Web. Most of the new tools, concepts and frameworks are still used as research activities; in fact, we have not yet seen any productive application that fulfils the complete Semantic Web vision. This has already been addressed by [8] as:

> *"after four years of work by the W3C (World Wide Web Consortium) and other global collaborations there are as yet no complete practical or commercial applications of the Semantic Web"*

The current extensive research work on Semantic Web is directed towards creating a global semantic-based knowledge management implementation. Semantic Web depends on the availability of domain ontology as well as of "intelligent" Web pages annotated with meta-data that express the syntax and semantics of the page contents in a machine-processable way. Semantic Web services and Agents can use these annotated pages not just for display purposes, but also for machine-processing guided by the ontology as a model of content semantics.

In order to prepare Web pages for the Semantic Web, we can directly add Annotation codes to a Web page using any text or HTML editing tool; this process is neither practical nor acceptable with today's technology because manual Annotation is a time consuming and expensive process [29]. The other option is to use Wrappers or Annotators. Web Wrappers are used to convert implicitly stored information in an HTML document into information explicitly stored as a data structure [92]. Wrappers can also be written to convert structured data, e.g. data bases, to semantically annotated documents that are generated on demand and guided by domain ontology. Web Annotators, on the other hand, are used to add Annotations to Web pages. These Annotations or meta-data are the RDF instances required for the Semantic Web implementation; such Annotations can be statically associated and stored within the Web document itself [97] or, for some multimedia applications that annotate video or images over the broadband networks [94], can be externally stored on an Annotation server.

Automatic Annotation of Web pages is vital and might be at the same level of importance as the current indexing systems used by search engines [97]. Automatic Annotation can not only ease some of the current knowledge acquisition difficulties

[104], it can also encourage users to annotate their important Web resources using on-the–shelf software, and load them to the Web ready in machine understandable form.

Annotation is still a difficult and time consuming task. The 2002 DARPA ACE evaluation [65] got entity extraction scores of about 80% and relation extraction scores of around 60% which is not acceptable in real world applications. Even with the most advanced and ambitious tools like Armadillo [27], there is still much to do to reach the degree of accuracy and adaptability that fulfils the Semantic Web requirements. The Annotation process then plays a key role in the successful implementation of semantic-based applications.

1.4 Motivation and Context

Although, there are many proposed automatic Annotation tools that can ease the knowledge acquisition problems, yet none of these tools solves the problem of Annotation and simplifies it enough to be useful for every day production of Semantic Web resources. The motivations behind this research are:

▪ We believe that the idea of having each website developer annotating the site pages and/or creating the site's own RDF repository would give the Semantic Web the similar chances the current Web had to succeed and expand. Semantic Web will never become a reality unless the user is provided with tools simple enough to encourage users' contributions. Unlike any other technology, big companies or research communities are not expected to deliver an accurate ready made package because that is far too complicated to handle without the contribution of the users.

- We claim that the context of Web pages influences the meaning of the text on the page; this context is better understood by the community that owns the website than any of the available Annotation tools. The social factors in different countries play an important role in the interpretation and semantics of a given text.

- Web pages with languages like Arabic need special types of language processing resources that impose another motivation to start such research work. This will at least clarify the processing resources needed for processing Arabic language as compared to what is being used for English documents.

- Most of the current tools for automatic Annotation have a high learning curve; too much time and energy are required before the tool can be used by the average user. In addition, the degree of trust in the quality of Annotation produced is still not encouraging and not accurately measured.

- It is important to start contributing to creating awareness about the Semantic Web; the availability of many tools and frameworks, which are created by research communities for Semantic Web, provide an adequate starting point.

- The current extensive research aims at creating global semantic-based knowledge management implementation; many users do not have the will or motivation to wait for such work to be available while not sure how accurate the processing of Web semantic will be, and whether or not it will be cost effective.

- The objective is to create a Stand-Alone Semantic Web implementation that can be used as a model by many academic institutions. Such implementation provides Internet users, through the site's portal and Web services, with all possible services that can be accessed on site Intranet. Such a stand-alone Semantic Web implementation is a step towards the global Semantic Web.

The outcome of this work is a framework for building Semantic Web applications based on Context-Oriented Controlled Automatic Annotation tools. The average user will be using a simple procedure to annotate the Web pages before uploading them to the Internet. The implementation uses the latest standard technologies available for Semantic Web, which are currently providing the guidelines for the development of a sound application. These standards prevent new systems from falling into the category of the uncontrolled "let's build a city" way presented in Jack Schofield comment [93]:

"For Microsoft and IBM, it's like designing a giant metropolis, laying out the roads, agreeing on traffic regulations, putting in plumbing, and so on. For the hackers, it's more like "let's build a city: everybody bring a brick." This is not such a bad idea: it's basically how the PC industry and the Web succeeded. But how it will turn out in this case is anybody's guess"

1.5 Conclusions

We investigated the rapid growth of the Internet; we recognise that the Semantic Web is the solution to the current problems of the Internet. We stated our motivations behind this research and investigated the problems facing the rapid implementation of Semantic Web.

In this chapter we identified the need for a framework that enables developing stand-alone Semantic Web applications; here we recognize the importance of the Web page's context and its role in analysing and inferencing the correct meaning of the page contents especially in a multi-lingual environment. The successful implementations of such applications prepares for the global Semantic Web that will eventually include all

those developed stand-alone applications. Second, we identified that the unavailability of an appropriate Annotation tools is the problem that hinders the Semantic Web from being a reality in the near future; such a tool is needed for adding meta-data to express the semantics of the contents of Web documents.

1.6 Organization

The rest of this thesis proceeds as follows: the next chapter covers the Semantic Web basic concepts and tools. Chapter 3 presents the proposed Semantic Web Implementation Framework (SWIF). It also discusses the case study used as the real world application; that is the development of a semantic-based application for an academic institution. Chapter 4 describes the Semantic Web Implementation System (SWIS), the sample documents used and the algorithm developed for this implementation. It demonstrates some experimental scenarios, and also covers the ontology created for the institution. Conclusions and future work are highlighted in the last chapter.

2. Background and Related Work

2.1 Overview

In this chapter we briefly present the key stages of developing a Knowledge Based Management system as it is related to our work on the Semantic Web implementation. We distinguish between Knowledge Acquisition, Information Retrieval and Information Extraction. We use a case study to describe the information extraction process used by the General Architecture for Text Engineering (GATE) [46].

We present some of the evaluation measures and procedures used to evaluate information extraction systems.

We discuss Semantic Analysis of human language and the concept of Ontology. We briefly present several well known Annotation tools. In the last few sections of this chapter, we investigate the current technologies that enable Semantic Web implementation.

2.2 Knowledge Based Management System

The Semantic Web is a Web-based knowledge management application. The architecture of such a system addresses all of the following key stages of the knowledge management life cycle with the exception of the methodology used [33].

Knowledge Acquisition

Knowledge acquisition is a complex and expensive process [19]. Even with a traditional knowledge management system, knowledge acquisition has been regarded as a bottleneck [98]. The Semantic Web has created extra challenges; information is not only to be explicitly harvested from a page, but also mapped to the domain ontology that acts as a semantic model. Information about the relationship between pieces of information in a document can either be inferred, by the use of ontology, or explicitly harvested from the document's text. For example, the fact that, the job title of "John Smith" is a "lecturer", is explicitly specified in the text, but that "John Smith" is an Academic staff member and a Person are inferred by using the domain ontology. The extracted knowledge is normally loaded in some data structure in memory or most commonly stored as RDF triple in a database.

Knowledge can be acquired during the creation of a page by using tools like DOME editor [72] or OntoMat annotator [51]. Automatic Annotation tools like Armadillo [27] and systems like Artequakt [6] can also harvest knowledge from Web pages after they are created and loaded on the Internet, hence the term "late harvesting" is used with such systems. Ontology itself can be either built in advance or extracted from the websites by some tools like Adaptiva [20].

Knowledge modelling

Knowledge modelling technologies provide a bridge between the acquisition of knowledge and its use. Ontology is the essential model used in the Semantic Web to model knowledge of a particular domain. Ontologies provide explicit meaning of the domain knowledge that can be shared between user and software applications. The other important notion in knowledge modelling is service interaction protocols that specify all

possible sequences of messages that would allow the normal dialogue to be conducted successfully. This enables agents to maintain their dialogues in a correct way.

Knowledge Reuse

Reuse is a major challenge in knowledge management systems. Acquired knowledge is normally expensive; hence reuse of such knowledge is required and expected. Various pre-existing knowledge models, repositories or services can be used. For a particular service or resources to be reused, the user may have a large number of services for a particular query. Brokering services are required in such cases to allow for management of services.

Knowledge Retrieval

The prime goal of the future Web is intelligent retrieval. The current Web includes mainly unstructured information that is found with difficulty, in addition to valuable implicit knowledge that can not be inferred systematically. Retrieval with Semantic Web is made using the knowledge base constructed from annotated pages, the structured data that is converted to the knowledge base, and the dynamically collected data from user interaction with the Web. In addition, the capabilities of architectures, agents, Web services are expected to be integrated and directed towards intelligent retrieval of explicit and implicit knowledge on the Web.

Knowledge Publishing

The aim is to deliver knowledge to the right user at the right time, when and where it is requested or needed. Different users need to see knowledge presented to them differently. Personalized presentation is important and it is one of the major goals of Semantic Web.

Knowledge Maintenance

The successful computer application is the one that maintains its content updated as soon as transactions that change its content are executed. The knowledge base must be validated and verified because mistakes in such a system have a very dangerous impact on the future of using such an expensive repository. Hence it is essential to identify the part of knowledge that must be verified, certified, updated, or discarded. In addition, it is equally important to specify the time at which knowledge is maintained [4].

The Semantic Web demands several foundational concepts and technologies; information extraction tools, knowledge representation languages and tools, Semantic Web implementation technologies, Web services and Agents. In the following sections we shall briefly examine some of the main Semantic Web technologies and the concepts behind them.

2.3 Information Extraction

Information Extraction, Information Retrieval and Knowledge Acquisition are three different concepts used in the Internet domain. They can be distinguished as follows:

Information Retrieval (IR), as it is currently used, is concerned with getting a set of documents which are hopefully relevant to the user's query and presenting these documents to the user to analyse and look for the interested information within the presented documents one by one.

Information Extraction (IE) [9, 30] is more difficult to perform than IR in the way that it examines the text in the documents to identify a set of pre-defined relevant items and produce these items as unambiguous data in fixed format. The output of IE maybe

stored in a database or spreadsheets or may be used for indexing purposes in applications such as Internet search engine. IE has several advantages to different applications:

- Improves IR and Data Mining by examining the text within the documents to identify entities in the text and uses them for search or indexing.

- Extracts data from unstructured data and converts it to structured data that can be used by application software like e-business.

- Automatically adds semantics to Web pages which provides for Semantic-based processing.

Knowledge Acquisition (KA) is the science of extracting information from the environment and find mapping from the environment and its extracted information to concepts described in the appropriate modelling formalism like Ontologies [97, 19]. KA is an important aspect for Semantic Web, it does not only mean to acquire knowledge from Web documents and the Web environment to populate some template or data structure, it also involves acquiring knowledge to create or update ontologies using tools like Adaptiva [20]. KA uses Knowledge Engineering and other Information Extraction techniques to acquire knowledge from the environment. The acquired knowledge can be from Web documents, Web usage, Ontologies or other modelling formalism. Acquired knowledge might be used as knowledge base on a database or ontology that acts as a knowledge base.

2.3.1 Types of Information Extraction

The Message Understanding Conference (MUC-7) [78] is a competition that is investigated with more details in 2.3.4.2. MUC-7 defined IE to be of the following five types [30]: more details and examples can be found in Section 3.3.2.6 and 3.3.2.7

2.3.1.1 Named Entity Recognition (NE)

Extracting entities from text has been the simplest of the above five types. At MUC-7 most systems were functioning at about 95% which is an excellent performance compared to the slowness of manual extraction that may not reach 100% accuracy. NE recognition means identifying all the predefined entities like names of organisations, time in different formats, dates in different formats, money, etc.

2.3.1.2 Coreference Resolution (CO)

Coreference resolution is domain dependent IE task that identifies identity relations between entities; more details and examples can be found in Sections 3.3.2.6 and 3.3.2.7. The top performer in MUC-7 scored around 60% accuracy only which is very low score compared to named entity recognition score.

2.3.1.3 Template Element construction (TE)

The output of NE and CO is represented in some data structure or database. Each entity is given an identification number that is used as a reference to that entity. Based on the scattered information in the text, TE adds descriptive information to corresponding entities description whenever it finds new information that is related to a particular entity. TE is weakly domain dependent. The top performer in MUC-7 scored 80% accuracy for TE.

2.3.1.4 Template Relation construction (TR)

One of the basic requirements for any information extraction is to find relations between the entities of the document. This means that a person entity for example can be related to an organization entity because the person works for that organization. TR identifies such relations between TE. TR is weakly domain dependent. The top performer in MUC-7 scored 75% accuracy for TR.

2.3.1.5 Scenario Template production (ST)

ST is restricted by the predefined scenarios specified by the user. It fits TE and TR results into the specified event scenarios. ST is the most difficult task of all IE type. The best MUC system performed at only 60% accuracy. ST is a domain dependent task.

2.3.2 Approaches to Information Extraction

There are two basic approaches to the design of an IE system [9]: the Knowledge Engineering approach and the Automatic Training approach. Each of the two approaches has its own advantages and disadvantages. The appropriate approach should be used for a particular situation. We introduce both approaches in the sections below.

2.3.2.1 Knowledge Engineering Approach:

The Knowledge Engineering approach is characterised as being rule based. It requires skilled knowledge engineers to write the rules needed to mark or extract the information of interest. Knowledge Engineers usually consult other experts in the application domain and in some cases make use of their intuition [9]. The development of rules and the grammar used usually requires a lot of labour and time to write, test and

modify. The system developed is modified until acceptable performance is reached, so performance can be estimated for a particular domain during the development process of the system.

With the Knowledge Engineering Approach, changes to the system can easily be accommodated in the extraction procedure, by changing or adding new rules. This approach normally requires language resources like lexicons and lists; such resources are not difficult to get because they can be built from current digital systems; some have already been provided by the GATE team for several languages, others are commercially provided by companies like Basis Technology and its Rosette Linguistics Platform [12].

2.3.2.2 Automatic Training Approach

The Automatic Training Approach uses machine learning algorithms that automatically derive corpus statistics or rules. It requires the creation of a large amount of annotated training data; this represents the main problem with this approach. Training data may be difficult or sometimes impossible to prepare, for example the consistency of Annotation by different annotators might not be controlled. In most cases, the developed systems require user supervision to some degree, and the interaction with the user guides the learning process. In preparing training corpus, domain relevant data entity like proper names used in the domain, and coreference components should be included because the quality and quantity of training data affects the measure of the learning algorithm effectiveness.

The Automatic Training Approach is used when the extraction procedures and specifications are stable, training data is cheap, language resources like lists and

lexicons are not available, the highest possible performance is not vital for the task, and no skilled rule writers are available [9].

2.3.3 Evaluation of Information Extraction Systems

Evaluating Information Extraction systems such as language processing systems needs special criteria, and success is not easy to measure. The system development cycle for language processing follows the same steps needed for any other developed software system; such a cycle includes performance evaluation that might be needed to fix some errors and update the system until the desired performance is reached. Language processing systems are normally domain dependent. The developer can claim that the system is performing with certain degree of accuracy, but the actual performance of the information extraction system might not be acceptable on unseen documents, especially when applied in a different domain. Therefore, information extraction systems need to be evaluated by some other procedure to verify the results. This section investigates the procedures used.

2.3.3.1 Standard Evaluation Measures

Precision, Recall and F-measures are the standard measures normally used in evaluating information extraction and information retrieval systems [68, 99].

Precision is a measure of exactness; perfect precision means that the system used has extracted the exact instances that should have been extracted.

$$precision = \frac{TP}{TP + FP} \qquad \text{.. (1)}$$

TP refers to True Positives which denotes that the fragments that should have been extracted and were actually extracted. FP refers to False Positives to denote the

fragments that were extracted and should not have been extracted. So if the text contains 100 fragments that should be found, and the system finds those exact fragments, such system has 100% precision. If the system finds 110 for example, these extra 10 instances are the FP that will lower precision.

Recall is a measure of completeness; perfect recall means that the system has extracted all possible fragments; if instead of extracting 100 expected fragments, the system extracted 90 for example, such a system will have 90% recall.

$$recall = \frac{TP}{TP + FN}$$ …………………………………… (2)

FN refers to False Negative to denote fragments that should have been extracted but were not extracted.

F-measure is a term that combines precision and recall in one term that is known as the harmonic mean. In mathematics, the harmonic mean is one of several kinds of average, it is used to denote sub-contrary mean; in our case it denotes the trade-off between precision and recall.

$$F = \frac{2 * precision * recall}{precision + recall}$$ ………………………………… (3)

F refers to the F-measure where recall and precision are evenly weighted

2.3.3.2 Evaluation Exercises

The solution to the problem of evaluating language processing system became the subject of competitions. In this section we investigate some of the well known practical competitions organized to measure and evaluate the performance of the information extraction systems competing in such events.

2.3.3.2.1 MUC (Message Understanding Conferences)

The most well known competitions are the MUC (Message Understanding Conferences) started in the early 90's; the latest is MUC-7 in 1998. The following universities were among the major competitors in MUC-7.

- University of Durham

- University of Manitoba

- University of Pennsylvania

- University of Sheffield

The evaluation of Information Extraction systems in practical competitions involves quantitative measurement of the outcome of the system's performance. Modular answers were prepared by human analysts in advance. This practice has been used by other competitions organized for similar purposes.

The scores of the competitions in terms of Precision, Recall and F-measure are made public through the MUC website [78].

2.3.3.2.2 ACE (Automatic Content Extraction)

The Automatic Content Extraction program [2] succeeded MUC; the program continued the competitive quantitative evaluation cycles of MUC. ACE implemented slightly different information extraction tasks. Unlike MUC, ACE evaluation results are not public.

2.3.3.2.3 Pascal Challenge

The Pascal challenge [64] on information extraction and machine learning was organized by many universities and research institutions. The main organisers were from the university of Sheffield namely Fabio Ciravegna and Neil Ireson. The workshop was sponsored by Pascal network and DOT.KOM. The aim of this challenge was to

assess Machine Learning methodologies developed by different researchers to extract implicit relations from documents. In fact, it was only geared towards evaluation of entity extraction; several entities were identified in a pre-prepared annotated corpus of documents used as the challenge's dataset. The dataset was prepared by 10 different annotators and took more than 4 months to prepare. It consisted of 250 conference call-for-paper documents and 850 workshop call-for-papers, all documents were no more than 4KB in size. Several versions of the annotated corpus were released before the annotators agree on the final version. Part of the corpus was used as training corpus and the rest was used in the other different testing tasks. The result of the challenge can be found on the Pascal challenge website [86]. The results showed the variability in performance of the different 12 information extraction systems that contributed in the challenge and it highlighted the need for more research work to be done to improve the current available systems.

2.4 Human Language Semantic Analysis

Semantic Analysis is defined by [68] as follows:

> *"Semantic analysis is the process whereby meaning representations are created and assigned to linguistic inputs"*

Meaning representation is the key factor to the success of any semantic based application. The fact that the meaning of a sentence can be represented in several alternative ways made the inference process vulnerable to different interpretations. Many techniques and approaches have been used to achieve this task, the key point used by most techniques are directed towards the role of the verb in the sentence. The fact is

that the main verb of the sentence dictates the number and type of the grammatical arguments it takes [68]. It also dictates the location of the phrases that are expected to accompany each grammatical category. Understanding this role of the verb in the sentence helps in finding suitable implementations of the semantic analysis. We introduce the concept of Thematic Role because it can be used to play key roles in the process of information extraction needed for the Semantic Web.

2.4.1 Approaches to Semantic Analysis

There are two main approaches towards developing a human language semantic analyser: syntax-driven approach and Information Extraction approach. Syntax analysis is driven by syntax where the meaning of a sentence is based on the meaning of its parts; such analysis uses static knowledge from a lexicon and the grammatical structure of the sentence to create a context independent meaning. The information extraction approach uses partial parsing of the sentences; this is preferred for the sort of work such as the work presented here because it is more practical and efficient. Information extraction is based on cascaded automata to extract pertinent information of interest by ignoring some of the text that is irrelevant.

2.4.2 Thematic Role

Thematic roles are a set of categories that can be used to express the meaning of certain arguments of verbs. Each verb can have its argument associated with one or more roles; this role can be used to identify the meaning of the phrases in the sentence. Thematic role was first introduced by Gruber in 1965 [49], yet scientists have not agreed on the

list of thematic roles. Table 1 below shows the commonly-used list of Thematic Roles.

Role	Definition	Example
Agent	The doer of an action	John opened the door
Theme	The most directly affected by the action	John opened the door
Instrument	The instrument used to carry out the action	The key opened the safe
Recipient	The destination of a transfer event it involves change in ownership or possession	I sent Mary the flowers
Goal	The destination of a transfer event or object	He went to the library
Experiencer	The experiencer of an event	John has a headache

Table 1 : Commonly-Used list of Thematic Roles

Thematic Roles simple applications can be illustrated as in the following examples:

John opened the door. …………………………………………………….. (1)

Mary ate the apple. …………………………………………………. (2)

The subjects in both sentences are the actor "doer" of the action. Both John and Mary are animate and according to the thematic role category they are assigned the role of Agent. The door and the apple are inanimate objects, they are affected by the action of the agent, the role of Theme or "Patient" is assigned to them.

Each verb has its own thematic description. Fillmore [43] stated that each verb selects a certain number of deep cases (i.e. semantic role) which form its case frame; a

case frame describes the important aspects of semantic valence of verbs. Fillmore also used a thematic hierarchy to identify the subject of an active sentence as follows:

Agent → Instrument → Theme

This hierarchy is used to determine the thematic role of the subject of the sentence. This means that if the case frame of a verb contains an Agent, this one is realized as the subject; otherwise; if there is an Instrument, it is promoted to a subject else the subject is the Theme. If the thematic description contains only Instrument and a Theme, then the Instrument plays the role of the subject. The hierarchy is used in reverse to determine the subject of a passive sentence. The following Example illustrates how the thematic role of the subject is determined in active and passive sentences:

John opened **the door**
↓   ↓
Agent   Theme

John opened **the door** with **the key**
↓   ↓   ↓
Agent   Theme   Instrument

The key opened **the door**
↓   ↓
Instrument   Theme

The door was opened by **John** …………………………….. Passive: run in reverse
↓   ↓
Theme   Agent
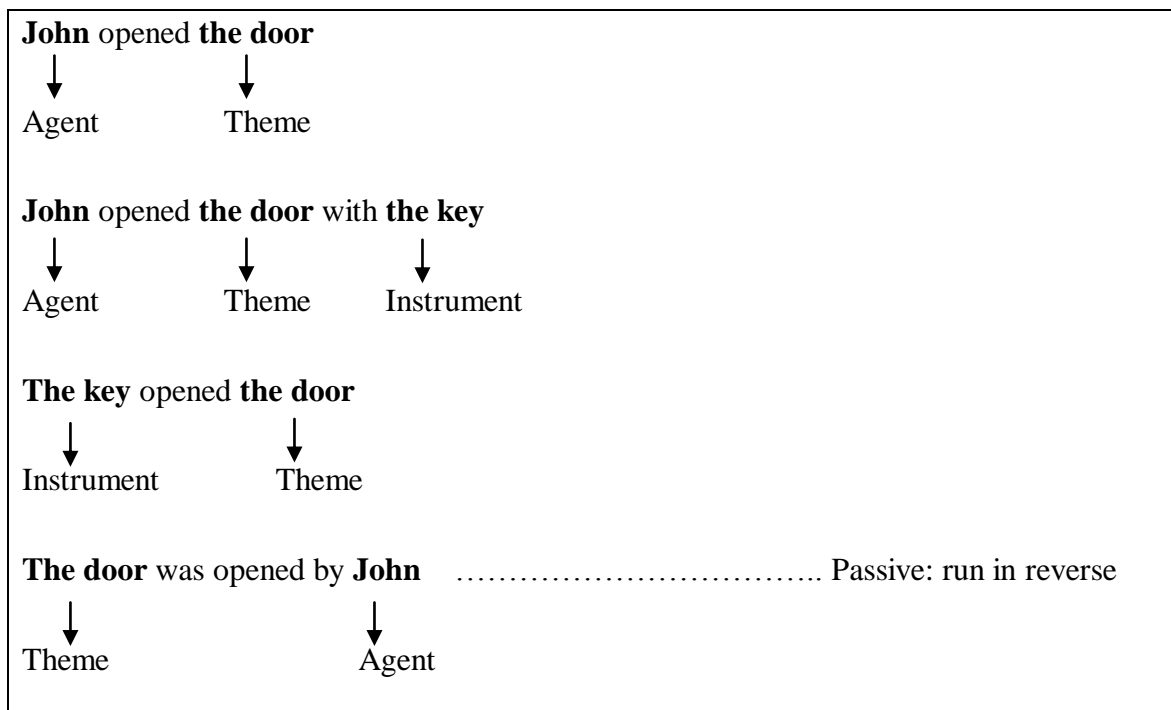
Thematic Roles can be mapped into the syntactic structure of a sentence. For example the verb open can behave according to the number of arguments associated with it:

The door opened.   One argument   - the door is subject - Theme

The key opened **the door**.   Two arguments   - the key is Subject- Instrument

                                                  the door – Object-Theme

John opened **the door** with **the key.**   Three arguments   - John is Subject –Agent

                                               The door – Object-Theme

                                               the key- Instrument

Many scholars like Levin [73] studied the semantic property and its effect on verb alternations. An alternation is the set of different mappings of the verb's conceptual role to grammatical function [68]. Levin summarised 80 alternations and associated them with different semantic classes of verbs, she also produced a list of verbs in each semantic class. Thematic role needs special handling for the verb unpredicted alternations. Normally a lexicon is needed to include for each verb all possible syntactic and thematic combinations.

The FrameNet project [10] has produced thematic roles in more specific categories than the ones listed in Table 1 above; this is because they used the concept of Frame. A frame is a script-like structure that instantiates a set of frame-specific semantic roles; this means that roles in FrameNet are specific to a frame and not to an individual verb. The FrameNet entries can be Core-Role or Non-Core Roles. FrameNet captures the generalization among frame elements by the use of inheritance because frames can inherit from each other. The FrameNet project aims at enabling inferences across different verbs and also between nouns and verbs. One main goal of the FrameNet project is to provide training data for semantic role labelling algorithms [68].

2.5 Knowledge Representation Languages

Several languages have been successfully used on the World Wide Web to provide information about the rendering of Web page content and its display effects [34]; examples of such languages are: the Hypertext Markup Language (HTML) and its eXtensible version XHTML, the scripting languages Cascading Style Sheets (CSS) and JavaScript. With the advancement in developing Web-based applications, especially e-business and Semantic-based applications, it became obvious that a full-fledged knowledge representation language is needed to express the structure's exact feature and the meaning of data. Many languages emerged; the latest is OWL announced as a standard in January 2004 [14]. In this section, we briefly present the knowledge representation languages that provide for the infrastructure of the future Web.

XML (Extensible Markup Language) is used only to add data format (structure) to the document, XML does not specify the data's use or semantics, even the vocabulary and the combinations of tags allowed are not defined in XML. A Document Type Definition (DTD) or an XML Schema might be used to specify this vocabulary. XML document is used to act as a means to hold the data in a structured way [34]. XML represents the basic language used by other successor languages like RDF.

RDF (Resource Description Framework) is a mechanism used to give meaning to the data. RDF consists of statements about resources, defined as subject-prdicate-object triples. The subject are resources on the Web, and the predicates are properties defined in RDF schema. A resource is any thing on the Web that has an identifier. RDF provides the RDF Schema (RDFS) as a mechanism to define domain-specific properties and classes of resources to which those properties can be applied; thus resources can be

defined as instances of subclasses or classes. RDF is written in XML, and it became a standard in February 2004 [106]. The following RDF document for example, describes some information of the resource 'http://computing.brad.ac.uk/people/dneagu" which is the home page of Dr. Daniel Neagu. The <rdf:Description> element contains the description of the resource identified by the rdf:about attribute. The elements: <uob:Tel>, <uob:email>, <uob:name>, and <uob:WorkFor> are properties(predicate).

```
<?xml version="1.0"?>
<rdf:RDF  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
          xmlns:uob="http://www.bradford.ac.uk/terms#">
  < rdf:Description rdf:about="http://computing.brad.ac.uk/people/dneagu">
   <uob:name>Daniel Neagu</uob:name>
   <uob:Tel> 01274 235704 </uob:Tel>
   <uob:email>D.Neagu@bradfrd.ac.uk</uob:email>
   <uob:WorkFor  rdf:resource="http://www.bradford.ac.uk"/>
  </rdf:Description>
</rdf>
```

RDF models statements as nodes and arcs in a graph. A statement is represented by a node for the subject, a node for the object, an arc for the predicate, directed from the subject node to the object node. Groups of statements can be represented by corresponding groups of nodes and arcs. RDF uses URI references instead of words to name things in statements. A common namespace is normally chosen for the vocabulary needed, for example the namespace **uob** in line 3 of the above example is used to prefix the vocabulary required by a domain.

OIL (Ontology Interchange Language) integrates intuitive modelling primitives, Web languages, and formal semantics into one language. One of its dialects, called DAML+OIL [77], The main additions to RDFS is formal semantics, based on a description logic, and more advanced modelling primitives, such as Boolean expressions and some axioms.

OWL (Web Ontology Language) is built on top of RDF and RDFS. It has a larger vocabulary than RDF, formal semantics, and stronger syntax. OWL can specify exact description of resources on the Web, and also gives high interpretation power to software applications. The major extension in OWL over RDF is its ability to specify restrictions on properties; for example "man" and "woman" can be declared as two *disjoint* classes, the owl:cardinality can be used to specify that a "woman" can have *one* "husband", the rdfs:range axiom can be used to specify that "husband" must be an instance of class "man" [14]. OWL has three increasingly-expressive sublanguages: OWL lite, OWL DL, and OWL Full.

2.6 Ontology

Ontologies specify the vocabulary of all possible terms used in the domain of discourse and the relationships that may exist between these terms. Ontology is simply defined by Gruber in [50] as:

"Ontology is an explicit specification of a conceptualization".

Several other definitions of Ontology have been presented. The Gruber's definition referenced by [51] as:

"Ontology is a formal, explicit specification of a shared conceptualization of a domain of interest".

This definition describes ontology as **formal** to mean that knowledge is represented in a machine-understandable language. **Explicit** to denote that the possible terms and the relationships between them, together with the constraints defined on them, are clearly specified by the vocabulary used in the Ontology. **Shared** means that the Ontology is

used by a group of people who reached consensus about common concepts, it also means that people and software agents have common understanding of the information. Conceptualization refers to an abstract model that identifies the concepts of some domain [39].

Ontology plays the key role in Semantic Web applications because it provides for the separation of domain knowledge from the operational knowledge. It also helps in analyzing domain knowledge; formal analysis of terms is extremely valuable for reusing or extending existing ontologies [76].

Large ontologies, such as **WordNet** that contains over 100,000 terms [39] was built much earlier than the introduction of Semantic Web; it uses natural language to describe its terms. Currently, there are many standard ontologies that can be used for an appropriate domain like **SNOMED** for the medicine domain [89]. The United Nations Development Program and Dun & Bradstreet developed the **UNSPSC** ontology for products and services [101]. The ontology **ecs** [100] has been developed and used by the department of Electronics and Computer Science at the University of Southampton. Some ontologies were built for conference purposes like **iswc.owl** [66]. Some ontologies were developed by information and knowledge organization like the **univ.owl** developed by Mondeca [62]. The most famous ontology used for ontology construction tutorial and demonstration purposes is the **Wine** ontology [63] that is developed by Stanford University. In short, currently there are many other ontologies that cover a wide range of domains from a Business to Business (B2B) to Weapon of Mass Destruction.

Current ontology research work is not only directed towards developing languages and tools that help in creating new ontologies but also to prepare for managing

ontologies that change over time, and combining separately developed ontologies to answer some queries that requires merging and relating several ontologies [39].

2.6.1 Ontology Preparation Tools

Ontologies can be created and maintained by using one of the following tools:

- **Simple editor** where the ontology is directly created using a simple editor like Notepad and saved as .xml or .owl file.

- **Ontology editor** where editor like Protégé [60] is used. Ontology editor supports the definition of concepts hierarchies, the definition of attributes for concepts, and the definition of axioms and constraints. They normally provide graphical interfaces. Ontology editor enables browsing, inspecting, and modifying ontologies and thus supports the ontology development and maintenance task.

- **Ontology Learners** where tools like Text-To-Onto [74] and Adaptiva provide an integrated environment for learning ontology from text in a semi-automatic manner.

- **Ontology Reasoners** where tools like the Fast Classification of Terminologies (FaCT) [61] can be used to derive concept hierarchies automatically.

- **Ontology Library Systems** are used to store, retrieve and manipulate existing ontologies. Systems like WebOnto [41], OntoLingua, DAML library system, Ontology Server and SHOE [40] feature re-use existing ontologies; new ontology can be constructed by assembling or modifying an existing ones.

- **Adapting Tools**. Chimaera [76] is a good example of such adapting environment that is used for merging and diagnosing ontologies. The PROMPT [39] tool is provided as a plug-in for Protégé-2000, and it is used for merging ontologies.

2.7 Annotation

With current technology, the "intelligent" kind of document as imagined by members of Delphi Group [35] has become feasible. Web pages in Semantic Web act as "intelligent" by adding Semantic Annotations. This process involves formal identification of concepts and the relationships between concepts in the document. Web pages are enriched with machine-processable information that made it possible to formalize the semantics of Web resources. The intelligent document is defined by [104] as:

"A document which "knows about" its own content in order that automated processes can "know what to do" with it"

To prepare Web pages for the Semantic Web, we can directly add Annotation code to a Web page using any text or HTML editing tool; this process is not practical and not accepted with today's technology, manual Annotation is difficult, time consuming and expensive [29]. Hence, many Wrappers and Annotators have been developed. Web Wrappers are used to convert implicitly stored information in an HTML document into information explicitly stored as a data structure [92]; Wrappers can also be written to convert structured data (e.g. data bases) to semantically annotated documents that is generated on demands and guided by domain ontology. Web Annotators on the other hand, are used to add Annotations to Web pages. These Annotations or markups are RDF or OWL instances required for Semantic Web manipulation; such Annotations can be stored within the Web document itself or, for some applications, can be externally stored on some Annotation server or local machine [109].

The current available Annotation tools can be classified according to the degree of automation they adopt as Manual, Semi-automatic, and Automatic Annotation tools.

Annotation is still a difficult and time consuming task. Even with the most advanced and ambitious tools like Armadillo [27], there is still much to do to reach the degree of accuracy and adaptability that fulfil the Semantic Web requirements. The following sections investigate current Annotation tools, present the specifications of the newly developed Annotation tools, and briefly present the standard measures used to evaluate Annotation systems.

2.7.1 Annotation Tools

Web pages in Semantic Web act as "Intelligent" by adding Semantic Annotations. This process involves formal identification of concepts and the relationships between concepts in the document. Web pages are enriched with machine-processable information that makes it possible to formalize the semantics of Web resources. Many tools have been developed to prepare Web pages for the Semantic Web. This section investigates the Annotation tools according to the degree of automation they adopt. At the end of this section, several Annotation and authoring environments are presented because they are receiving a great deal of attention nowadays.

2.7.1.1 Manual Annotation Tools

Most of the current manual Annotation tools provide a user friendly GUI with Ontology browser and document browser, they also offer the drag-and-drop feature to help in connecting certain data (instance) on the document to its type (class), also help in creating relationships instances.

Mangrove is a system developed at Washington University [103]. The Mangrove system provides services to help in improving the Who's Who entry by annotating the

personal home page and to adding an event to the Department Calendar. The most recent addition to the Mangrove services is the semantic email service that aims to make the emails that we interact with both human- and machine-understandable [75].

Mangrove users annotate their documents by using either the Mangrove Graphical Tagger which is the graphical tool provided or through a traditional editor guided by the simple hierarchy of allowable tags Figure 1 shows the Mangrove user friendly graphical tagger. The tool is Context-Sensitive; the pop up menu, with all valid Annotation tags, appeared when a right-click on a text was done. Figure 2 represent the html version of the annotated document and shows Annotations stored within the annotated page.
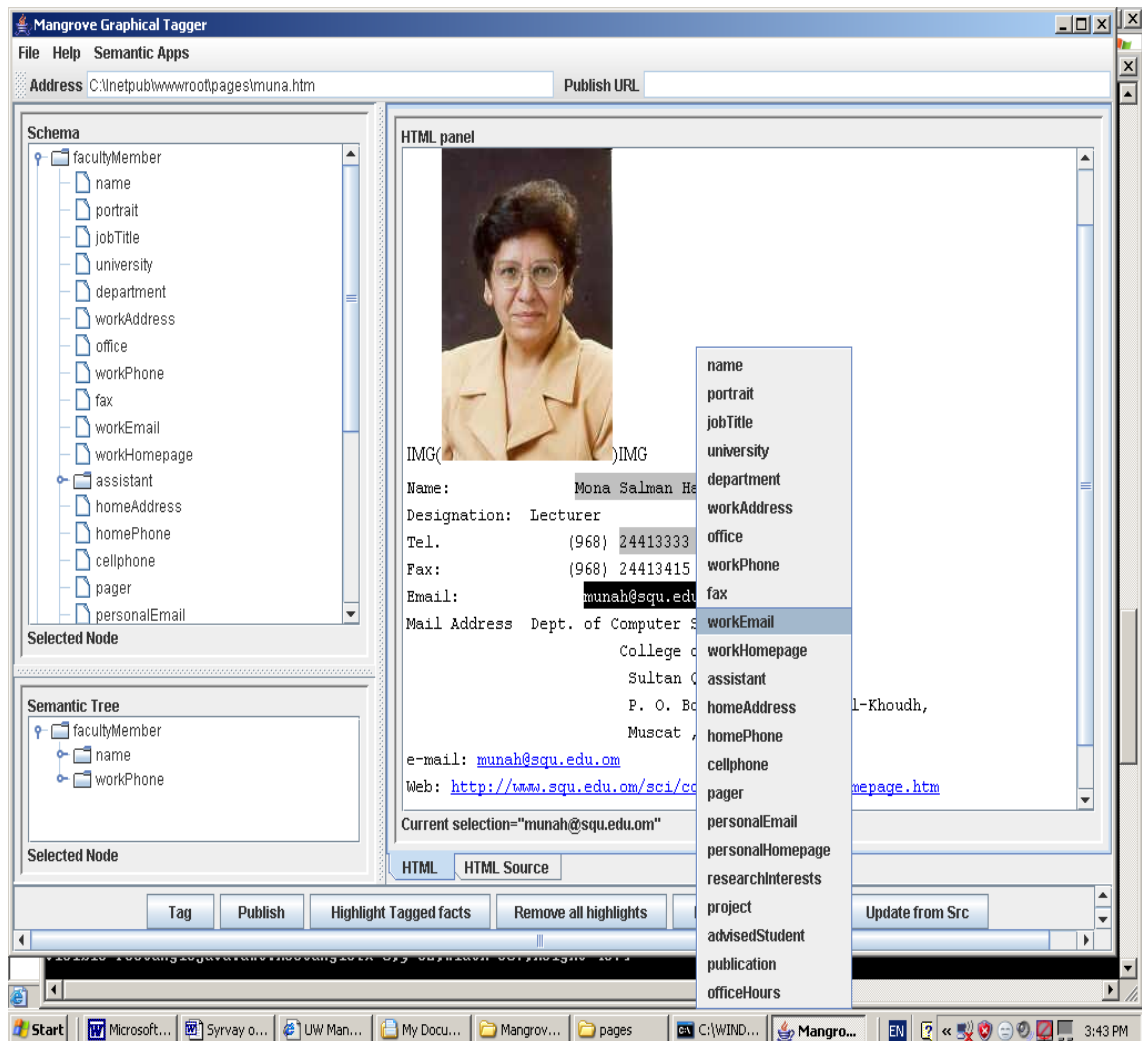
Figure 1: Mangrove Graphical Tagger
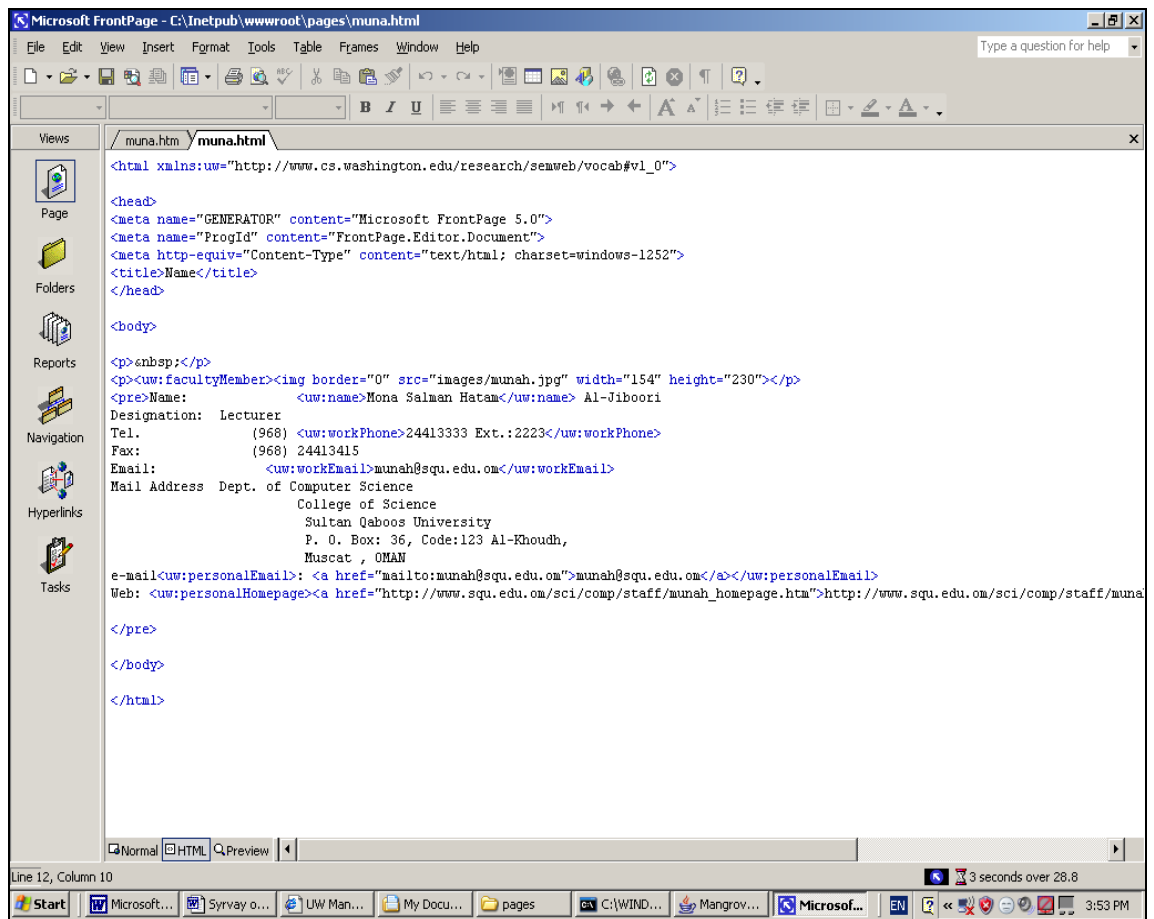
Figure 2: HTML Version of the Document Annotated by Mangrove Graphical Tagger

Vannotea has been developed at the University of Queensland [94]. It is different from other Annotation tools because its use is not intended for a stand-alone environment. Vannotea architecture uses the Annotation tool Annotea [105] is developed by the W3C. Vannotea is a collaborative video indexing, Annotation and discussion system for broadband networks. It annotates multimedia files like images, video, and audio resources. It enables geographically distributed groups connected across broadband networks to perform real time collaborative sharing indexing, discussion and Annotation of high quality digital films and images. The Annotation produced by the collaborating users is stored as external Annotation on the Annotation server. Annotation is done for defined regions of the images or individual shots of a video film. This feature of inputs from distributed users together with the flexible architecture

allows it to be used for different domains ranging from museum items to surgical and clinical multimedia [102].

SMORE provides the user with a flexible environment to markup a document with minimal knowledge of RDF terms and syntax. It provides for simple creation of OWL entities using drag-and-drop and menu-based interaction. It also provides the ability to work with OWL entities and data values in a triple format. SMORE also has a smart individual editor that presents a list of eligible targets for all object property links from an individual. It also provides an editor for each built-in data type [80]. SMORE uses the SWOOP [81] ontology browser and its Built-In HTML Editor. It allows the user to perform basic editing of HTML documents. The user can create his ontology from scratch and can import multiple existing ontologies into the created ontology. Figure 3 shows a SMORE GUI screen shot.
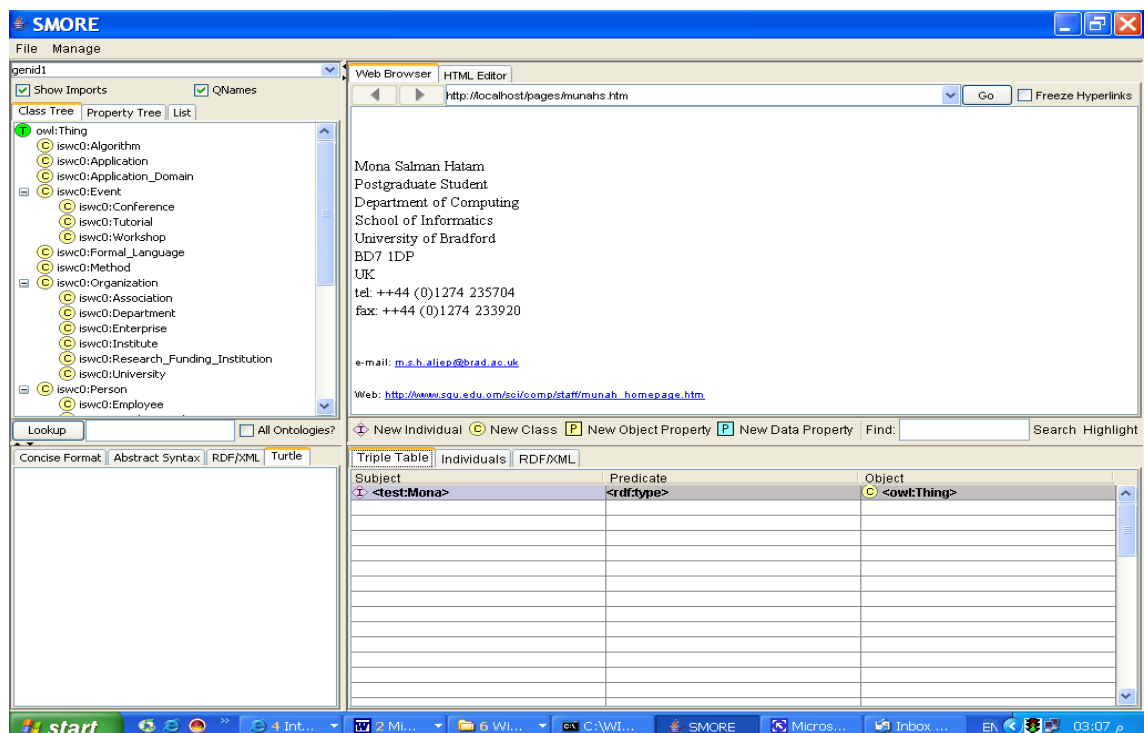
Figure 3: SMORE GUI

2.7.1.2 Semi-automatic Annotation Tools

In addition to those capabilities provided by manual Annotations, semi-automatic tools rely on some degree of information extraction automation that mainly needs large training corpus and manual Annotation for the training phase.

OntoMat [51] started as a completely manual annotizer and then evolved to include support for semi-automatic Annotation facilities based on using the information extraction system Amilcare [28]. OntoMat annotizer has been commercially supplied as **OntoAnnotate**. OntoMat has a rich GUI with special pane for ontology viewer, attributes and object properties. The HTML browser is used for the display of the document as HTML page, Annotation or the deep Annotation associated with pages generated from databases. OntoMat allows the annotator to highlight relevant parts of the Web page and create new instances via drag-and-drop interactions [51]. The latest research extension on OntoMat aims at the creation of M-OntoMat-Annotizer that supports manual Annotation of images and video data [104]. Figures 4 and 5 show the screen shots of OntoMat GUI window and the Annotation viewer.
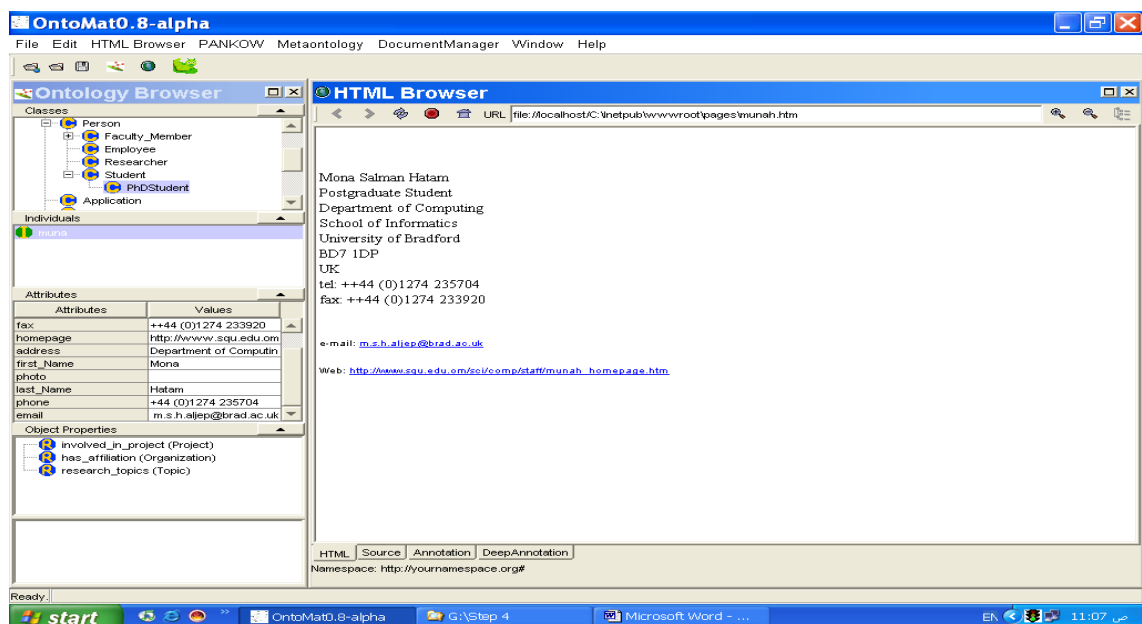
Figure 4: OntoMat GUI window
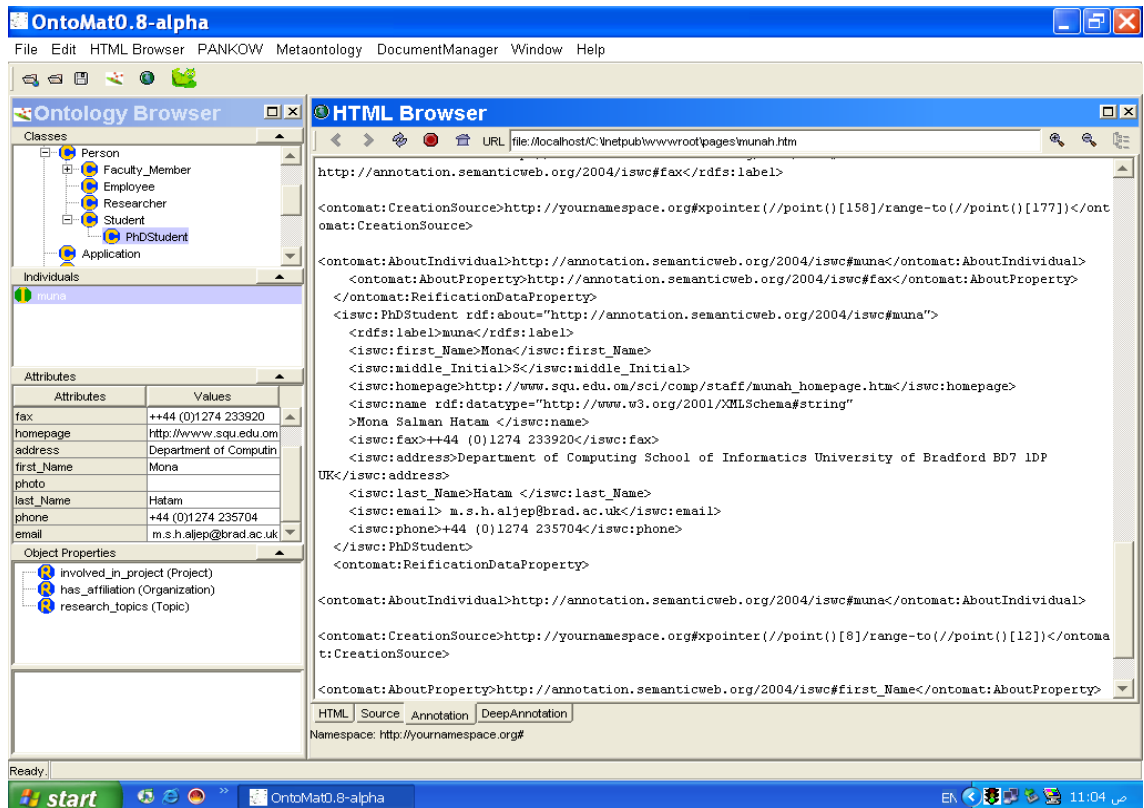
Figure 5: OntoMat Annotation Viewer

Melita [29] is an Annotation interface that uses the adaptive information extraction system Amilcare to extract information from texts. The user annotates the documents in a similar way to a OntoMat user, at the same time, Amilcare runs in the background learning how to reproduce the inserted Annotation. Rules are generated from this learning process and applied to the new documents to suggest some preliminary and partial Annotations. The user then monitors and supervises the automatic Annotations produced. The users have to correct mistakes and add missing Annotations, actions are monitored by Melita and new Annotations are inputted back to the learner for retraining. The user can specify the level of accuracy needed and the system takes over to become fully automatic.

2.7.1.3 Automatic Annotation Tools

Automatic Annotation of Web pages is vital to the success of Semantic Web, it is as important for Semantic Web as today's indexing systems are for current search engines [97]. Current research focuses on automatic Annotation; several new tools have emerged others have been extended.

The degree of automation in automatic Annotation tools is high because they adopt complex algorithms for Information Extraction; the combination of different techniques used by the algorithms increases the degree of information extraction and provides some degree of verification to the information extracted.

Automatic Annotation normally relies on machine learning techniques; the state of the art algorithm, that we can identify at this stage, is the Learning Pattern by Language Processing $(LP)^2$ [26]. $(LP)^2$ has been used to induce symbolic rules for information extraction by learning from a training corpus. Amilcare [28] is the system that implemented $(LP)^2$. It has been adapted by many automatic and semi-automatic Annotation tools like Melita, Armadillo, MnM, and the extended version of OntoMat.

Automatic Annotation aims at maximizing Annotation accuracy and minimizing user intervention in the Annotation process. The more unsupervised machine learning technique the tool use, the better automation it is. The accuracy of automatic Annotation tools is still limited [104]. The accuracy measured for some of the best automatic Annotation tools in term of precision and recall is still far below the accuracy required by the Semantic Web; a system that makes nearly one mistake out of two relations extracted is hardly acceptable in real life applications [65]. The area of automatic Annotation is still an open research area and finding the practical and systematic way to reach the "intelligent document" needed for the Semantic Web is still the priority in the road to Semantic Web.

Armadillo is a domain independent system that can be used to produce domain specific Annotation with minimum user intervention [27]. It starts with domain initial lexicon and domain ontology; the lexicon is automatically expanded to include new learned instances from the recognized regularities in the repository that Armadillo creates.

Armadillo uses an adaptive information extraction technique together with Information Integration and Machine Learning techniques. It extracts information from structured sources and use this information as a seed for the Machine Learning process. It uses Web services for performing several sub-tasks. Several services are used to deduce correct instances of some concept; these instances are added to the list of seed instances. For this purpose, Armadillo selects Web pages that normally contain organized structures such as lists and tables. A classifier trained with linguistic and formatting criteria is used to relate other instances in the selected documents to the seed instances and hence learn more instances for some concept. The extended seed instances are used for the next Machine Learning step. Amilcare [28] is used in a cycle of Annotation–learning-Annotation process to discover more instances for the required search with respect to the seed samples.

Armadillo triple store contains the extracted facts from the processed Web pages. Knowledge in the triple store is the RDF triples which define relations in the form subject-verb-object where the verb is the relation between the subject and the object. The triple store is used by different services to add new triple or confirm existing ones. In addition to the RDF triples, the triple store contains confirmation information and links to the source Web pages [27].

The reliability of information in the triple store is measured by the type and number of confirmation assigned to it. The strategy for confirmation used is application dependent. Figure 6 below shows the Armadillo architecture as illustrated by [27].
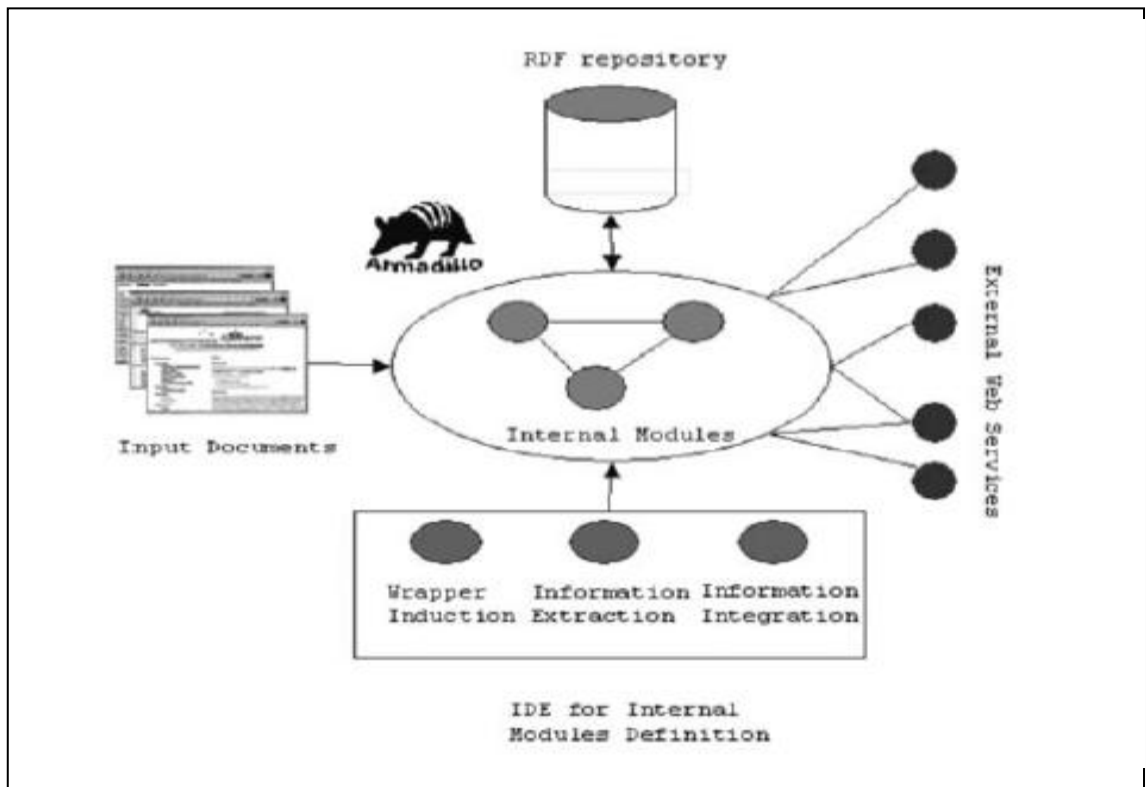
Figure 6: The Armadillo Architecture

Since the beginning of this decade, research on Semantic Web and automatic Annotation has been intensified, many big research projects with huge budgets are working in this direction. The research is directed towards the global knowledge base system that fulfils the Semantic Web vision. Systems like Cafetiere [18], KnowItAll [42], SemTag [38], KIM [87], and AeroSWARM [70, 95] have all been developed; yet still the accuracy of Annotation is limited and more has to be done in this direction.

Currently, the Semantic Web Annotator (SWAN) [36] is a very big and ambitious project in this direction. SWAN is about large scale Annotation of human language for the Semantic Web using human language technology. The project is based at the Digital Enterprise Research Institute (DERI) in Ireland and cooperates with the GATE [46] research team in the University of Sheffield and the OntoText laboratory of Sirma AI Ltd in Bulgaria [85].

2.7.1.4 Annotation and Authoring Environments

The idea of integrating Annotation process within authoring environment has received great deal of attention during the last few years. The aim is to get the document annotated while it is written. Systems like WiCKOffice [79], and ActiveDoc [5] are based on this idea. There is even a commercial Annotation system OntoOffice [11] supplied by Ontoprise.

Amaya [108] is a complete Web browsing and authoring environment it includes a collaborative Annotation application tool; Amaya annotates a Web document without editing it. The Annotations produced are free text statements about the document added by one or more users and stored on local machine or one or more Annotation servers. This type of Annotation is not formal Annotation that produces "Intelligent" document that is semantically annotated with machine processable meta-data. Annotation though provides a portion of such meta-data marked in XML or HTML according to the normally simple RDF Annotation schema used. Amaya presents Annotations with pencil Annotation icons attaching XLink attributes to these icons. If the user single-clicks on an Annotation icon, the text that was annotated is highlighted while double-click displays the Annotation text and the associated meta-data in a separate window as shown in Figure 7 below.
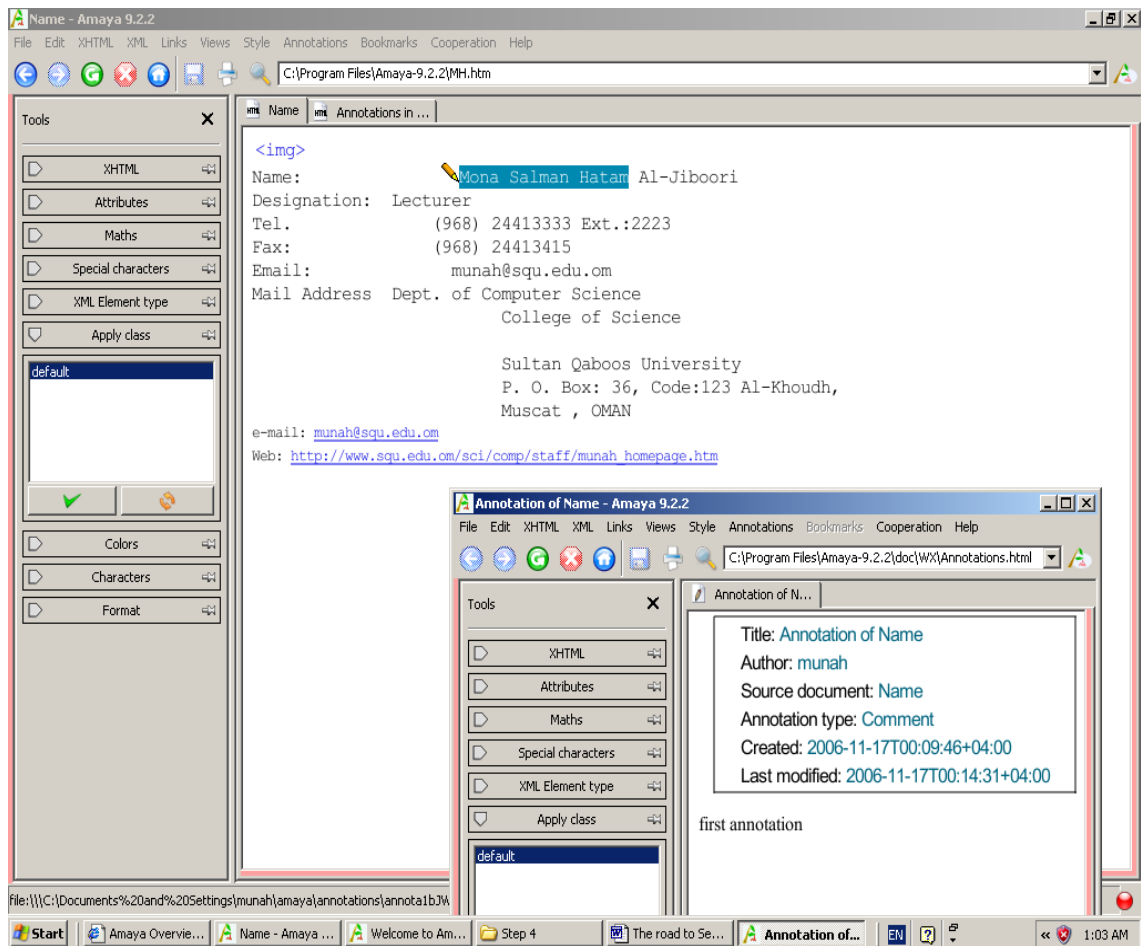
Figure 7: Amaya Annotation Icon and Meta-data

WiCKOffice [79] is a tool that uses the simplicity of the Microsoft Office Smart Tags syntax and extends it to support new features which made it suitable for deployment in a knowledge based environment. WiCKOffice provides automatic assistance for form filling using extracted data from knowledge bases.

SemanticWord [96] is an environment based on Microsoft Office Word (MS Word) that is used for document authoring and semantic Annotation. It allows simultaneous generation of content and semantic Annotation. The MS Word is extended to include new contents in GUI that facilitates Annotation using Annotation schema. Semantic Word integrated the automatic information extraction system AeroDAML [70] to capture proper nouns and frequently occurring relationships. A customizable library of

templates containing partially annotated text is used to help speed up the Annotation process. It also provides an environment for refining and augmenting the results of the information extraction process.

AktiveDoc [5] is a tool that supports document writing and reading. It uses three different kinds of document Annotation: Web technologies to support the production of ontology based Annotations; comments can be added to the document; external services are used to enable easy knowledge reuse, existing Annotations can be connected to knowledge bases and ontologies to suggest content for inclusion in the document. AktiveDoc saves Annotations in a separate database and classifies in three levels of confidentiality as public, private or confidential.

2.7.2 Measures for Evaluation of Annotation Systems

As Annotation is an Information Extraction process, Precision, Recall and F-measures mentioned in Section 2.3.3 are used for the evaluation of Annotation systems. For example if perfect manual Annotation adds 10 Annotation instances to a document and the Annotation system performs the same Annotation, then the Annotation precision is perfect or 100% accurate, annotating extra instances that should not have been annotated denotes low precision. Perfect recall means that the system has annotated all instances that should have been annotated. A system may have perfect precision even if it has imperfect recall; if the system has annotated 9 of the required 10 instances then the system precision is still 100% accurate in terms of precision, but only 90% accurate in terms of recall. It is also possible that, a system have perfect recall and imperfect precision; such system annotated extra instances that should not have been annotated.

2.7.3 The Specification of Annotation Tools

The process of creating new Annotation tool needs precise specification before the start of developing such a tool. The size and complexity of such tool is highly affected by many factors especially the following:

Specification of degree of automation: this means that the decision should be taken in advance as whether the tool should be for manual, automatic or semiautomatic Annotation.

Identification of the information to annotate: this means that data items of interest should be clearly stated; for example, the annotator only annotates entities and relationships, others resolve pronominal coreferences. Examples clarify what is expected from the annotator; to annotate an item as Date entity, all possible way in which date can be expressed should be well described. Date can be in different formats: "1st Jan. 2008", "1st January 2008", "January, 1 2008", "1-1-2008", "01-01-08" etc. Annotator may be required to resolve pronominal coreferences, so for example on the identification of a pronoun, what action should be taken and whether or not this pronoun should be linked to the entity object it denotes. Processing of abbreviations and synonyms might be required to be treated as denoting the same entity instance of the same document; examples of such cases are: "UK" and "United Kingdom", "Britain", and "Great Britain". Processing of orthographic matching like "John Smith", Mr. Smith", "Dr. Smith", "J.Smith", "Smith,J." In addition, the domain of Annotation can be specified or declared to be variable domain.

Representation of Annotations: the syntax and method of representation should be clearly stated; some Annotation tools used template mark-up approach where information is wrapped to a structure. Most Annotation tools use the XML like tag to denote the start of an entity and the end of the entity with the entity itself in between these two tags. The choice of adopting external Annotation or embeding in the document or both has to be specified in advance.

Availability of Language Resources: this denotes the availability of lexicons, gazetteers, lists, etc., in some case these resources are not easy to provide.

2.8 Human Language Technologies and Concepts

In the previous sections, we discussed the importance of automatic Annotation for Semantic Web. Automatic Annotation can not be feasible without Human Language Technology (HLT) that supports entity and relation extractions. The General Architecture for Text Engineering (GATE) is the world leading HLT infrastructure that supports Semantic Web. In the section to follow, we briefly describe the main features of GATE that we shall use in this work. In Section 2.8.2 we examine GATE in action.

2.8.1 Overview of GATE

The GATE architecture is developed at the University of Sheffield. It is suitable for usage in many different kinds of text processing applications and for various purposes. It has proven to be a powerful tool for entity Annotation of text, RTF, SGML email, XML, and html documents. GATE includes an Information Extraction (IE) language and processing set of components called ANNIE (A Nearly-New Information Extraction

system). The ANNIE entity extraction components are based on a Knowledge Engineering approach for IE; samples of such components are the tokeniser, sentence splitter, verb group chunker, and noun phrase chunker. GATE is open source and its components are evolving rapidly.

GATE components are Java Beans. The collection of resources in GATE is called CREOLE (Collection of Reusable Objects for Language Engineering). GATE components can be classified as follows:

- **Language Resources** such as ontologies , gazetteers.

- **Processing Resources** such as parsers, tokenisers, part of speech tagger.

- **Visual Resources** that constitute the GATE's GUI they include editing and visualization components.

Although GATE comes with a very useful **GUI**, some application developers want to use GATE through their own programs. GATE framework is supplied as two JAR files, these files can be linked to the Java environment used by the application developer, which makes it possible to use all GATE facilities from within the new application. The simple handling of the framework makes GATE the candidate of choice for language engineering.

The GATE Unicode Kit (**GUK**) provides for processing documents in different languages; documents written in languages like Arabic, Chinese or any other language can be processed in GATE because it provides the facilities needed to work with Unicode, an example of such facilities is GATE's abilities to read different character encodings.

The Java Annotation Pattern Engine (**JAPE**) available with GATE enables writing special grammar or rules that search for certain patterns to associate actions to it. The

actions can be adding, altering or deleting previous Annotations or features. Such grammar is the key facility used for entity or relation identification.

GATE provides the **Annotation diff** tool that can be used for comparing Annotations produced by GATE for some document with Annotation produced for the same document using another Annotation method like manual Annotation. This is very helpful in measuring the performance of automatic Annotation produced by GATE.

There are many other useful features in GATE like the use of machine learning algorithms, and many more that we can not cover within the scope of this work.

2.8.2 Case Study: Information Extraction using GATE

In the section below, we shall investigate the information extraction process in detail using GATE system of the University of Sheffield. Inspired by the discussion of the GATE developer in [30], we shall demonstrate GATE using a sample text document of a news article. Figure 8 is a screen shot of the GATE User Interface that shows the sample document. The objective of this exercise is to investigate GATE as human language processing tool used in this research work.

2.8.2.1 Named Entity Recognition in GATE

As an example, the GATE system is used to show samples of the processing of different types of information extraction; Figure 8 shows a sample document to be processed for extracting entities.

Figure 9 shows the entities of interest highlighted in different colours, Figure 10 shows the details of the code generated by the Named Entity Recognition process, and it shows the names of the rules, as named by the GATE's Java Annotation Pattern Engine

(**JAPE**) grammar, applied to deduce any information about the different tokens of the processed text.
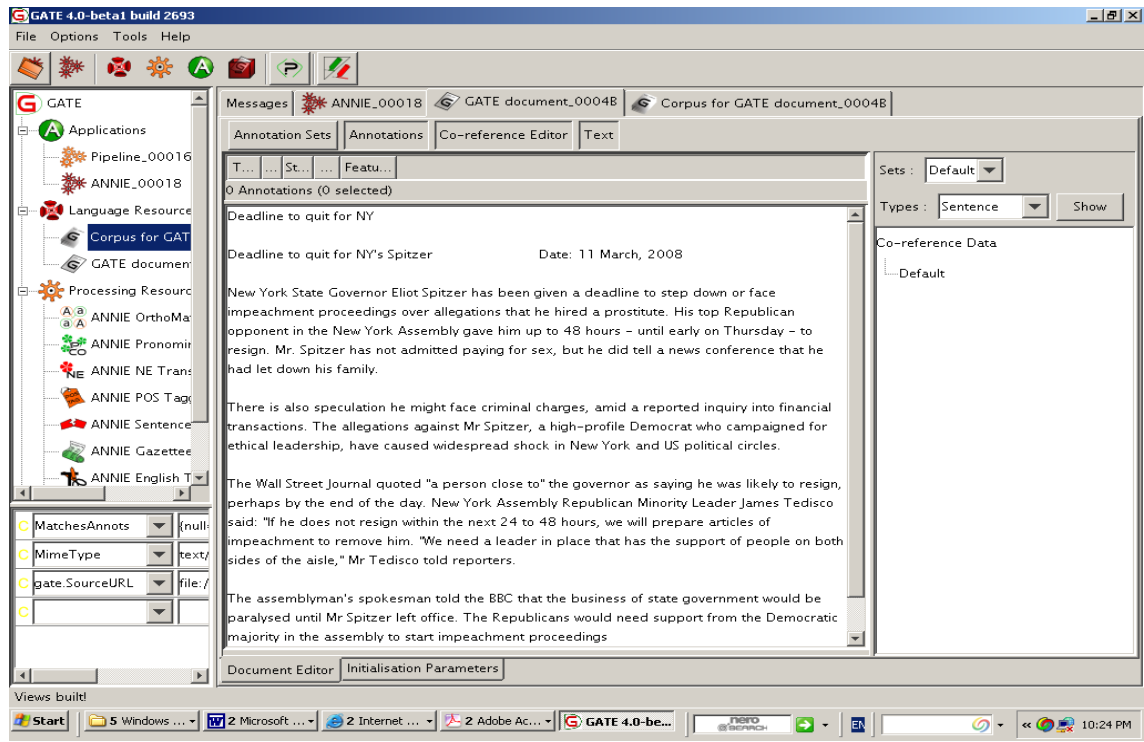
Figure 8: Sample Document and GATE GUI
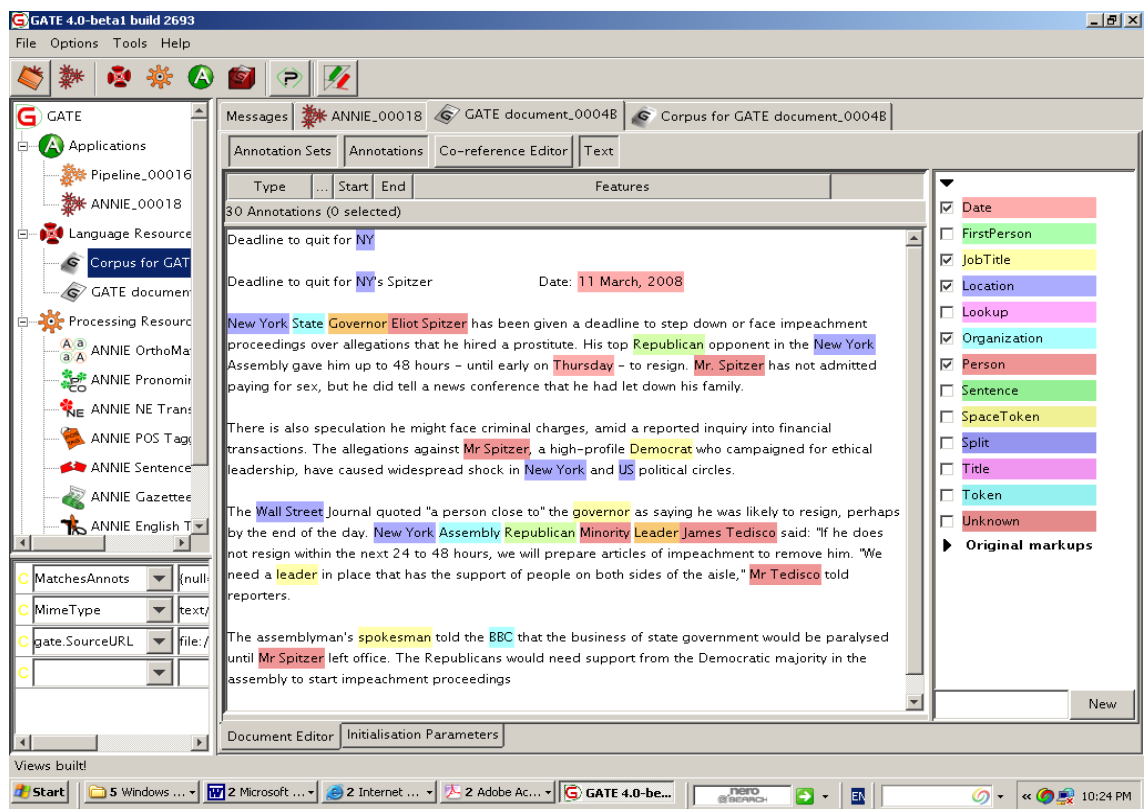
Figure 9: Named Entity Recognition
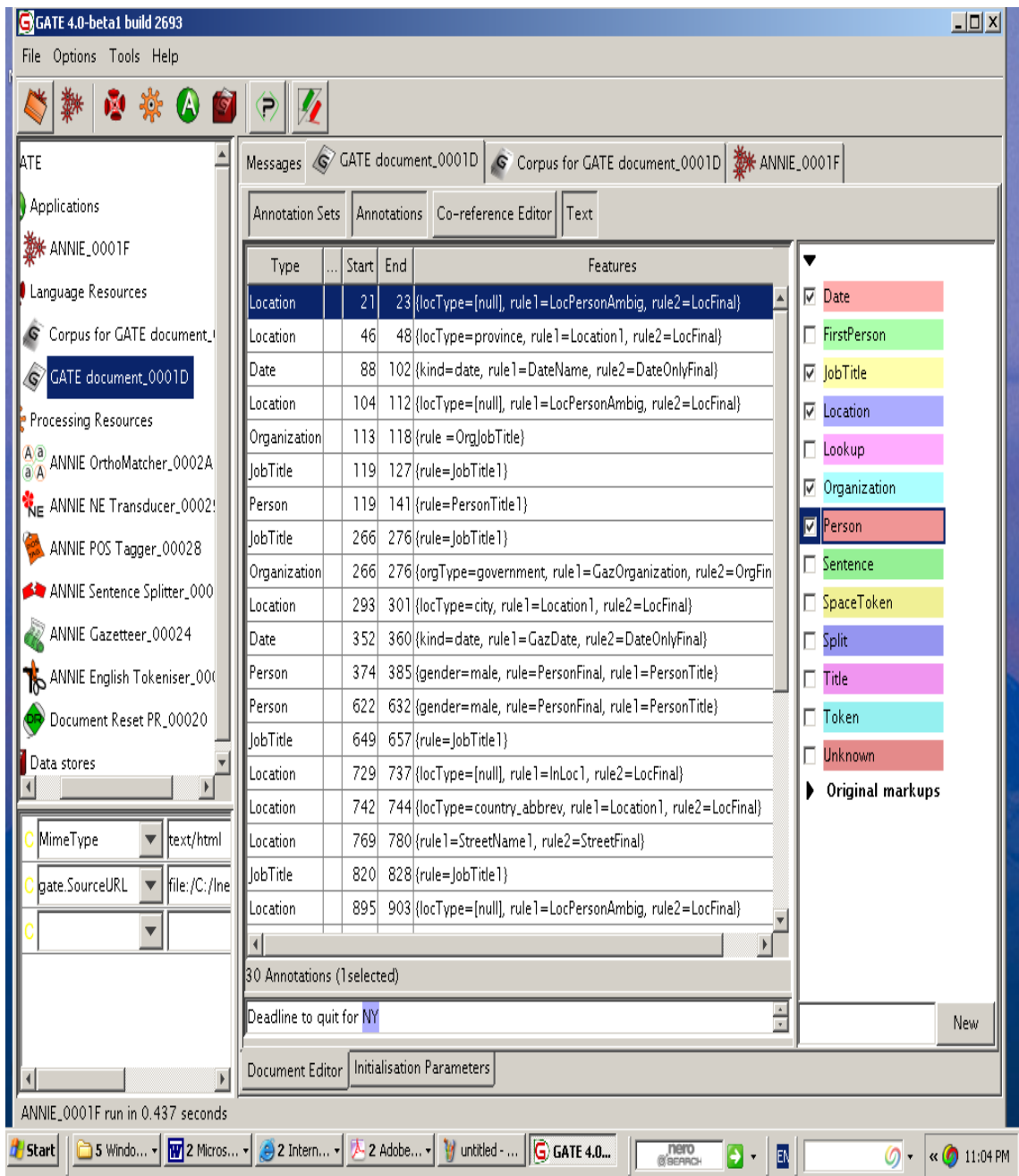
Figure 10: Annotation Sets Produced by Name Entity Recognition

2.8.2.2 Coreference Resolution in GATE

Coreference resolution (CO) does two tasks in processing the first paragraph of the sample document presented in Figure 8 above:

- Pronominal or anaphoric resolution: CO identifies identity relations between entities in texts. CO would tie "him" and "His" with "Eliot Spitzer" and that means solving anaphoric references.

- Orthographic coreference resolution: CO matches "Eliot Spitzer" with "Mr. Spitzer" stating that they are same person.

Figure 11 shows the result of the anaphoric resolution identified by the ENTITY-MENTION-TYPE added to the Annotation set in Figure 10. Figure 12 shows the Orthographic coreference resolution.
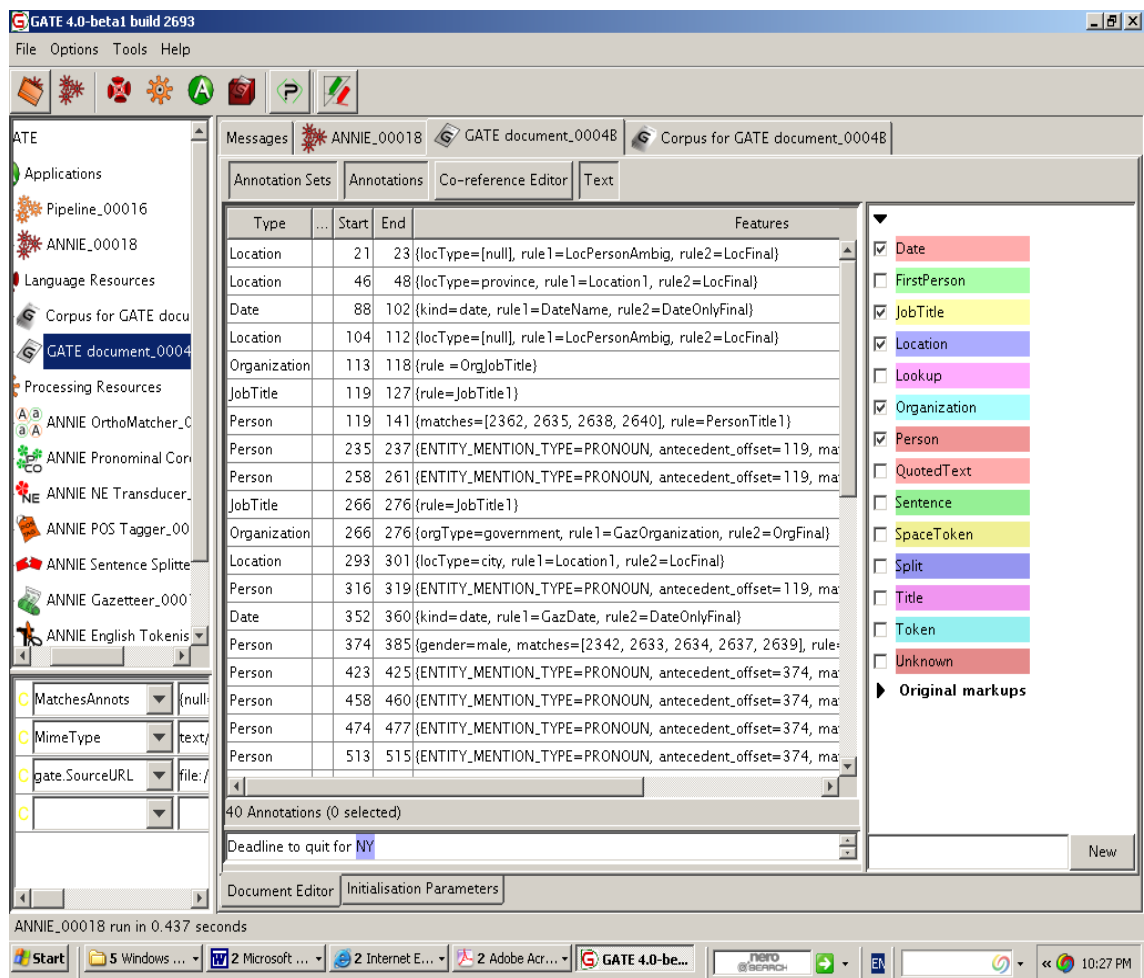
Figure 11: Coreference Anaphoric Resolution
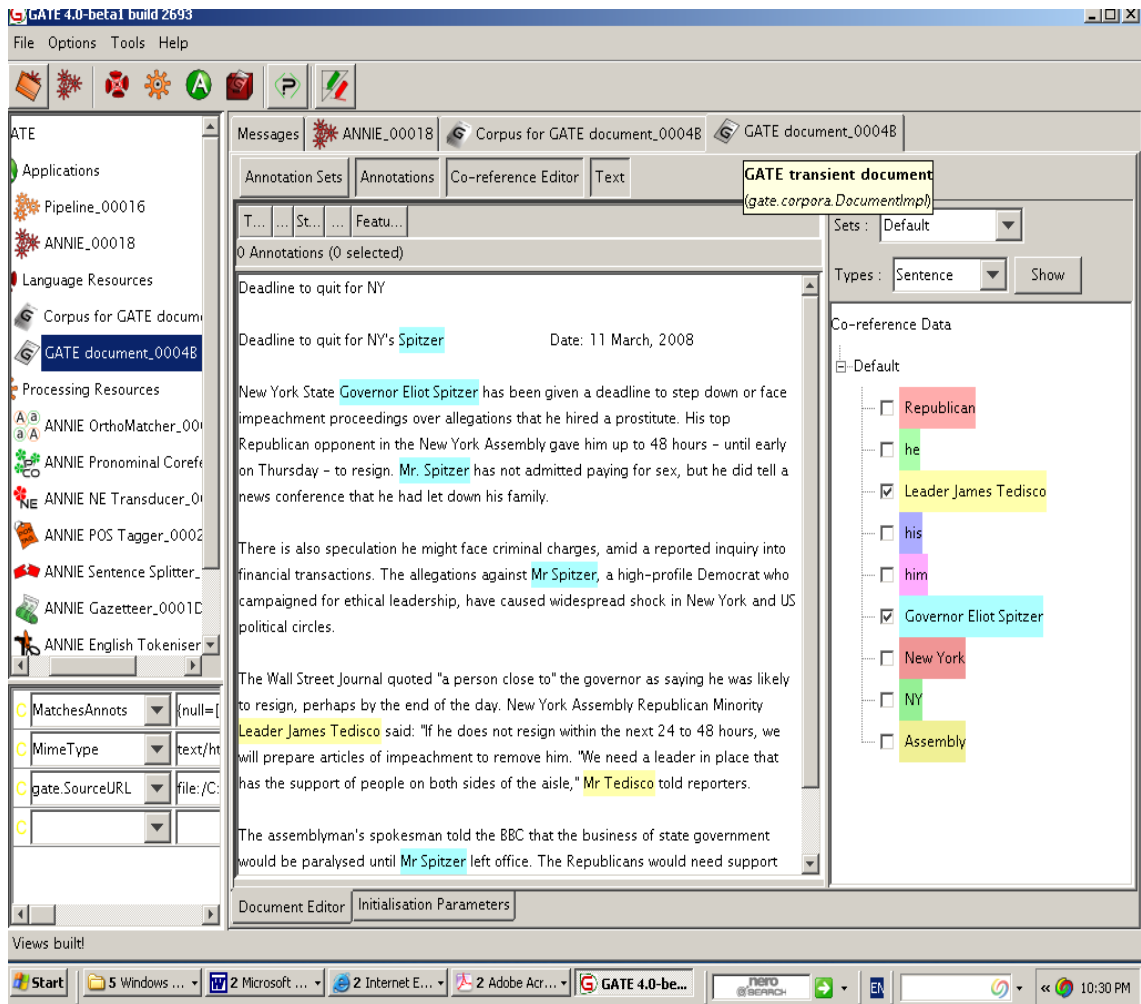
Figure 12: Orthographic Coreference Resolution

2.8.2.3 Outcome of the exercise

Although the GATE User Interface (GUI) is not very practical to use with a large number of documents, GATE is an open source and the API provides all the necessary Java code that can be included in user programs. The user can use all or part of GATE API to speed up the development of applications. We found that GATE performance in NE and CO is considerably accepted and we are encouraged to include GATE processing resources in the proposed framework that will discover and correct mistakes made by GATE. Template element, template relation and template scenario are normally event and scenario dependent and hence we did not, at this stage, investigate their use in this exercise.

2.9 Semantic Web Implementation Technologies

In addition to the tools discussed so far, many other tools contributed to the implementation of Semantic Web applications: Parsers, RDF validator, Resoners or Inference engines, RDF Query languages (RQL), RDF browsers, Crawler, etc. New and more advanced technologies have emerged recently, technologies like Jena [67] provide for most of the Semantic Web application process. In this section, some of the most popular RDF triple stores technologies and the Jena framework are briefly presented.

2.9.1 RDF Schema-based Repository and querying facility

RdF repositories are databases that usually have large amount of RDF triples extracted from Web pages and stored as a relational database. An RDF parser is usually supported by an Application Programming Interface (API), or RDF browser. The API is used to simplify querying the repository by using Query Language for RDF (RQL), simple Structured Query Language (SQL) or even programming languages like C, Java or Perl.

The most well known repositories are:

Sesame [23] is an open source system developed by Aidministrator Nederland as part of the On-To-Knowledge project in Europe. It is an architecture for creating efficient RDF repository and it provides querying facilities over large quantities of meta-data written as RDF and RDFS entries. A variety of storage devices, such as relational databases, triple stores, and object-oriented databases can be used by Sesame; it supports many databases like Oracle, PostgreSQL and mySQL. The Query Language for RDF (RQL) is used by Sesame when it was first developed because it offers support for RDF and RDF

Schema. RQL was developed by Institute of Computer Science at FORTH in Herakklion, Greece.

Sesame 2.0-beta6 was released on 12-10-2007 and can be downloaded from sourceforge.net website [22] it includes new features such as the support of SPARQL; the standard query language and a protocol for accessing RDF. Many projects have been initiated to improve Sesame and empower its use, projects such as "SesameJenaAdapter" aims at providing access to a Jena model through the Sesame API.

3Stores: Developed by AKT [3], it is a C library that uses MYSQL to store raw RDF data. 3Store Server does not expose any interfaces directly to the user, but it can be queried by a number of services, including a column based view and a direct RDF browser.

rdfDB: Developed by the R.V.Guha [90]. It is a simple, scalable, open-source database for RDF.

2.9.2 Jena

Jena is a Java framework for building Semantic Web applications. It is supplied by Hewlett Packard (HP) Labs Semantic Web Program in Bristol [59]. Jena is an open source. It includes Application Programming Interface (API) for OWL and RDF, a rule based inference engine, support for in-memory and persistent storage such as MySQL and Oracle databases, and supports the use of the query language SPARQL [107] to interrogate its modules.

Jena API includes methods for manipulating RDF triples, resources and their properties. It also provides support to Semantic Web implementation requirements like literal type, RDF containers and enhanced resources.

The reasoning subsystem in Jena provides different rule based inference engines, and provides for external resoners to be plugged into Jena.

Ontology data sources are handled by Jena as ontology model that is created as an extension of the Jena RDF model. This model can either be built from an existing ontology written in RDF, DAML+OIL or OWL or it can be constructed by Jena from scratch. The ontology subsystem includes a document manager that helps in managing imported ontologies.

SPARQL is a query language and a protocol for accessing RDF. SPARQL is designed by the Data Access Working Group of the World Wide Web Consortium (W3C) and became a W3C recommendation on 15 January 2008. SPARQL is built on top of RDF Data Query Language (RDQL) that was previously supported by Jena. SPARQL is supported by Jena since version 2.3.

SPARQL is "data-oriented" language in that it only queries the information held in the models using SELECT clause to identify the variables to appear in the query results and WHERE clause that specify a triple pattern. It returns the information needed in the form of a set of bindings or an RDF graph. It provides facilities to construct new RDF graphs based on information in the queried graphs and facilities to extract information in the form of URIs, blank nodes, RDF sub graphs, plain and typed literals.

Jena has been our choice for implementing this work for the following reasons:

- It is the most up to date technology.
- It has been used in many projects like GATE.

- It comes with good documentation and support.

Jena is provided with a wide range of worked examples and documentation that makes it possible for users to start using it and build Semantic Web application. In Chapter 4 we shall demonstrate more of Jena and its capabilities during the implementation of this work.

2.10 Agents

Artificial Agents are Programs that may exhibit the skills of acting autonomously, learn from the environment, and adapt themselves to new environments [37]. The Semantic Web is demanding a greater role from Agents; powerful reasoning capabilities are needed to meet the new demands. Agents on Semantic Web should be able to extract relevant information, "understand" and "reason" information on Web pages. Berners-Lee stated that

> *" the real power of the Semantic Web will be realized when people create as many programs that collect Web content from diverse sources, process the information and exchange the results with other programs"* [16].

Artificial Agents then, enable the Semantic Web to behave as a system that is built on a collection of Agents. Such Agents express human like actions. In [24], it was reported that an Agent should possess goals, intention, behaviours and belief. Agents can become more effective when they find more machine-processable information on the Web.

Agents of Semantic Web are expected to perform each of following roles:

Searching: Where Agent is expected to provide relevant answers for users queries.

Service Discovery: The Agent must be able to automatically find the appropriate Web service, where Web services properties and capabilities are also described somewhere on the Semantic Web.

Proofs: Agents should be able to translate their internal reasoning into a proof delivered to the user or to another Agent; this explains how that Agent reached the results presented.

Digital Signature validation: Agents are expected to play a greater role in verifying the source of the digital signature.

Figure 13 shows the well-known pyramid of the levels of the Semantic Web [13], the top level is that of trust that is expected to be achieved via the use of Agents and proper Web services in addition to the digital signature that helps in all layer of the Semantic Web pyramid.
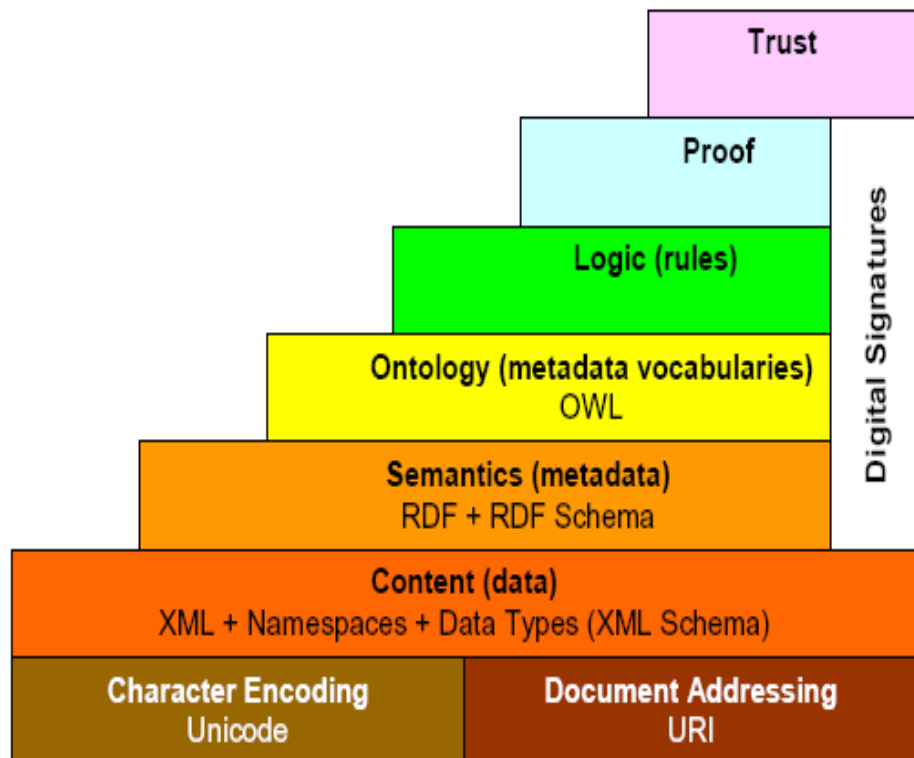
Figure 13: The levels of the Semantic Web

2.11 Web Services

In the scope of this work we do not involve ourselves in the details of Web services; we assume that once the implementation works on the Intranet, Web services and the interaction with the users on the Internet will be an added-on facilities.

We assume that many Web services will be developed and used by the users. Different services will be available from many heterogeneous sources, broker services might be used to best understand and decide on the services needed.

2.12 Conclusions

We found that the subject of Annotation is a very active research area. A great deal of effort and resources are required to annotate any of the old or new websites in the way required by the Semantic Web.

The meta-data needed to make a Web document "intelligent" provides both class and instance information for each entity in the text. Figure 14 below presents the semantic Annotation as illustrated in [69]. The dotted arrows represent links, the meta-data, between the entities in the text and the semantic description as specified by the ontology in the KB.
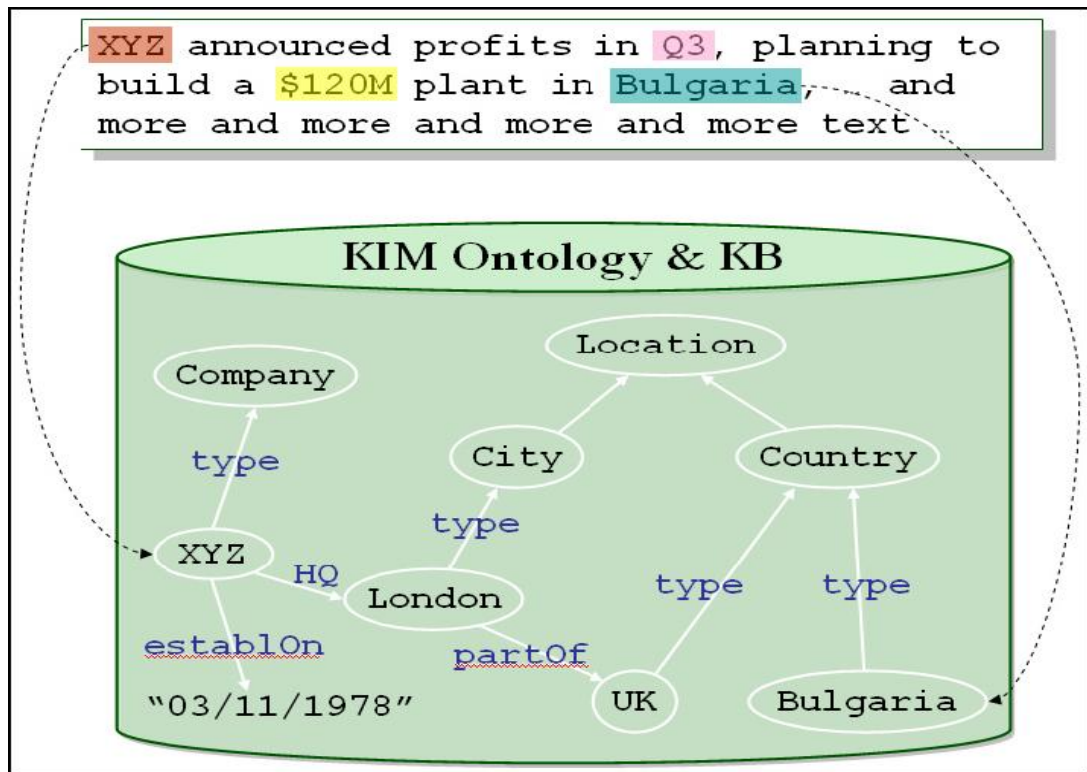
Figure 14: Sample Annotation and KB contents

Information Extraction is the main concept behind the process of Annotation for Semantic Web. Most of the current Annotation tools rely on manual or semi-automatic Annotation. Successful implementation of Semantic Web requires automatic Annotation.

Although there are standard evaluation measures for information extraction, the evaluation process of any information extraction system is very difficult and requires a lot of time and resources to build the suitable corpus that is needed for testing. The performance of the developed systems is still questionable and none of the available systems can be regarded as the best because the implementation is normally affected by the domain, the type, and the number of training documents used with systems that rely on machine learning.

We presented several Semantic Web implementation technologies, we believe that there has been great success in introducing such technologies especially those that have

been regarded as standard by the W3C; such technologies have made it possible for research like this work to proceed.

In this chapter we make three main contributions. First, we showed how the concept of Thematic Role together with the list of semantic classes of verbs can play key roles in the process of information extraction needed for Semantic Web. Second, we use a case study to investigate the use of GATE as HLT that can be used to prepare for the suggested Semantic Web implementation; the case study showed that GATE is the appropriate choice for Named Entity Recognition and Coreference resolution needed for the pre-processing of text. Third, we investigated many of the current Annotation tools, we have observed that some challenging issues in this area have not been addressed yet, we found that most of the current Annotation tools rely on user manual intervention which normally includes too many shortcomings that lead to inaccurate Annotation. We also found that, tools that rely on Machine Learning algorithm are still a way ahead from being suitable for Semantic Web. The top performer, Armadillo, uses LP^2 algorithm that relies on the redundancy of pages on the Internet to verify and adjust its performance. With Semantic Web such reliance on the Internet contents can hardly be accepted for verification.

We conclude that the solution to the Knowledge Acquisition problem and the production of "intelligent" Web documents with high degree of Annotation accuracy and automation can only be achieved using a framework for domain dependent automatic annotator that utilizes the context in which the documents are generated; We suggest such a framework that will be investigated in detail in the next chapter.

3. Framework for Semantic Web Implementation

3.1 Overview

This chapter introduces the suggested framework for Semantic Web mining and implementation, for short, we call it the Semantic Web Implementation Framework (SWIF). The key issues and limitations concerning the design and implementation of the proposed framework are presented in the next section. The architecture of SWIF is described in Section 3.3.

A case study has been used in Section 3.4 to demonstrate the implementation of the suggested framework. The case study provides a brief description of the current web-based application at Sultan Qaboos University (SQU) and proposes a conceptual architecture for the Semantic Web based systems, it also introduces the implementation plan according to the suggested framework.

3.2 Key Issues of the Semantic-Based Implementation

The aim of this work is to provide website developers with a framework that enables them to implement semantic-based applications on the Intranets or Internet and verify the implementation. The idea is to build semantic based applications regardless of the actual existence of world wide Semantic Web; this will give the Semantic Web a better chance to grow because individual sites developed using the suggested framework will be ready to join the global Semantic Web implementations through Agents and Web services.

The key issues and limitations concerning the design and implementation of the proposed framework can be clarified as follows:

- The work is based on the fact that most websites on the Internet work with their full functionality on the owner's Intranet; the possibility of using Web Services and Agents has made it feasible to assume that what works on the Intranet will definitely work with, at least, the same functionality on the Internet. The philosophy behind building stand-alone semantic-based applications is that the whole application will be developed, verified and tested in the application context; this will ensure better quality of Annotation and Information Extraction.

- The scope of implementation test is to include part of the Web-based applications and services provided by any academic institution. In particular, it will be tested on SQU data aiming at creating and maintaining an RDF repository of experts and providing meta-data to new Web pages related to academic staff.

- Although the framework proposed is general, the implemented case study will not include such capabilities of annotating Arabic Web pages. Such pages are included in the implementation after being annotated by the manual annotator OntoMat. This is because currently we did not include language resources for Arabic language.

- The key point in handling Web pages in this framework is that, a Web page is annotated before it is uploaded to the website. This method ensures adequate Annotation because pages are annotated before being affected by noise; such as the contents of the page frames, the advertisements added to the page in a later publishing process, and network noise.

- The approach employs a fully unsupervised algorithm that relies on what we call Control Knowledge base (CK). The CK acts as a semantic model for Annotation

where it represents the domain memory, new knowledge is continuously learnt and added to the CK. Learning is based on the number of occurrences of the same piece of information in several documents.

- This implementation uses the SQU staff homepages and the biography pages that are already written by each researcher together with the digital abstracts of the staff publications.

- Arabic names are manipulated using special procedures. People's names in Arabic can be represented with different English string representations; for example, names like "Hatam", "Hattam, "Hatim" and "Hatem" are all acceptable representations of the Arabic male name "حاتم". Other representations of the same name can also be used; such representations are usually known to the person involved and may not be predicted in advance. Special procedures are needed to ensure proper manipulation of Arabic names.

3.3 The Architecture of the Semantic Web Implementation Framework (SWIF)

The proposed framework for Semantic Web Implementation that is based on context-oriented controlled automatic Annotation can be described through its implementation system. To be able to use a shorter name for the framework, we use Semantic Web Implementation Framework (SWIF) and we named the system that implements this framework as the Semantic Web Implementation System (SWIS).

SWIF is an architecture for Semantic Web implementation of stand-alone websites that automatically annotates the Web pages before uploading to the Intranet, it

maintains persistent storage of RDF data for both the domain memory, denoted by Control Knowledge, and the meta-data of the Web pages on the website. SWIF provides for semantic based queries issued by Intranet and Internet users on both its RDF repository and its annotated pages. It enables Internet user to access the implementation via Agents and Web services which make SWIF implementations as part of the global Semantic Web.

Figure 15 illustrates the SWIF's general architecture, it consists of 4 key phases. The first concerns the construction of the Control Knowledge. The second concerns the pre-processing of documents by HLT that is used to extract entities of interest together with related information such as the Part of Speech (POS), stem, etc. of each component in the document. The third is the information extraction and semantic processing that enables management and storage in the RDF repository; this phase also produces the annotated version of the document being processed. The fourth phase uses the global and local user interface that enables users to submit documents for processing and enables researchers to update their profiles and contribute with information needed for the multiple representations of their bi-lingual names. The interface also provides Web front ends that enable retrieval and query; several services can be used by the different Agents to do the semantic based retrieval and processing.
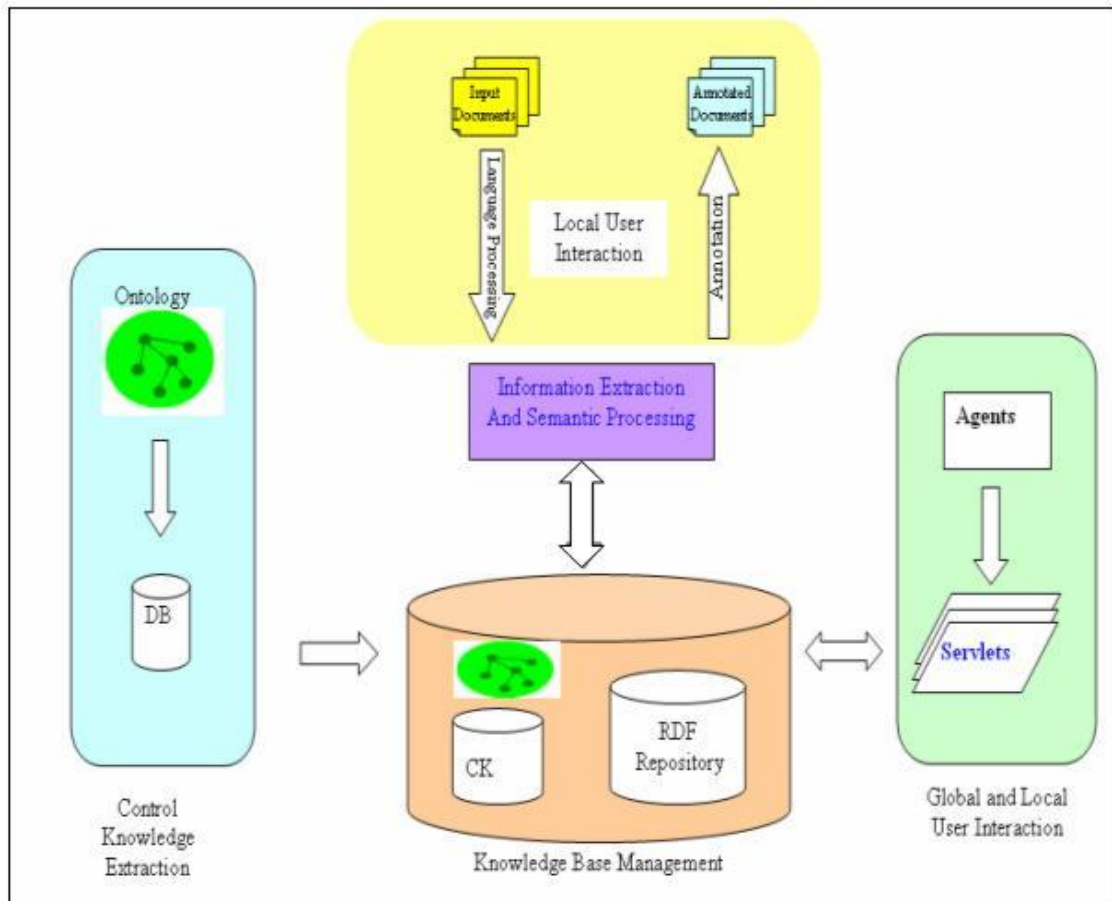
Figure 15: The Architecture of SWIF

3.3.1 Construction of the Initial Control Knowledge

The input to this stage is the database or data files of the Personnel System at the academic institution together with the research publications that can be found in digital forms. Guided by the appropriate ontology for that particular institution, the programs in this phase wrap the input data structures into RDF triples written on an intermediate RDF data file, the gazetteer lists are generated at this stage too. The CK constructor program converts the RDF data file to persistent storage as a Jena model implemented using Oracle database. Appendix D shows a sample RDF data file.

3.3.2 Pre-processing using Human Language Technology (HLT)

There are currently several HLT tools that can be used to process the input documents. GATE components are used in this work for several reasons: GATE is widely used by the Semantic Web community, it is maintained by a dedicated team of researchers, and it is an open source project that allows us to alter some components and tailor it to this work's needs.

In this section, GATE components are described to illustrate the minimum processes that are needed by any HLT tool selected for the implementation of SWIF-based system.

3.3.2.1 Unicode Tokeniser

The tokeniser is used to split the text into tokens like words, punctuations, symbols, spaces, and numbers. The tokeniser assigns a token kind to each type of word, it also specifies the length of each token, with special identification to the spaces in the text as SpaceToken . It uses rules to identify and assign a value for the attribute "orth" that can be "upperInitial, allCaps, lowerCase, or mixedCaps" to denote whether the initial letter of the word is uppercase and the rest are lower, or the word's letter are all in uppercase, lowercase or mixed. The tokeniser produces both Annotation of type SpaceToken or Token together with a list of features as seen in the following example:

```
Type=SpaceToken; features={kind=control, length=1, string=}
Type=Token ; features={kind=word, orth= upperInitial,
length=7, string=Britain}
```

3.3.2.2 Gazetteer lists:

They are group of lists compiled into finite state machines. Each list is a plain text file that contains one entry per line. Individual list represents set of words like the set of the names of countries, currency names …etc. Normally each list contains all possible representations of a name for example "United State", "US", and "United State of America" are all included in the country.lst file as three separate entries. The index file "lists.def" is used to access the gazetteer list; this file contains an entry for each list in the gazetteer where the first field in the entry denotes the list name, the second denotes the major type, and the third denotes the minor type for example the entry country_abbrev.lst:location:country_abbrev specifies the list country_abbrev.lst that denotes a location major type with country_abbrev country abbreviations as minor type. The country_abbrev.lst contains entries like "UK, USA, AUS…etc " each abbreviation on one line. The default gazetteer processing resource "DefaultGazetteer" produces a Lookup Annotation and features for example if the text contains country like "USA" the gazetteers list are searched by the DefaultGazetteer to find all occurrences of a matching entry and when it finds it in the country_abbrev.lst it produces the followings:

```
Type= Lookup; features= {minorType = country_abbrev ,
majorType = location};
```

3.3.2.3 Sentence Splitter

This processing resource is a cascade of finite-state transducers that segments the given text into a sequence of sentences. It is implemented by the **SentenceSplitter.java** that uses the output of the previous two processes looking for sentence-marking identified by punctuation, line breaks or a series of special domain dependent punctuations such as "?!?". The sentence splitter outputs two types of Annotations; the Annotation of type

sentence is associated with each sentence and the Annotation of type **split** is given to each sentence break like punctuation.

```
Type=Split; features= {kind=external};
Type=Sentence; features={};
```

3.3.2.4 Part of Speech Tagger

The part of speech (POS) tagger used in GATE is the Hepple tagger which is a modified version of the Brill tagger [21]. A default lexicon and rule set are used by the tagger to produce a part-of-speech tag for every token found by the tokeniser. The tagger adds a category feature to the list of features a token have. The tag denotes the part of speech for that particular token. For example the NNP tag , that denotes a singular proper noun according to the list of tags used in the Hepple tagger, is added in the following example as a category feature.

```
type=Token; features={category=NNP, kind=word, orth=
upperInitial, length=7, string=Britain};
```

3.3.2.5 Semantic Tagger (ANNIE Transducer)

The Semantic tagging in GATE is achieved using a Java Annotation Pattern Engine (JAPE) grammar which is based on Common Patten Specification Language (CPSL). JAPE grammar runs on data stored in the Annotation graph produced by GATE, it treats such a graph as a simple sequence that can be matched with regular expression. JAPE set of pattern–action rules run sequentially forming a cascade of finite state transducer over Annotation. Each rule in JAPE consist of left-hand-side (LHS) and right-hand-side (RHS) separated by the symbol "-->" . The LHS contains the Annotation pattern and/or the regular expression it contains, the RHS specifies the action to be taken written as

Annotation manipulation statements. For example the following JAPE grammar is used to locate previous lookup Annotation with majorType **topicidentification** and generate new Annotation of type **topic** at the same time the action includes java statements that remove the old lookup Annotation from the final Annotation graph.

```
Phase:     topicFinder
Input: Lookup
Rule:subjectConverter
({Lookup.majorType == "topicidentifcation"}):match
-->
:match.topic = {rule = "topicrule"},
{
  Annotation lookupAnn = (Annotation)
    ((AnnotationSet) bindings.get("match"))
    .iterator().next();
  inputAS.remove(lookupAnn);
}
```

The JAPE grammar is a very powerful tool that is used to produce relation instances and Annotations of several forms. JAPE rules can be given a priority using either explicit priority, the length of the matching pattern, or the order of executing the rule. It allows not only for Java code to be written in the RHS of the rule, it also allows context of the pattern to be expressed in the LHS; for example we can have a rule to specify email address as having {Token.string == "@"} between the words of the address as specified in email.jape provided by GATE.

3.3.2.6 Orthographic Coreference

The OrthoMatcher processing module identifies entities that are previously found by the semantic tagger as both of the same type, e.g. person names, or one of them is tagged as "unknown". For example the orthomatcher discovers that "Tony Blair" and "Mr. Blair"

are referring to the same person in the document, i.e. there is a match between these two instances. Matches features are added to the Semantic tagger Annotation as in the example that follows.

```
type=Person; features={gender=male, matches=[4453,4463],
rule=PersonTitle1}
```

3.3.2.7 Pronominal Coreference (Coreferencer)

Anaphoric resolution is done by the coreferencer; it runs over the Annotation produced by the previous processing resources and produce two types of features **antecedent_offset** and the **matches** list that denotes all the pronouns used to refer to a particular object like the person in the next example.

```
type=Person;   features={ENTITY_MENTION_TYPE=PRONOUN,
antecedent_offset=119, matches=[5166, 5147, 5156, 5146]}
```

In addition to pronominal module for coreference resolution the coreferencer includes two other modules that deal with **quoted text** and **pleonastic it** and produced special Annotation type that is used later by the pronominal module.

3.3.2.8 The Stemmer (SnowballStemmer)

The SnowballStemmer used for English language in this work is a plugin processing resource that is based on the Porter algorithm [88]. The implementation of the stemmer's rules is in Snowball which is a string processing language that is normally used for creating stemming algorithms. In the following example the stem "play" is found for the token "played".

```
<Token gate:gateId="13" category="VBD" string="played"
stem="play" kind="word" orth="lowercase" length="6">
```

3.3.2.9 Noun Phrase Chunker (NP_Chunking)

The NP_Chunking processing resource is used to produce **NounChunk** Annotation over the output produced by the tokeniser, sentence splitter and the POS tagger. The NounChunk Annotation do not have feature set associated with it; the start and end of the noun phrase that accompany the NounChunk Annotation is the information required for identification of the noun phrase borders. Example of such Annotation is:

```
type=NounChunk; features={}; start=NodeImpl: id=2;
offset=0; end=NodeImpl: id=3; offset=13
```

3.3.2.10 Verb Group Chunking

The VerbGroup.jape chunker is a rule-based chunker written in JAPE. It is loaded as any other JAPE grammar and executed to identify all different verb constructs. The methodology used by GATE is based on the grammar of English. The chunker does not only produce Annotation type with VG value, it also produces features like the voice, the tense and the type of the verb. Example of the verb group chunker is:

```
type=VG; features={voice=active, tense=SimPas, type=FVG};
start=NodeImpl: id=12; offset=19; end=NodeImpl: id=13;
offset=27
```

Where, FVG denotes Finite Verb Group classification adopted by the JAPE rules, SimPAS denote Simple past tense.

3.3.3 Information Extraction and Semantic Processing

The output from the GATE pre-processing components is an XML document that contains partial Annotations such that:

- Not all NE Annotations performed by GATE component are correct, a verification phase is needed.

- Relation extraction is not performed by the default processing components, relation extraction is the main weakness in all of the present Annotation tools, and hence we need to build our own inference engine that enables us to extract relations with the help of the CK created and the domain ontology.

Information extraction and Semantic processing in SWIF contains additional language processing on GATE outputted XML documents, it also includes an inference engine for semantic processing. SWIF functionalities are illustrated in this section as follows:

3.3.3.1 Manipulation of Individual Sentences

The aim of the following components is to convert the complex and embedded sentences into simple sentences that can easily be handled by the semantic analyser. Sentence manipulations involve chunking the sentences into simple clauses and then identify the subject of each sentence as follows.

3.3.3.1.1 The Clause Segmenter

By definition, a clause is any structure that contains its own verb [15]. A simple sentence is an independent clause. With human languages, recursion is possible when one clause occurs inside another which makes it very difficult to parse and extract relations in a correct way.

For example in the following sentence, the embedded clause changes the simple sentence to a more complex simple sentence.

> *The head of the department, determined to solve the problem, made several changes to the program.*

Another more complex example of embedded sentences from [15] is illustrated below. Figure 16-a shows the functional category components of the following sentence.

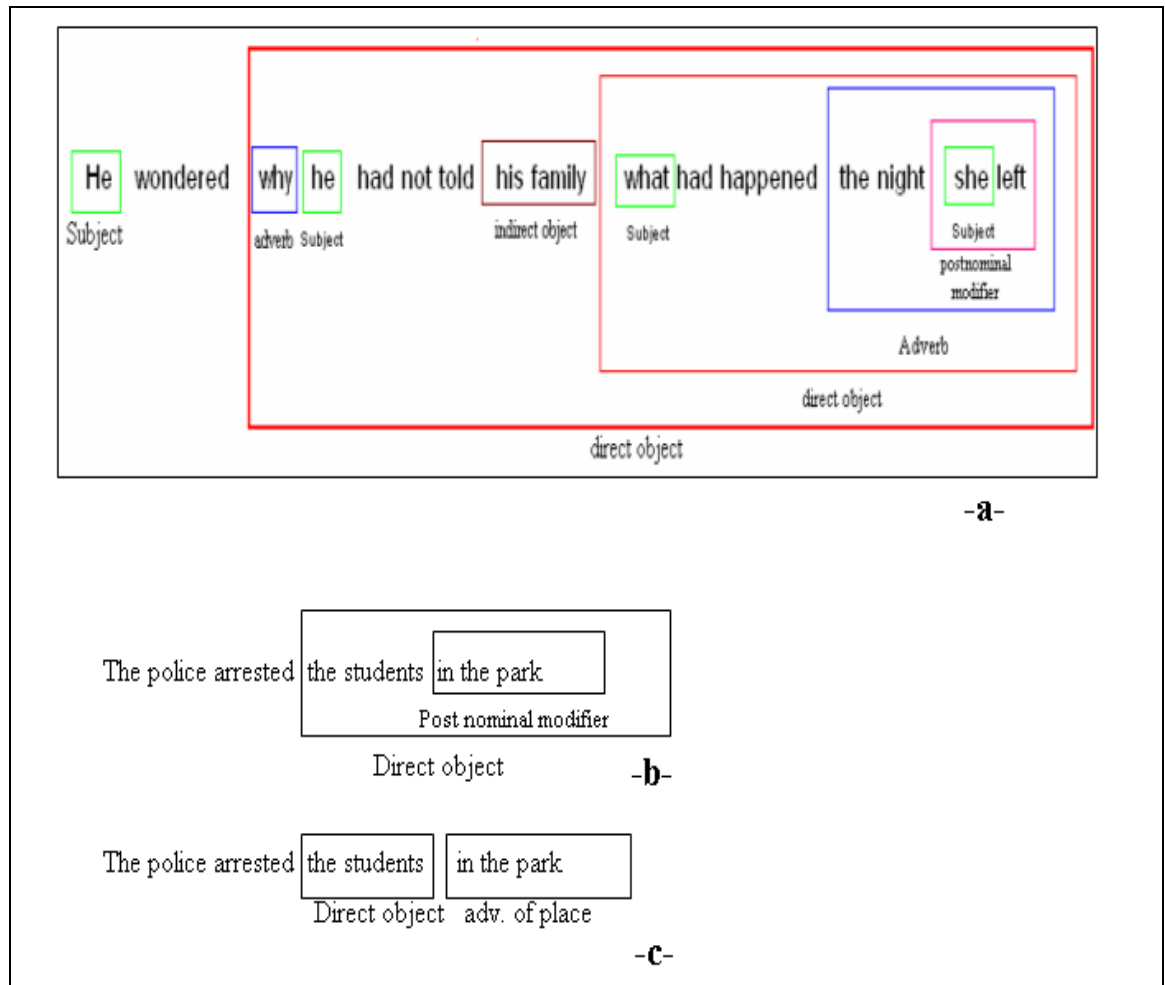*He wondered why he had not told his family what had happened the night she left.*

Figure 16: functional category components and meaning interpretation

Figures 16-b and 16-c show sample sentence to demonstrate the different meaning interpretation of the same sentence.

To segment sentences of the samples above, full parse and more complex algorithms together with a lexicon should be used. For the purpose of this work we are limiting clause segmentation to simple and compound sentences because this is what is being

73

mainly used in the domain we deal with, normally either simple or compound sentences are used and these are easier to handle than embedded sentences that need processing beyond this work. This work is also limited to declarative statements because such statements are used to make the sort of statements normally found in the biography corpus and they use expected order of functional categories (Subject, Verb, Object, etc.) that requires easier processing than with other types of statements..

The function of the clause segmenter we developed is to convert compound sentences into simple clauses where each clause contains at most one main verb with possibly an infinitive clause. The segmenter scans the text from left to right looking for the second main verb in the sentence, a predicate conjunction, or a statement conjunction like the words "and", "or", and "but". If it finds one a new clause is started.

This sort of segmentation prepares for subject and object identification. By the time the system executes the clause segmenter, the shallow parsing of the input text would have been done and the selected entities annotated by the pre-processor will only be examined by the clause segmenter. For example given the following sentence, the segmenter will process the sentence as shown below:

> *Dr Khaled Day received an undergraduate degree in computer science from the University of Tunis in 1986 and the M.Sc. degree from the University of Minnesota in 1989.*

This will be chunked into two clauses:

1- *Dr Khaled Day received an undergraduate degree in computer science from the University of Tunis in 1986*

2- *and the M.Sc. degree from the University of Minnesota in 1989*

3.3.3.1.2 The Syntactic Role Identifier

The Syntactic Role Identifier uses the grammatical structure of a sentence to identify the subject and object of each clause. For example in the case of an active sentence, the

subject must precede the verb phrase, direct object and indirect object succeed their verb phrase, and the infinitives may have their own direct object.

This step prepares for the relation extraction. The clauses produced by the clause segmenter are examined to locate the main verb, the subject and object. Only those annotated entities are considered for identification. The Sundance [91] method of identifying the missing subject of a sentence is used to assign one of the noun phrases from the preceding clause as a subject of the current clause. In addition we handled such cases where both subject and verb are missing as in the example above. In such a case we propagate both the verb and subject to the new clause instead of "and'.

3- *Dr Khaled Day received an undergraduate degree in computer science from the University of Tunis in 1986*

4- *Dr Khaled Day received the M.Sc. degrees from the University of Minnesota in 1989*

Both simple sentences in 3 and 4 are converted to the following form:

Person received **Degre**e from **Organizatio**n in **Date**

Consider the special construct in the following sentence that uses "Respectively".

Dr Khaled Day received an undergraduate degree in computer science from the University of Tunis in 1986 and the M.Sc. and Ph.D. degrees from the University of Minnesota (USA) in 1989 and 1992 respectively.

This sentence after being processed by GATE takes the following form:

Person1 *received an **Degree1** in **Subject1** from the **University1** in **Date1** and the **Degree2** and **Degree3** from the **Unversity2 (Country1**) in **Date2** and **Date3** respectively*

The clause segmenter and Syntactic Role identifier can not directly process such a sentence, a special JAPE Rule **Respectively_Rule** was added to the GATE transducer to convert this sentence to the suitable clauses. The word respectively is removed and the sentence is converted into the following three clauses like:

*Person1 received an **Degree1** in **Subject1** from the **University1** in **Date1***

and the **Degree2** from the **University2 (Country1)** in **Date2**.

and **Degree3** from the **University2 (Country1**) in **Date3.**

Then the syntactic role identifier assigns the subject and verb needed for each clause as follows:

*Person1 received an **Degree1** in **Subject1** from the **University1** in **Date1***

Person1 received the **Degree2** from the **University2 (Country1**) in **Date2**.

Person1 received **Degree3** from the **University2 (Country1)** in **Date3.**

3.3.3.2 Semantic Analyser

There are a number of different approaches to achieve Semantic Analysis [68]; first, the **syntax driven approach** where meaning is represented depending on the meaning of the text parts and the static knowledge from the lexicon and the grammar. Such a type of representation is normally used in machine translation. Second, the **Semantic grammar approach** where the elements of the grammar are strongly motivated by the entities and relationships of the domain. This approach is normally used with interactive dialog systems. The final approach is the **information extraction approach** that is used where a small amount of information is to be extracted from a large amount of information. An information extraction approach normally adopts a series of finite state automata cascaded in a way that leads to a robust semantic analysis. The information extraction approach has been adopted in this research.

As seen in the previous sections, part of the semantic analysis has been done in the pre-processing phase by GATE where entities have been identified. At this phase the semantic analyser's role is to extract relation instances, it categorizes the noun group

and verb group instances to those concepts of the ontology. This involves each of the following tasks:

- The identification of the **thematic role** of the verb's arguments; it is important to identify the entity that is the "doer" of the action and the entity that is being acted on by the verb. These two roles are very important for relation identification. Hence, a special lexicon is created and used; this is the **Syntactic and Thematic structure lexicon.** This lexicon includes the argument structure of the verbs, their categorical status, and the number and types of thematic role they assign to the argument structure. Examples of entries in this lexicon are the two entries created for the verb give in the following two sentences:

    John gave Mary the book . .... (1)
    John gave the book to Mary. .... (2)

    The lexicon entries are:

    Give: [1 <NP, Agent>, 2 <NP, Beneficiary>, 3 <NP, Theme>] ……………entry 1
         [1 <NP, Agent>, 2 <NP, Theme>, 3 <pp>, 4 <NP, Beneficiary>]…….entry 2

- The second important function here is to relate the main verb in the clause to its corresponding **semantic class of verbs**; such a classification relates a group of verbs to one property in the ontology. For example verbs like receive, get, obtain, awarded, etc) are all classified as one class to correspond to the property has-degree that appears in the domain ontology. Such classification has already been done by linguists; Levin classified over 3,000 English verbs according to shared meaning and behaviour [73], Levin's classified list can be used in our system, but for the sake of this work we shall construct such a similar simple lists as full lists are not obtainable to us at this stage.

- The third is the function that deals with the notion of **verifiability**. Jurafsky and Martin highlighted the importance of verifiability of meaning representation; they stated that the straightforward way to implement the notion of verifiability is to compare the representation of the text's meaning with the representation in the knowledge base [68]. In this work, extracted entities and relations are verified by the use of CK and verifiability table.

3.3.3.3 Knowledge Base Management

The knowledge base consists of three different modules: the Ontology, the Control Knowledge, and the RDF repository. Any identified piece of information that constitute an entity or a relation is checked with the control knowledge for verification and then added to the RDF repository following the algorithm that is presented in Section 4.7 of the following chapter.

3.3.3.4 Annotation Generation

Each identified relation and each entity verified by the semantic analyser results in adding Annotation to the annotated version of the document. The RDF annotated document is generated as a result of the semantic analysis process.

3.3.4 Global and Local User Interface

The global and local user interface provides for the interaction with the implementation using either of the following two ways:

Portal or any Web Front End: the Semantic Search Engine (SSE) can be invoked and used for querying the RDF repository. The repository is used to answer any query

related to the implementation. The web front end used enables users to submit documents for processing. In addition, it enables researchers update their profiles and contribute with information needed for the multiple representations of their bi-lingual names.

Agents, Web services, and external Semantic Search Engines: external users are allowed to access part of the RDF repository in addition to all of the annotated pages. The annotated pages will be made available on the Web and can be accessed by any of the external SSEs or Agents that work on annotated pages.

3.4 Case Study: Development of Semantic Web Application for Academic Institution

3.4.1 Introduction

Sultan Qaboos University (SQU) is one of the leading academic institutions in the Arabian Gulf; it is the only governmental university in the Sultanate of Oman. The first Sultan Qaboos University students were enrolled in 1986 and started with 516 students only, in 2007 the number rose to 14722 registered students. Currently the university have nine colleges; namely Medicine, Engineering, Agriculture, Education, Science, Arts, Commerce, Economics, Nursing, and Law. The University aims at playing a considerable role in the academic sector in the Gulf region in particular, Arabic and international countries in general.

SQU is fully financed by the Omani government, which aims at adapting the state of the art technologies in all of SQU's colleges. The government is supporting and

encouraging scientific research in the university to enable it to be among the best universities in the region.

The University has many advanced Web-based systems like the student registration system and employee system, the library system with its links to international services, the e-learning portals, together with a large number of static pages that represent the websites of the different colleges. More than 70% of Web pages on the site are in English language, and almost half of the pages in Arabic have an English version.

The university is continuously modifying its Internet services not only to provide better services and facilities for its staff and students, but also to cope with the demands of information needed from SQU. The current university site stands short in answering many queries that can lead to possible cooperation with other universities and research bodies in the region. For example, a university project in Saudi Arabia is looking for experts in the region who worked on "Water Resources" for a certain case; such information is difficult to find with current Internet service. With Semantic Web, such information can be directly provided and people involved could be located and contacted without having to wait for a long time going through the paper work communications with the administration that may or may not manage to deliver the accurate information in a reasonable amount of time. Most of the information about this case study has been published in [57] as part of this research work.

3.4.2 Sultan Qaboos University Web-based Applications

Currently SQU provides the following Web-based applications:

- **The university website** where each college and administrative department has their own website. Web pages are mainly static, a great deal of effort is required to update and maintain them.

- **Students Information System (SIS)** where students and faculty can use this service for on-line registration and different reports and statistics.

- **The Employee Information System (EMPINF)** is used online by administration and SQU employees. EMPINFO helps in providing leave information, employee salary details and other information.

- **WebCT** and **Moodles** are the E-learning portals used at SQU.

- **The University libraries system** provides on-line access to various library materials and international databases.

In short, SQU Web services use heterogeneous data; semi-structured data is found on the static Web pages, structured data is found in the Students Information System, Employee Information System, and the Library System in addition to the E-learning portals data files.

3.4.3 Our Approach to SQU Semantic Web based Implementation

The conceptual architecture of our proposed system is shown in Figure 17. It presents the overall SQU semantic based architecture. The implementation of this structure is a multi step process comprising the following:

- **SQU Ontology Creation:** The knowledge engineer creates and maintains SQU ontology. The Ontology learner captures new concepts and updates the ontology accordingly.
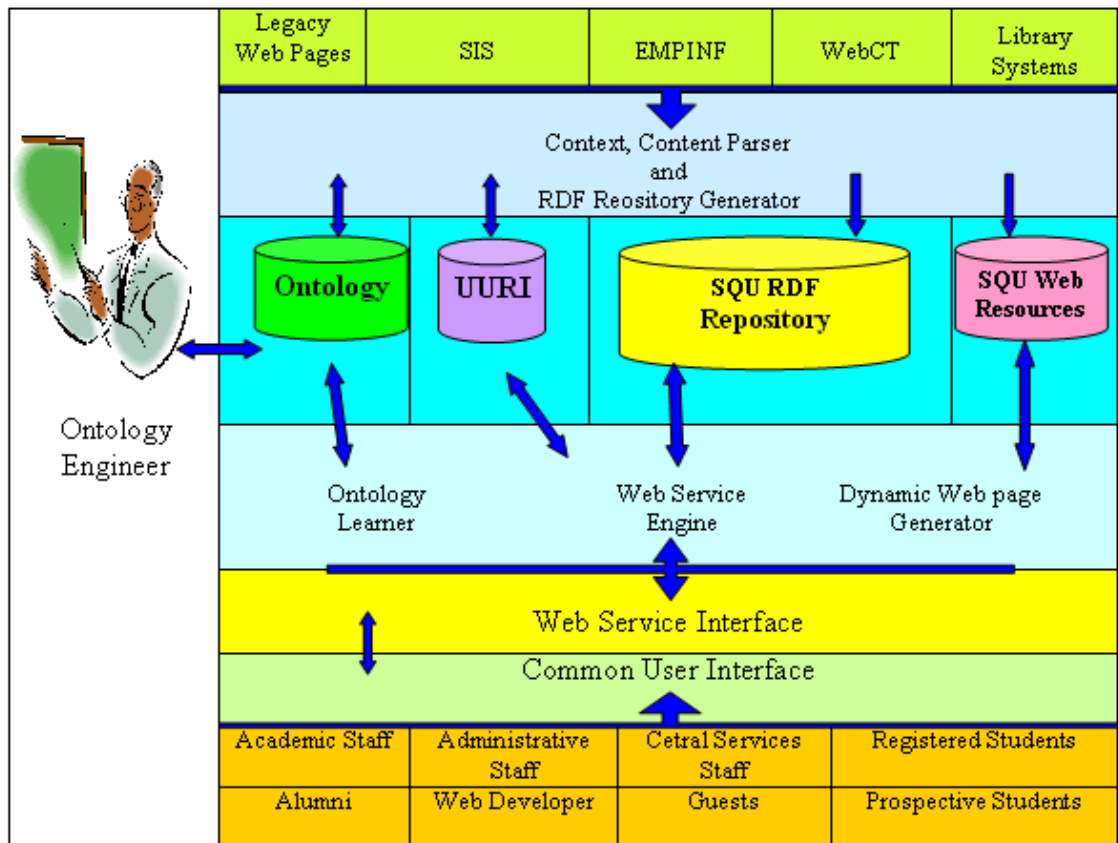
Figure 17: The Conceptual Architecture of SQU Semantic Web based System

- **RDF repository:** the integrated RDF repository is to be constructed. It includes RDF triples of every Web page that can be provided by any of the subsequent services. The context and content parser will be used to generate and update the RDF repository. The Web services engine captures new RDF and updates the RDF repository. Off-line updates on any of the databases involved can be imported and processed by the context and contents parser on a regular basis.

- **SQU Web Resources:** This is the warehouse that contains all the Web page templates and the multi-media resources needed by the Web page generator. This warehouse is created by the context and content parser and updated by Web developer.

- **Unique Uniform Resource Identifier (UURI):** The central service staff assign UURI for every user at SQU and follow certain method to allocate such URI for every resource like country names or university names, etc. such URI does not necessary map to any actual Internet address.

- **The Web Services Interface:** It represents the Semantic Web portal. The dynamic Web generator displays a portal page for each user according to the user profile that is automatically constructed from the KB and the user interaction with the system.

- **Software Agents and Web Services:** Access to SQU RDF repository and Web resources can be permitted to software agents; they use query language, such as the object-oriented Query Language (SPARQL), to provide the functionality of querying the RDF instances in the repository.

- **Common User Interface:** SQU Web users access their pages via the common user interface. The various users of the website are required to enter their username and password. The users' profiles database is checked for authorized access. The Web service interface displays the suitable user interface for each user depending on the decision made by the Web service engine.

3.4.4 General Implementation Plan

Implementation of the Semantic Web application on the case study selected will be in steps:

- The first step in our implementation plan is to implement and test the system on a test case that covers only a portion of the university Web services and that is building RDF Repository of Experts and the generation of Annotated Web pages.

- The second step is the e-learning services at SQU, this needs special team at the university to cooperate in selecting the courses required and implementing the application following the successful test in first step.

- Third step will include forming special research group to provide for creating a new version of the automatic Annotation tools developed in this work to handle Arabic pages. The team will also work on providing all necessary language resources and contribute in modifying the SQU ontology.

- All other parts of the current SQU Web-based applications can then be gradually converted to semantic-based implementation.

It is expected that this work will require 2-5 years to convert all SQU services to the new system. This work can be used as a model for any academic institution especially those in the Gulf region.

3.4.5 Maintenance of User Profile

The maintenance of User Profiles represents another major activity. The Unique Uniform Resource Identifier (UURI) we suggested in [54] is used to allow for multiple representations of multi-script researchers' names. We claimed that such UURI is required to uniquely identify each user. We suggested a practical method for creating and maintaining user profiles for the Semantic Web; the idea is to have UURI for every user and provide users with the ability to update their profiles. This is demonstrated by the screen shot example in Figure. 18 which shows that non-English names can be represented with different strings, so an Arabic name like "حاتم" can be written as "Hatem", "Hatam", "Hattam", "Hatim", etc. and others. The existence of all these name values for a particular expert in the RDF repository denotes that all these different

representations are to be treated equally only for that particular person who holds the UURI associated with the expert instance.

Figure 18: Sample form used for manual update of user profile

If some external computer agents on the Web need to process their queries directly on the experts' annotated pages produced from the suggested framework, they can only do so when they can use the SQU ontology directly or translate it. Normally, external users use the SQU portal to retrieve experts' information.

3.4.6 Annotation of Arabic Documents

Arabic documents are not processed automatically; they will manually be annotated and included in the implementation. The bi-lingual text in the RDF repository will enable users to retrieve Arabic information in reply to a SPARQL query with English text; for example, the address of an expert is returned in Arabic because the address mention was found in an Arabic document which was annotated by OntoMat with English tags conforming to the ontology used. Appendix F shows a sample of Arabic document annotated using the OntoMat annotatoter.

The other advantage of the suggested work is that, the user can search for an expert whose name is represented by any representation in the RDF repository and still get all the publications associated with the expert. This is true no matter how the author's name was written in the publication. This feature is very useful not only for Arabic names because researchers are asked to use different formats for their names depending on the requirement of the publishers. Our suggested treatment of names will be applicable for experts' names written in any language; for example, whether the author uses "Neagu, D.", "Daniel Neagu" or "D.Neagu", these names are treated equally.

3.5 Conclusions

The Semantic Web Implementation Framework (SWIF) presented can be used for websites with multi-languages. The only difference is the use of different processing resources dedicated to the language being procced. Many HLT tools are currently available. Arabic language processing tools are not used at this stage because we need to test the framework on one language first. GATE has been used because we consider it

as the best HLT tool available for semantic based implementation: it is currently being used by many research projects, it is backed by a dedicated team of researchers at Sheffield University, it has an active GATE group of users exchanging research ideas [47], and it is open source.

The choice of the Experts Finding System at SQU as a case study will benefit this research in many ways: the system can be tested on a real world data, and a research group at SQU can be formed to continue with this project and annotate Arabic documents at SQU.

Our main contribution is that we introduce the novel Semantic Web Implementation Framework (SWIF). This will open new opportunities for diverse area of Semantic Web applications; it provides for the implementation of stand-alone Semantic Web applications that can be developed, modified, tested and evaluated before enabling Internet user to use its annotated Web pages and/or its RDF repository. We designed a special Unique Uniform Resource Identifier (UURI) for the maintenance of User Profiles to allow for multiple representations of bi-lingual researchers' names. We initiate the idea of using Control Knowledge. We developed our own Clause Segmenter, Syntactic Role Identifier and Semantic Analyser. The semantic analyser makes use of CK, Thematic Role and verb class lists; it also has the unique feature of verifying the identified entities and relations. The suggested framework also provides for including bi-lingual Web pages after being manually annotated with tags that corresponds to the Ontology used.

In the chapter to follow, the details of the experiment on the case study implementation is presented.

4. Semantic Web Implementation System (SWIS)

4.1 Overview

The Semantic Web Implementation System (SWIS) is developed as an exercise to implement the framework (SWIF) suggested in this work. The case study outlined in Section 3.4 is used to demonstrate the implementation. Several programs have been written using Java, several technologies are used namely Jena, Oracle, GATE components and OWL.

In the section to follow the ontology preparation step is described. Section 4.3 presents the creation of Control Knowledge and the programs used. Section 4.4 outlines the pre-processing of Web documents by GATE components. The creation of the knowledge base models is presented in Section 4.5. The processes of managing and updating the RDF repository and Control knowledge together with the generation of the annotated documents are illustrated in Section 4.6. The algorithm used by the semantic analyser which is part of the SwissProcesses is illustrated in Section 4.7. In Section 4.8 sample queries are presented. The processing of the sample document is shown in details as a case study in Section 4.9. The evaluation and conclusion of the exercise is presented in the last section.

The source code of all the programs described in this chapter can be found on the compact disc labelled as SWIS CD that accompanies this thesis. A brief description of these programs is given in Appendix B.

4.2 Ontology Preparation

Although there are many available ontologies that could have been used for free and that are suitable for the university domain, we choose to develop our own ontology because the new ontology will meet the current work requirements.

4.2.1 Editing the Ontology File

In Section 2.5.1, we outlined several methods that are used for developing ontologies; we choose the simplest method to avoid the complexity of ontology editors especially that the knowledge base in this implementation is separated from the ontology. The simple Notepad editor is used in generating the ontology file Squ-Ontology.owl found in the accompanying SWIS CD. The Squ-Ontology.owl prepares only for the first phase implementation and will be modified to include more classes and properties as required by future implementations.

4.2.2 Validation of Ontology File

The Defence Advanced Research Projects Agency (DARPA) is the research and development organization for Department of Defence (DoD) in the USA [31]. This research organization has intensive research programs towards Semantic Web languages and technology, one of their developments is the knowledge representation language DARPA Agent Markup Language (DAML). The DAML project provides many tools for semantic based development, the dumpont.java [33] is a program for validating

ontology files and displaying the class and property hierarchies present in an OWL ontology , RDF Schema or DAML+OIL.

The dumpont.java program can be used to process any ontology available on the Internet by providing its URL. For the purpose of this work, the dumpont.java program was downloaded and modified to accept Squ-Ontology.owl as an off-line input file, and executed for syntax validation and to produce the classes and properties structure as shown in Appendix C.

4.3 Creating Control Knowledge

A simple data conversion program was used to extract the required data from SQU data base and files to produce the RDF file "squrdf.rdf" of the form found in Appendix D. The tags used in the RDF file represent classes and properties that correspond to the classes and properties in the Squ-Ontology. The creation of Control Knowledge starts with one or more RDF files and the CreateCK program is used as described below and illustrated in Figure 19.
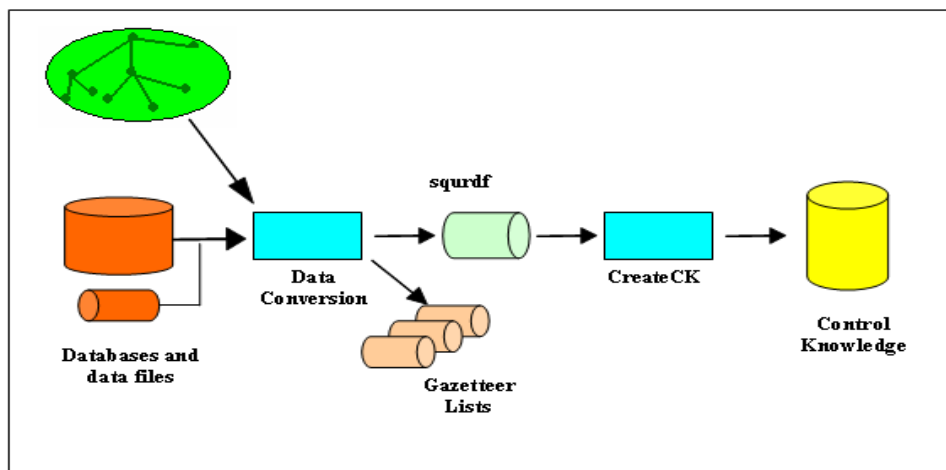
Figure 19: Creation of Control Knowledge

4.3.1 Selection of Implementation Framework and Persistence Storage

The java framework for building Semantic Web applications Jena is used as the implementation framework; Jena uses a back-end data base engine that is used by the Jena Model class. The Data Base Management System (DBMS) Oracle is one of several supported DBMS that can be used with Jena. We choose to use Oracle implementation because SQU's current database is on Oracle and having the semantic implementation on Oracle will ease future enhancements to the system. In summery Jena supported by Oracle persistent storage is used for this implementation as follows:

- Jena in its latest version Jena_2.5.4 was downloaded [67] and installed under a directory named as C:\Jenaroot, the environment variables were carefully set according to Jena requirements to enable easy and accurate use of the library of Jena classes.

- Oracle 10g with the oracle developer has been installed from the Oracle distribution (10201_database_win32.zip) and (jdev1012_1.zip) respectively on a computer named munahatam with the SID as muna and port 1521. An oracle user has been created as munah with password oracle, the usage of such information appears in the listing of the programs outlined in Appendix B.

- The execution environment for Jena and Oracle has been set by using the batch file set-classpath.bat to enable easy setting when needed. In addition, we use the OWLReasoner program of Jena to create the inference model on the Oracle persistent storage; this program requires large amount of memory to run. The Oracle environment has been set by increasing the number of running cursors to avoid **Out of Memory** errors.

4.3.2 Creation of Control Knowledge as Jena Model

The program CreateCK.java is used to generate the "SquCK" that is the Jena model that acts as Control Knowledge. The program involves creating a data base connection to an Oracle data base and loading the RDF instances from the file "SquRdfCKData.rdf" to the Control Knowledge. The output of this step is a graph in persistent storage as it appears in Figure 20.

Figure 20: The Control Knowledge as Represented in the Oracle Jena Model

4.4 Pre-processing with GATE components

The input documents were pre-processed using the GATE pre-processing components described in Section 3.3.2. This is done by executing the "GateProcesses.java" program. This program is based on the sample program StandaloneAnnie.java that comes with the GATE documentation. StandaloneAnnie.java has been modified by including the execution of several processing resources that are not part of Annie but provided by GATE. Several gazetteer lists were altered and a few new lists were added.

The Orthomatcher.java processing resource had to be altered to remove a bug found in this open source program. This bug introduced an incorrect result where two texts were denoted as matching text while they were not. "The University of Aston" was matched with "The University of Bradford", we had to correct this mistake, inform GATE team and use the new version of the modified Orthomatcher.java.

The output of GateProcesses.java is an XML version of the document that contains suggested entity Annotations with detailed information about the processed text including its attributes like POS tagging, stemming, length of each token, etc. Figure 21 below illustrate this step.
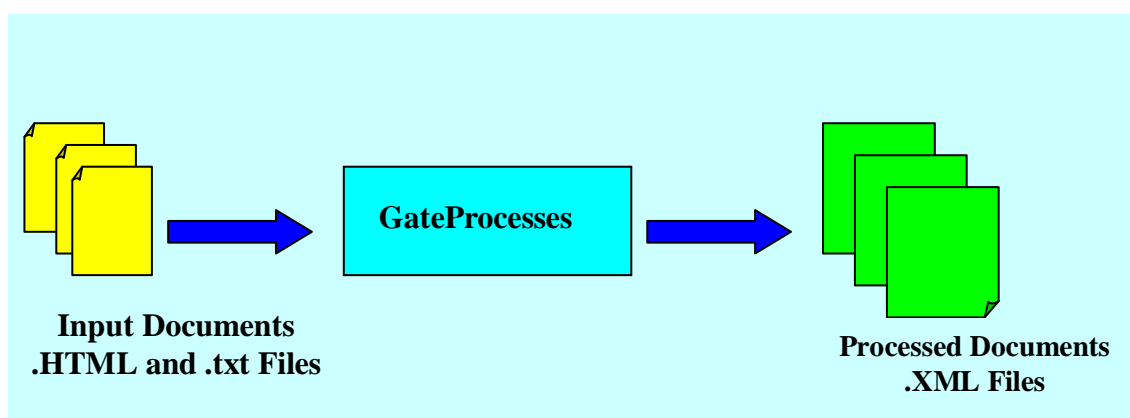
Figure 21: Pre-processing Using GATE Components

Figure 22 shows the flow of the executed processes on the input document. The XML document shows the identified entities of interest. Each processing resource used produces Annotation of some type, like Token, Sentence…etc, or add feature to previous Annotations or produces both Annotation and features.

Several gazetteer lists were created the gazetteer definition file **list.def** has been modified to provides for the newly added gazetteer lists. Three JAPE rules have been created namely **Degree.jape, Topic.jape and Subject.jape**. The list of JAPE rules in **main.jape** was also modified to prepare for the execution of this program. Appendix B describes the GateProcessing directory on the accompanying CD. It includes a copy of the source program, the sample document muna.txt and the output files produced by executing the GateProcesses.java. It also contains a copy of the rules developed. A sample of the newly added JAPE rules and Gazetteer lists can also be found in Appendix J. A sample of a document and its selected annotations are presented in Appendix K.
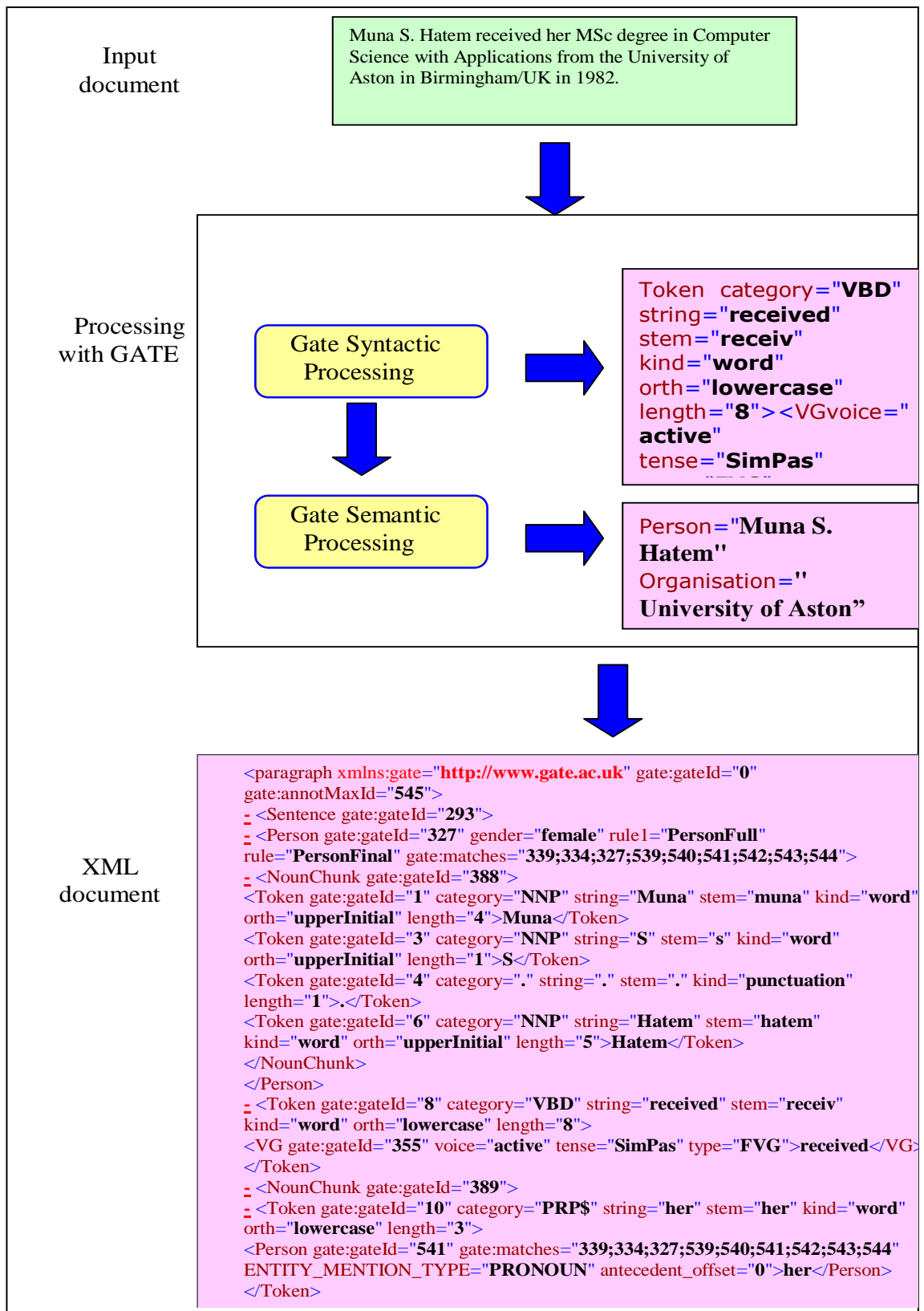
Figure 22: Sample Document processing Using GATE' Components

4.5 Creation of Knowledge Based Models and Database

This step involves creating two models and the database table used as the verifiability table. The Ontology model is created as a default Jena model that exist on the Oracle database. The program OntologyM.java reads the ontology file Squ-Qntology.owl into the default model that is given the name "SquOnt". Figure 23 shows a screen shot of the content of the created ontology model.

Figure 23: The Ontology as the Jena Model SquOnt

The RDF Repository model "SquRDF" is created also as an Oracle backed Jena Model. CreateSquRDF.java program creates the initial RDF triple of Sultan Qaboos University, it only includes the name and address of the university. Figure 24 shows screen shot of the content of the created ontology model.

Figure 24: Screen shot of the initial SquRDF model

The Oracle database table Verifiability_Tab is created using the following simple SQL code from the SQL prompt.

```
CREATE TABLE Verifiability_Tab
        ( subj VARCHAR2(250)      NOT NULL,
          prop VARCHAR2(250)      NOT NULL,
          obj VARCHAR2(250)       NOT NULL,
          VCount        NUMBER,
          PRIMARY KEY (subj,prop, obj)
                        )
```

4.6 Knowledge Base Management and Annotation

Figure 25 below illustrates this step. The processing program "SwisProcesses.java" consists of several programs and modules that play the main role in the system, its role includes:

- Verification of the entity Annotations produced by the pre-processing phase.

- Relation extraction and verification.

- Population and update of the RDF repository (RDF).

- Update the control knowledge (CK).

- Update the verifiability table Verifiability_Tab.

- Generation of the annotated version of the input document.

In Appendix B, several modules or programs of the SwisProcesses are described as the full code of these programs are on the SWIS-CD.
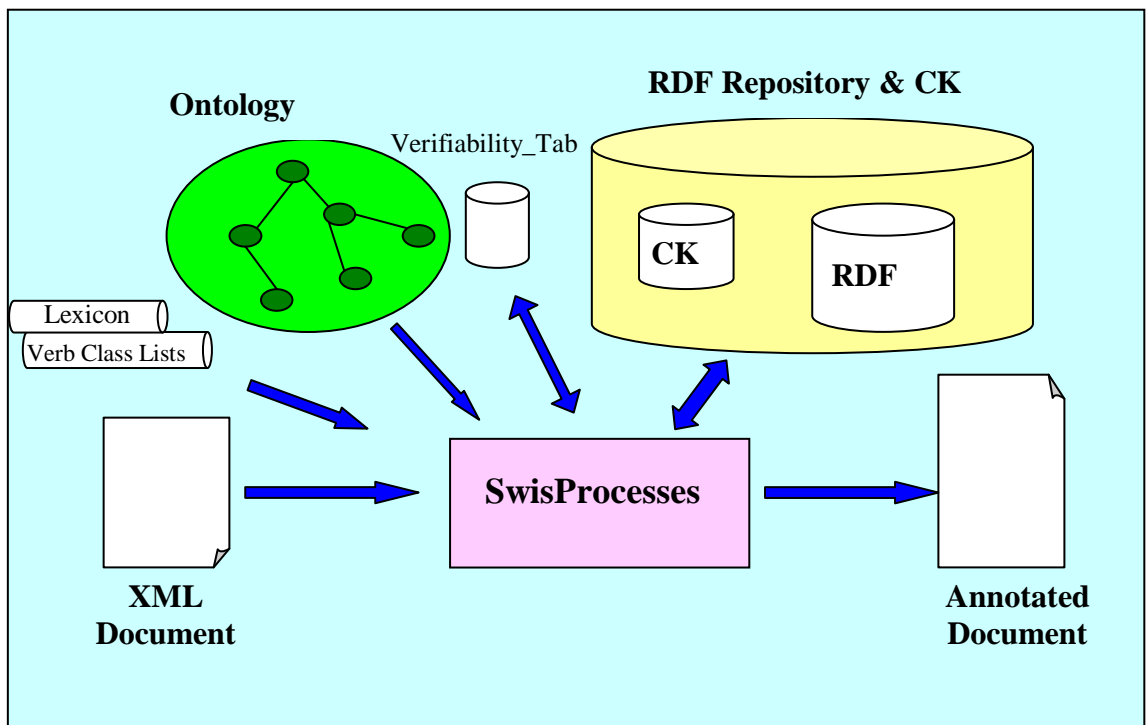
Figure 25 : Annotation and Knowledge Base Management

4.7 The Algorithm used by the Semantic Analyser

The algorithm used by the processing program SwisProcesses.java can be described as follows:

```
Input:
        o  Pre-processed XML document produced from the
           pre-processing phase
        o  Verb Class Lists
        o  The lexicon that includes the verbs syntactic
           and  thematic structure
        o  Ontology module
        o  The RDF repository (RDF)
        o  The control knowledge(CK)
Output:
        o  Updated RDF repository
        o  Updated  control knowledge
        o  Annotated version of input document.


Process:


For each sentence in the input document,
Do      use the ClauseChunker.java to produce the
        corresponding simple sentence or sentences


For each  simple sentence
Do
   ▪ Identify the entities of interest and the verb
     groups
   ▪ Assign Type to each entity    // the Type corresponds to the
                                   // Class in the Ontology
   ▪ Assign mention identifier to each entity identified
     of certain Type                    // the GATE ID is used as
                                        // mention identifier
   ▪ Examine the lexicon to identify the verb syntactic
     structure and the verb Thematic structure
   ▪ Identify the UURI of the entities and process
     coreferenced entities by assigning one UURI to the
     coreferenced group of entities
```

- Generate the meta-data for the extracted entities and execute the validation process

// this completes entities identification and Annotation

// now, we work on identifying relationships between entities

//

- Identify the Thematic Role of each entity with respect to the associated verb
- Identify the class list associated with the verb using the Verb Class List   *// Disambiguate the verb with*

// respect to the ontology

- Generate the triple for the relationships between the identified entities and execute the validation process.

*// *****   The validation process   *******

// it includes update of CK repository, RDF repository

// and the Annotation generation

For each entity or relationship found

Do

if   a similar instance exists in the CK
then   add the triple to the Annotation in output
       document;
       Add the instance to RDF repository if it is not
       already There   *// because we do not want to sore the same*
                    // information more than once

else **if** it exist on the RDF repository
      then   add 1 to the verifiability_count of the
             validation table;
             If   the verifiability _counter > N
             then copy the triple to the CK
                  and remove its corresponding entry from
                  the Verifiability_Tab
             endif
             Add the triple Annotations in the output
             document;
          else
             Add the triple to the Annotation in output
             document;
             Add the instance to RDF repository ;
                                    // artially verified
             Create an entry for the triple in the
             Verifiability_Tab with the validation count
             set to 1
      endif
endif

4.8 Querying the SQU's Semantic Based Implementation

4.8.1 Querying the RDF repository

The final phase of SWIS implementation discussed in Section 3.3.4 includes using SPARQL [107] code to semantically interrogate the repository via the command screen or from within a Java program. In Section 4.9 we use several scenarios for sample queries. The two programs SwisQuery1.java and SwisQuery2.java are used to query the RDF repository. See Appendix B for full lists of programs on the SWIS-CD.

4.8.2 Querying the Annotated Pages

While SQU relies on the RDF repository to get answers to the different queries shown above, external users have more than one choice:

- They can directly access the annotated pages via Agents that operate on these pages extract annotated information depending on the user requirements.

- They can access the RDF repository via SQU portal to get answers to their queries

See Appendix E2 for sample annotated document.

4.9 Case Study: Walk through example

The pre-processed XML document, such as the one in Figure 22, produced by GateProcesses.java is processed as shown in the following steps:

Step1.   The program CreateSquRDF.java is used to create the initial RDF triples that include only the name of the university and the country as in the program output that follows.

```
/********************************
//Display the content of SquRdf as  RDF XML output
***********************************/
<rdf:RDF
    xmlns="http://munahatam/Squ-Ontology#"
    xmlns:dct="http://purl.org/dc/terms/"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:dc="http://purl.org/dc/elements/1.1/" >
  <rdf:Description
rdf:about="http://munahatam/Org/Sultan_Qaboos_University">
 <rdf:type rdf:resource=
 "http://munahatam/Squ-Ontology#University"/>
<org_name>SultanQaboos niversity</org_name>
    <country>Oman</country>
  </rdf:Description>
</rdf:RDF>
/*******************************
//Display the content of SquRdf as  RDF Triples
**********************************/
http://munahatam/Org/Sultan_Qaboos_University
    http://www.w3.org/1999/02/22-rdf-syntax-ns#type
        http://munahatam/Squ-Ontology#University.

http://munahatam/Org/Sultan_Qaboos_University
    http://munahatam/Squ-Ontology#org_name
        "Sultan Qaboos University ".

http://munahatam/Org/Sultan_Qaboos_University
    http://munahatam/Squ-Ontology#country
        "Oman ".
```

Step2. Clause segmentation, this step produces simple sentences with one main verb and possibly infinitive. The SwisProcesses.java calls the ClauseChunker.java to construct the data structure that enables clause segmentation. See Appendix B for the description of the program ClauseChunker.java.

Step3. Examine each sentence in terms of its extracted entities and verb groups. For example the simple sentence (1) is considered in terms of its identified entities and verb groups as in (2) .

K.Day received PhD degree in computer science from the University of Minnesota in 1992
………………………………………………………………………….(1)
***Person** received**(VG)** **Degree** in **Subjec**t from the **University** in **Date** …………… (2)*

Step4. Assign the **gateId** (GI) associated with each entity as mention identifiers for each entity mentioned in the text as in the example below.

Dr Khaled Day received an undergraduate degree in computer science from the University of Tunis in 1986 and the M.Sc. and Ph.D. degrees from the University of Minnesota (USA) in 1989 and 1992 respectively. Dr. Day is currently Professor and Head of the Department of Computer Science at Sultan Qaboos University in Oman. His areas of interest include interconnection networks, parallel algorithms, distributed systems, grid computing, and wireless networks. He is a senior member of the IEEE …………...(3)

***Person1-GI** received an **Degree1-GI** in **Subject1-GI** from the **University1-GI** in **Date1-GI** and the **Degree2-GI** and **Degree3-GI** from the **Unversity2-GI** (**Country1-GI**) in **Date2-GI** and **Date3-GI** respectively. **Person2-GI** is currently **JobTitle1-GI** and **JobTiltle2-GI** of the **Department1-GI** at **University3-GI** in **Country2-GI**. **Person3-GI** is areas of **interest** include **Topic1-GI**, **Topic3-GI**, **Topic3-GI**, **Topic4-GI**, and **Topic5-GI**. **Person4-GI** is a **membership1-GI** of the **Organisation2-GI**……………………………………………………(4)*

Step5. Identify the verb syntactic structure of the sentence by accessing the verb lexicon that contains the arguments structure of each verb to identify the number of

arguments the verb takes together with the thematic structure. Example of such cases for the verb **give:**

John gave Mary the book. ……………………………………………………*(5)*
Person1 gave(VG) person2 the book(NP) …………………………………………*(6)*

The lexicon entries are:

Give: [1 <NP, Agent> , 2 <NP, Beneficiary> , 3 <NP, Theme>] ………………*entry 1*
  [1 <NP, Agent> , 2 <NP, Theme> , 3 <pp>, 4 <NP, Beneficiary>]………entry 2

The lexicon shows entry 1 matches the sentence in 5 and 6. It also shows that the verb **give** requires three arguments Subject (John), Object (the book) and indirect object (Mary).

Step6.   Examine the Thematic Relation of the arguments to identify the arguments' thematic role. The thematic role for a verb is listed as an entry in the lexicon we use because they are unpredicted.

Example

K.Day received PhD degree from the University of Minnesota in 1992 ……………………… (7)
Person *received(VG)* **Degree** *from the* **University** *in* **Date**   ………………………… (8)

From the lexicon we can see that the verb receive requires a subject of an **Agent** role, for the sentence is active, the object Degree is the **Theme**

Receive:
 [1 <NP, Agent> , 2 <NP, Theme>, 3 <pp>, 4<NP, Source>, 4<Comp>] ……..entry 3

Step7.   Disambiguate the verb with respect to the ontology; this is done by accessing the verb class lists to identify the corresponding class list, the verb is then related to a property in the ontology which enables relation extraction. For example,

the verbs receive, get, obtain, collect, attain, accept, and acquire are in the same class list, when this verb is followed by a degree entity the relation **has_Degree** is identified with the degree entity as Theme and the subject of the sentence as Agent.

Step8. Examine the coreferenced entities using the match matrix produced by GATE, and identify the UURI of the entities being identified by the longest string in the text, then substitute these entities with their UURIs. Appendix L shows sample intermediate output produced by the Semantic Analyser.

Step9. This step includes the validation process and update of CK, SquRDF and the Verifiability table. The triples extracted from munah.html are processed by the **SwisProcesses.jave** namely the module SwisProc2.java and SwisProc3.java according to the algorithm described in 4.7. The sample document munah.html has been processed, the RDF Annotation on the annotated document produced can be added under the head tag of the .html document as seen in the annotated document munah.html on SWIS CD, or Annotation can be added within the document body with tags preceding the identified entity and succeeding it, such Annotations can be seen as the Annotation instances of the sample document in Appendix E, both cases can be used, this is an implementation choice.

Step10. After running SWIS processes on several input documents, we can display the contents of SquRDF as RDF XML output and as as RDF Triples (Subject, Predicate, Object) as shown in Appendix H.

Step11. Run the SwisProc4.java to create the inferenced model of the RDF repository,

Step12. Query the inferenced model using sample queries.

Query 1:Building the profile of a person

The SquRDF repository together with the Verifiability_Tab can be used to produce the profile required stating which data items have not been validated. For example one can issue the following simple query:

> *Can I have the profile of Muna Hatem …………………………...Query 1*

The semantic Search Engine is expected to convert this text to the following SPARQL query as in SwisQuery2.java program that uses queries as in the following sample query:

> **SELECT ?x   WHERE {?x <http://munahatam/Squ-Ontology.owl**
> **#full_name> " Muna Hatem "}**

The results returned by such a query can then be displayed to the user in the appropriate format; the above query returned the following raw output that is extracted from the RDF repository in its inferenced Jena model.

```
 (http://munahatam/person/5927          http://munahatam/Squ-
Ontology.owl.owl#last_name         'Aljepoori')

 (http://munahatam/person/5927 rdf:type http://munahatam/Squ-
Ontology.owl.owl#Lecturer)

 (http://munahatam/person/5927 rdf:type owl:Thing)

 (http://munahatam/person/5927 rdf:type http://munahatam/Squ-
Ontology.owl.owl#Academic)

 (http://munahatam/person/5927 rdf:type rdfs:Resource)

 (http://munahatam/person/5927 rdf:type http://munahatam/Squ-
Ontology.owl.owl#Person)

 (http://munahatam/person/5927
rdf:type 3a81f305:11b9cb67cd0:-7fe4)

 (http://munahatam/person/5927
rdf:type 3a81f305:11b9cb67cd0:-7fe3)

 (http://munahatam/person/5927
rdf:type 3a81f305:11b9cb67cd0:-7fea)

 (http://munahatam/person/5927
rdf:type 3a81f305:11b9cb67cd0:-7fe8)

 (http://munahatam/person/5927owl:sameAs
http://munahatam/person/5927)
```

The objects produced in the last few lines are the Jena internal identifiers for some nodes called BNodes. The information displayed contains some information that has not been fully verified. Simple SQL commands from within the java program are used to retrieve the yet to be verified knowledge related to the person required.

The java program SwisProc3.java returns the triples related to the particular resource; the information contains the verifiability counts that indicate the number of times the information occurred on the processed documents. Below is a sample output of SwisProc3.java

```
SUBJ                                                    PROP
     OBJ            COUNT
http://munahatam/Person/5927    http://munahatam/Squ-
Ontology.owl#last_Name  Aljaboori    5

http://munahatam/person/5927    http://www.w3.org/1999/02/22-
rdf-syntax-ns#type   http://munahatam/Squ-Ontology#Assistant
Proffisor 2

http://munahatam/person/5927  http://munahatam/Squ-
Ontology.owl#last_name    Hatim  1
```

Query 2: Semantic Search

The following is an example of query used in SwisQuery2.java that uses semantic request to retrieve the names and addresses of all Academics whose data are included in the implemented system; the results includes all academics regardless of any explicitly included Academic property.

The query in English may be:

> *Get me the names and address of all Academics working at SQU ……Query 2*

The query in SPARQL can be:

```
SELECT ?name ?address

 WHERE {?person   ns:type squ:Academic" +

    " ." + " ?person    squ:Address  ?address" +

    " ." + " ?person     squ:last_name ?name" }

USING squ FOR  <http://munahatam/Squ-Ontology#>

USING ns    FOR  <http://www.w3.org/1999/02/22-rdf-syntax-
ns#>
```

4.10 **Evaluation and Conclusions**

There are many available standard data sets that are manually annotated and made available to be used as benchmark dataset for evaluating Information Extraction system; the Job Posting dataset [25], the Reuters Corporate Acquisitions dataset [45], and the Seminar Announcements dataset [44] are the most well known datasets that can be used mainly for evaluating the Entity Extraction process. Relation Extraction is still active research topic [65] and there has been no annotated corpus that is available to be used as a benchmark for relation Annotation.

As we are dealing with a domain oriented system, the context of the Web pages is important and the contents itself play the major role in the validation process, there is no way that we can use any of the available dataset to evaluate the work presented here. Hence, it is obvious that the most challenging task in this work is the evaluation process. Our framework assumes the existence of CK to validate the entities extracted by the pre-processing phase and the relations extracted during semantic analysis phase, but there is no such CK benchmark that can be used for evaluation.

To evaluate our system, our only choice is to use the simplest method available, and that is to compare our results with some manually annotated documents selected from the domain of the case study. Even this method has its own shortcomings in that the output might unintentionally produced to fit the expected results because the documents are being annotated by the same person.

In brief, full evaluation can only be done once a considerable size corpus of documents is manually annotated to be used as the modular expected output. This part is expected to be part of the future work.

In this chapter we make several important contributions that can be summarised as follows:

- We design, develop and implement the Semantic Web Implementation System (SWIS) as an implementation of the framework suggested in Chapter 3. We produced several programs using Java.

- Examine several Semantic Web technologies and use Jena, GATE components and OWL language; we prove that such technologies are adequate for Semantic Web implementation.

- Identify and fix a shortcoming in GATE; we discovered that one of the GATE's modules is producing inaccurate results; we find a way for fixing this problem and informed GATE developer about it. Appendix I shows the details of the messages sent to GATE support team and their reply.

- Create an algorithm for the semantic analyser.

- Design a system for creating CK using Jena backed by Oracle database.

- Demonstrate the use of the application using sample queries.

- Show the detailed steps of the implementation using walk through example.

The source code of all the programs described in this chapter can be found on the compact disc labelled as SWIS CD that accompany this thesis, brief descriptions of these programs are given in Appendix B.

5. Conclusions and Future work

5.1.1 Introduction

In general, IE systems are the backbone of any successful implementation of Semantic Web. Current IE systems often combine a variety of components to ensure "better" performance. We investigated the results published by the MUC, the ACE and the Pascal challenge and we concluded that general and non-domain dependent implementation based on current IE tools can not guarantee the required performance needed for successful implementation of Semantic Web.

5.1.2 Conclusions

We developed a framework for Semantic Web implementation that uses Control Knowledge extracted from the domain being processed to validate the information extracted by the HLT used for pre-processing. Information Extraction is enhanced by the Context of the documents and the use of the linguistic material; the Verb Class Lists and the linguistic concept of Thematic Role. For each verb in the lexicon, the arguments structure and the thematic structure of the arguments are stated. This enables accurate extraction of entity and relationship instances. We demonstrated our method in generating validated Annotations and validated RDF repository.

Our system provides for the inclusion of bi-lingual documents with the exception that Arabic documents are currently annotated manually using the Ontomat-annotizer. The RDF instances from these pages can be extracted and added to the RDF repository.

At this stage, full evaluation of the framework is not feasible because there are no benchmark systems that can be used to compare our system with. In fact, there is no information of any system that uses control knowledge to validate the extracted information. As far as we know our system is the first to attempt this strategy.

To summarise our contributions:

1. We developed a framework competitive with the current state of art Annotation tools and knowledge management systems because it handles input documents in the context in which they are created in addition to the automatic learning and verification of knowledge using only the available computerized corporate databases. The framework has been developed and implemented on the case study presented in our publications [53, 56, 57].

2. We introduce the concept of **Control Knowledge (CK**) that represents the application's domain memory and use it to verify the extracted knowledge [53].

3. We introduce the concept of **Verifiability** in the context of Annotation by comparing the meaning of the extracted text with the information in the CK and the use of the proposed database table **Verifiability_Tab**. Our approach employs a fully unsupervised algorithm that relies on CK and Verifiability status to learn new knowledge and add it to the CK [53].

4. We introduce the use of the linguistic concept **Thematic Role** in investigating and identifying the correct meaning of words in text documents, this helps in correct relation and entity extraction.

5. We introduce a new method to chunk conjoined statements and identify the missing subject of the clauses produced.

6. We use special linguistic lists to identify the **semantic class of verbs** and to relate a list of verbs to a single property in the ontology, this helps in disambiguating the verb in the input text to enable better information extraction and Annotation.

7. We use the term **"Intelligent Document"** defined in [104] to denote the semantically annotated document and introduce the following **new definition**:

   "The Intelligent Document is the document that clearly expresses its syntax and semantics for human use and software automation".

8. We identified the problems in bi-lingual and multiple representations of pesonal names. The Unique Uniform Resource Identifier (UURI) we suggested in [54, 55] provides each user with the ability to update their information on the RDF repository. Our suggested method in handling bi-lingual name is expected to be used in diverse area of application that ranges from legal access to government resources like NHS, banking, security, education, to immigration

5.1.3 Future Work

The first future direction to this work is investigating the specialized processing resources that are currently available for Arabic Language processing and determine what is needed to fully automate the processing of Arabic documents to provide for easier implementation of our system on Arabic and bi-lingual websites. We need to enhance the functionality of the SWIS by the use of adaptive website facilities, modifying the clause segmentor to include processing of embedded sentences and creating suitable graphical user interface to enable easier use of the system. We also need to motivate the use of a uniform Degree Coding System; educational institutions

should be encouraged to include their degree code on every certificate they issue to make it easier to track and use in computer applications. The code uniquely identifies the degree earned by each person. The system is to assign a specific code to each degree using the code that includes Country, University, Year, Faculty, Department, Degree-level, and Program. The Semantic Search Engine also needs to be developed; this enables converting queries entered as normal text to SPARQL queries that can directly run to interrogate the RDF repository.

The second direction for future work is to work on the implementation of E-learning application that consider using features of the Learning Management System (LMS) Moodle; Moodle is an open source software that is used at SQU, so is it feasible to add, alter or change any component in Moodle to provide for both Semantic Web implementation and LMS. This direction involves investigating Human Computer Interaction (HCI) and the development of Adaptive Web Sites. Our publication [57] received obvious interest during the presentation of the paper in the EBEL05 conference in Jordan and we received a number of contacts from different researchers about this publication. In [58] we suggest building new e-learning systems based on Semantic Web applications, we stated that:

> *"Current research in the Semantic Web area should eventually enable Web users to have an intelligent access to Web services and resources. The e-Learning will particularly benefit from the Semantic Web"*

We justify our claim that e-learning systems will particularly benefit from the Semantic Web for two reasons. First, Semantic Web provides for creating adaptive websites based on the user needs and abilities; knowledge about such needs and abilities can not only be acquired from user's previous interaction with the Internet but also related to the semantically represented knowledge of that person or class of individuals. Second,

knowledge on the Semantic Web can be verified which provides new horizon for adaptive generation of websites.

Future work also includes investigating methods for formal analytical analysis and evaluation of the proposed IE based on CK; such theoretical methods can be used to derive the performance of such systems.

Several faculty members at the Department of Computer Science in Sultan Qaboos University have shown interest in forming a new research group that is based on this research work to investigate developing several Semantic knowledge-based systems that handle documents in Arabic. The new research group is expected to have a network of researchers in the Middle East and will start with one of the most needed applications that is processing of **Legal** documents; such system that serves the Ministries of Justice in the Arabian Gulf region, lawyers and policy makers is not possible with the current limited technologies that deals with the semantics of Arabic languages. This research is a step in the right direction towards the use of Semantic Web technologies in such diverse areas of applications.

References

[1].   Aarts, B.(1997) "English Syntax and Argumentation", Macmillan Press Limited, UK, 1997

[2].   ACE (2004) "Annotation Guidelines for Entity Detection and Tracking (EDT)", February. Available from: http://www.ldc.upenn.edu/Projects/ACE/, accessed on 14 Mar. 2008

[3].   Advanced Knowledge Technology, "3Stores". Available from: http://www.aktors.org/technologies/3store/, accessed on 20 Jun. 2008

[4].   Advanced Knowledge Technology, Avalilabe from: http://www.aktors.org/akt/objectives/ accessed on 14 Mar. 2008

[5].   AktivDoc, http://nlp.shef.ac.uk/wig/aktivedoc.htm, accessed on 20 Jun. 2008

[6].   Alani, H., Kim, S., Millard, D. E., Weal, M. J., Hall, W., Lewis P. H. and Shadbolt, N. (2003) "Automatic Ontology-Based Knowledge Extraction from Web Documents", IEEE Intelligent Systems. January/February, 18(1): pp 14-21.

[7].   Alesso, P.(2004) "Preparing for Semantic Web Services", Sitepoint artical , May,2004. Available from: http://www.sitepoint.com/article/semantic- Web-services, accessed on 20 Jun. 2008

[8].   Anderson, T., Whitelock, D.(2004) "The Educational Semantic Web: Visioning and Practicing the Future of Education", Journal of Interactive Media in Education(JIME), ISSN: 1365-893X, 7(1): pp 1-15

[9].   Appelt, D., Israel, D. (1999) "An introduction to information extraction Technology", International Conference on Artificial Intelligence (IJCAI-99) , Sweden 1999 Avalilabe from : http://www.ai.sri.com/~appelt/ie-tutorial/ , accessed on 14 Mar. 2008.

[10].   Baker, Collin F.,Fillmore, Charles J. , Lowe, John B.(1998) "The berkeley framenet project". Proceedings of the 17th international conference on Computational linguistics - Volume 1: pp 86-90, Montreal, Canada, 1998

[11].   Bartlett, W. (2005) "Comparison of Techniques for Exposing Legacy Data to Semantic Web Technologies", 21st Annual Computer Science Conference, 2005, Rensselaer at Hartford, USA. Available from: http://www.rh.edu/~rhb/cs_seminar_2005/SessionD2/bartlett.pdf

[12].   Basis Technology, http://www.basistech.com/about/, accessed on 29 Nov. 2008

[13].   Bassiliades, N.(2005) "Semantic Web: Vision and Technologies", invited paper at 2nd Int. Scientific Conf. on Computer Science, IEEE Computer Society, Bulgarian Section, Chalkidiki, Greece, 30th Sep-2nd Oct 2005.

[14].   Bechhofer, S., Harmelen, F., Hendler,J., Horrocks, I., McGuinness D.L., Patel-Schneider, P.F, Stein, L.A. "OWL Web Ontology Language", Poole: W3C. Available from: http://www.w3.org/TR/2004/REC-owl-ref-20040210/, accessed 26 Feb.2008

[15].   Berk, L.M (1999) "English Syntax From Word to Discourse", New York Oxford, Oxford University Press 1999.

[16].   Berners- Lee,T., Hendler, J., Lassila, O. (2001) "The Semantic Web", Scientific American, 284(5): pp 34-43. Available From http://www-personal.si.umich.edu/~rfrost/courses/SI110/readings/In_Out_and_Beyond/Semantic_Web.pdf

[17].   Berners-Lee, T. (1998) "Semantic Web Road Map", Pool: W3C. Available from: http://www.w3.org/DesignIssues/Semantic.html, accessed on 14 Mar. 2008

[18].   Black W.J., McNaught J., Vasilakopoulos A., Zervanou K., Theodoulidis B., Rinaldi F. (2005) "CAFETIERE Conceptual Annotations for Facts, Events, Terms, Individual Entities, and RElations", Parmenides Technical Report TR-U4.3.1, 11 Jan, 2005. Available From: http://www.nactem.ac.uk/files/phatfile/cafetiere-report.pdf , accessed on 1 Jan. 2009

[19].   Brewster, C., Ciravegna, F. and Wilks, Y (2001), "Knowledge Acquisition for Knowledge Management", Position Paper, Proceeding of the IJCAI-2001 Workshop on Ontology Learning held in conjunction with the 17th International Conference on Artificial Intelligence (IJCAI-01), Seattle, August, 2001

[20].   Brewster, C., Ciravegna, F., Wilks, Y. (2002) "User Centred Ontology Learning for Knowledge Management", Proceedings of the 7th International Workshop on Applications of Natural Language to Information Systems, Stockholm, June 27-28, 2002, published by Springer Berlin / Heidelberg, Natural Language Processing and Information Systems, Lecture Notes in Computer Sciences, ISBN 978-3-540-00307-6 , 2553: pp 203-207, 2002

[21].   Brill, E. "A simple rule-based part-of-speech tagger". In Proceedings of ANLP-92, 3rd Conference on Applied Natural Language Processing, pp 152-155, Trento, IT, 1992.

[22].   Broekstra, J. "Sesame RQL: a Tutorial", http://www.openrdf.org/doc/rql-tutorial.html, accessed on 20 Jun. 2008

[23].   Broekstra, J., Kampman, A., van Harmelen, F. (2002) "Sesame: A generic architecture for storing and queryingrdf and rdf schema", Proceedings of the First International Semantic Web Conference (Horrocks, I. & Hendler, J. A. Eds.), volume 2342 of Lecture Notes in Computer Science. Springer. Available From: http://www.cs.vu.nl/~frankh/postscript/ISWC02.pdf

[24].   Bryson, J., Martin, D., Macllraith, S., Stein, L.(2002) "Toward Behavioral Intelligence in the Semantic Web, IEEE computer Journal, ISSN:0018-9162, 35(11): pp 48-54, 2002.

[25].   Califf, M.E., Mooney, R.J (1999) "Relational Learning of Pattern-Match Rules for Information Extraction", Proceedings of the Sixteenth National Conference on Artificial Intelligence and the eleventh Innovative applications of artificial intelligence conference innovative applications of artificial intelligence (AAAI-99), Orlando, FL, ISBN:0-262-51106-1: pp 328-334, July 1999

[26].   Ciravegna, F.(2001) "(LP)2 an Adaptive Algorithm for Information Extraction from Web-related Texts", Proceedings of the IJCAI-2001 Workshop on Adaptive Text Extraction and Mining , held in conjunction with the 17th International Conference on Artificial Intelligence (IJCAI-01), Seattle, August, 2001

[27].   Ciravegna, F., Chapman, S., Dingli, A., Wilks, Y.(2004) "Learning to Harvest Information for the Semantic Web", Proceedings of the 1st European Semantic Web Symposium(ESWS-2004), Greece, 2004.

[28].   Ciravegna, F., Wilks, Y. (2003) "Designing Adaptive Information Extraction for the Semantic Web in Amilcare", in Handschuh, Siegfried and Staab, Steffen, Eds. Annotation for the Semantic Web. IOS Press, Amsterdam, 2003 Available from: http://eprints.aktors.org/314/

[29].   Ciravegna,F., Dingli, A., Petrelli, D., Wilks,Y.(2002) "User-System Cooperation in Document Annotation based on Information Extraction", Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management, Ontologies and the Semantic Web EKAW02, ISBN:3-540-44268-5 , 2473: pp 122-137,2002. Available from: http://eprints.aktors.org/123/01/ekaw2002.pdf

[30].   Cunningham, H., (2000),"Software Architecture for Language Engineering", PhD thesis, University of Sheffield , 2000

[31].   DAML, "dumppont". Available from: http://www.daml.org/2003/09/dumpont, accessed on 20 Jun. 2008

[32].   DARPA, http://www.darpa.mil/, accessed on 20 Jun. 2008

[33].   Davies, J., Fensel, D., Harmelen,F.(2002) " Towards The Semantic Web: Ontology-Driven Knowledge Management", Wiley, ISBN:0470848677, 2002

[34].   Deitel, H.M., Deitel, P.J., Nieto, T.R.,(2002),"Internet & World Wide Web How To Program",Prentice Hall

[35].   Delphi Group, (1994) "The document is the process", White Paper, Publ: Delphi Consulting GroupInc., Available from: http://www.delphigroup.com/research/whitepapers/DocIsProcess.pdf.

[36]. DERI, "SWAN: Semantic Web Annotataor". Available from: http://www.deri.ie/projects/swan/, accessed on 20 Jun. 2008

[37]. Dietrich J, Kozlenkov A., Sxhroeder M, Wagner G.(2003) "Rule_based Agent for the Semantic Web", Science@Direct 2(4): pp 323-338, 2003

[38]. Dill S. Eiron N. Gibson D., Gruhl D., Guha, R., Jhingran A., Kanungo T., McCurley K.S.,Rajagopalan S., Tomkins A., Tomlin J.A., Zienberer J.Y. (2003) "A Case for Automated Large Scale Semantic Annotation", Journal of Web Semantics, 1(1): pp 115-132, December 2003

[39]. Ding, Y., Fensel, D., Klein, M., Omelayenko, B.(2002) "The Semantic Web yet another hip", Science Direct - Data & Knowledge Engineering, 41(2-3): pp 205-227, June 2002.

[40]. Ding. Y., Fensel, D., "Ontology library systems the key to successful ontology re-use", Proceedings of the First Semantic Semantic Web Working Symposium, SWWS-01, Stanford, USA, August 2001.

[41]. Domingue, J." Tadzebao and WebOnto : Discussing, Browsing, and Editing on the Web", In B. Gaines and M.Musen (editors), Proceedings of the 11th Knowledge Acquisition for Knowledge-Based Systems Workshop, April 18-23, Banff, Canada, 1998. Available from: http://kmi.open.ac.uk/people/domingue/banff98-paper/domingue.html

[42]. Etzioni, O., Cafarella, M., Downey, D., Kok, S., Popescu, A., Shaked, T., Soderland, S., Weld, D.S., Yates, A.., (2004) "Web-scale information extraction in KnowItAll" (preliminary results). In WWW 2004, pp 100-110, Available from: http://turing.cs.washington.edu/papers/www-paper.pdf.

[43]. Fillmore, Charles J. (1968) "The Case for Case". In Bach and Harms (Ed.): Universals in Linguistic Theory. New York: Holt, Rinehart, and Winston, pp 1-88.

[44]. Freitag, D. (1998) " Information Extraction from HTML: Application of a General Machine Learning Approach", Proceedings of the fifteenth national/tenth conference on Artificial intelligence/Innovative applications of artificial intelligence, American Association for Artificial Intelligence AAAI/IAAI 1998, pp 517-523

[45]. Freitag, D.(1998) "Toward General-Purpose Learning for Information Extraction", Proceedings of the 17th International Conference on Computational Linguistics - Montreal, Quebec, Canada , Volume 1, pp 404-408, 1998

[46]. Gate, http://gate.ac.uk/, accessed on 20 Jun. 2008

[47]. GATE-users Archives of the user mailing list GATE-users@lists.sourceforge.net. Available from http://sourceforge.net/mailarchive/forum.php?forum_name=gate-users , and http://news.gmane.org/gmane.comp.ai.gate.general , accessed on 28 Jun. 2008

[48]. Graham, S., Davis, D., Simeonov, S., Daniel, G., Brittenham, P., Nakamura, Y., Fremantle, P., Konig, D., Zentner, C. (2005)" Building Web Services with Java", Second Edition, Sam Publishing, Indiana.US

[49]. Gruber, J.S. (1965),"Studies in Lexical Relations", PhD. Thesis, MIT, Cambridge, MA.

[50]. Gruber, T.R. (1993), "A Translation Approach to Portable Ontology Specifications", Knowledge Acquisition, 5(2): pp 199-220, June 1993. Available from: http://www-ksl.stanford.edu/KSL_Abstracts/KSL-92-71.html

[51]. Handschuh, S., Staab, S . Maedche, A. (2001) "CREAM - Creating relational meta-data with a component-based, ontology-driven Annotation framework", In Proceedings of the 1st international Conference on Knowledge Capture (Victoria, British Columbia, Canada, October 22 - 23, 2001). K-CAP 01. ACM, ACM Press 2001, ISBN:1-58113-380-4, p76-83.

[52]. Handschuh, S., Staab, S.,Volz, R.I. (2003) "On Deep Annotation", ACM press, WWW2003, May 20-24, 2003, Budapest, Hungary. ACM 1-58113-680-3/03/0005.

[53]. Hatem, M., Neagu, D., Ramadan, H., (2008). "RDF Repository of Experts based on Context Oriented Automatic Annotation Framework", Second Asia International Conference on Modelling & Simulation (AMS2008). Kuala Lumpur, Malaysia 13 - 15 May 2008.

[54]. Hatem, M., Neagu, D., Ramadan, H.,(2006) "Towards personalization and a Unique Uniform Resource Identifier for Semantic Web Users with in an Academic Environment", The Journal of Instructional Technology and Distance learning, Vol. 3. No. 6. ISSN 1550-6908 ed. by Donald G. Perrin, June 2006, USA.

[55]. Hatem, M., Neagu, D., Ramadan, H.,(2006) "Towards personalization and a Unique Uniform Resource Identifier for Semantic Web Users", 9th International Conference ICL2006, Villach, Austria, September 2006. Proceedings ed. Michale E. Aure : KasseI university press: ISBN 3-89958-195-4

[56]. Hatem, M., Daniel Neagu, Ramadan, H. (2005), "Context Oriented RDF Repository for Semantic Web: Application to Sultan Qaboos University Web Services", The 6th Informatics Workshop for Research Students, University of Bradford, UK, March 2005. Proceedings ed. by Rigas,D. Bradford: University of Bradford: ISBN 1851432205, pp 67-70

[57]. Hatem, M., Ramadan, H., Neagu, D.,(2005),"e-Learning based on Context Oriented Semantic Web", Journal of Computer Science 1(4): 499-503, ISSN 1549-3636, Science Publications, 2005, New York, USA.

[58].   Hatem, M., Ramadan, H., Neagu, D. (2005) " e-Learning based on Context Oriented Semantic Web", The first International Conference on E-Business and E-Learning EBL05 , Princess Sumaya University for Technology, May 23-24 2005 Amman, Jordan. Proceedings (ed. by Chapelet. P, Awajan, A.) Princess Sumaya University for Technology: ISBN 9957-8585-0-0, pp 51-56

[59].   Hewlett-Packard Laboratories, "HP Labs Semantic Web Research", http://www.hpl.hp.com/semweb/, accessed 2nd March 2008

[60].   Horridge, M., Knublauch, H., Rector, A., Stevens, R., Wroe, C. "Practical Guide To Building OWL Ontologies Using The Protégé -OWL Plugin and CO-ODE Tools", Edition 1.0, The University Of Manchester, 2004

[61].   Horrocks, I., Patel-Schneider, P.F., "Optimizing description logic subsumption", Journal of Logic and Computation, 9(3): pp 267–293, 1999

[62].   http://www.mondeca.com/owl/moses/univ.owl   and http://www.mondeca.com/owl/univ1_1.xml accessed on 25 Dec. 2005

[63].   http://www.w3.org/TR/2003/CR-owl-guide-20030818/wine.rdf accessed on 25 Dec. 2005

[64].   Ireson, N., Ciravegna, F.(2005)" Pascal Challenge: The evaluation of machine learning for information extraction", Dagstuhl workshop on Learning for the Semantic Web, 13-18 February 2005, Dagstuhl, Germany.

[65].   Iria, J., Ciravegna, F.(2005) "Relation Extraction for Mining the Semantic Web", Proceedings Machine Learning for the Semantic Web Dagstuhl Seminar 05071, Dagstuhl. DE Available from: http://eprints.aktors.org/405/01/iria-ciravegna.pdf, accessed on 29 Nov. 2008

[66].   ISWC, 2004 , http://Annotation.Semantic Web.org/iswc/iswc.owl , accessed on 20 Jun. 2008

[67].   Jena, http://jena.sourceforge.net/, accessed on 20 Jun. 2008

[68].   Jurafsky, D., Martin, J. (2008) "Speech and Language Processing", Prentice Hall, 2008

[69].   Kiryakov, A., Popov, B., Ognyanoff, D., Manov, D., Kirilov, A., Goranov, M. (2003) "Semantic Annotation, Indexing, and Retrieval", Extended and updated version of [KiryakovEtAl2003]. Elsevier's Journal of Web Semantics, Vol. 2, Issue (1), 2005.

[70].   Kogut, P., Holmes, W.(2001) "AeroDAML: Applying information extraction to generate DAML Annotations from Web pages". Proceedings: KCAP-2001 Workshop on Knowledge Markup and Semantic Annotation.

[71].   Larry, P.(2002) "The State of the Internet: Growth and Gaps", California State University,USA Avaiable from: http://www.isoc.org/inet2000/cdproceedings/8e/8e_4.htm#s4

[72].   Leonard, T. "Advanced Knowledge Technology" Poole: University of Southampton. Available from: http://www.aktors.org/technologies/dome/, accessed 2nd March 2008

[73].   Levin, B, (1993) "English Verb Classes and Alternations A Preliminary Investigation", University of Chicago Press, Chicago.

[74].   Maedche, A., Volz, R.(2001) "The ontology extraction & maintenance framework: Text-to-onto", Proceedings of the IEEE International Conference on Data Mining, California, USA (2001)

[75].   McDowell L., Etzioni O., Halevy A. (2004) "Semantic email: theory and applications", Journal of Web Semantics, 2(2), pp153-183

[76].   McGuinness, D.L, Fikes, R. Rice, J., Wilder, S. (2000), "The Chimaera Ontology Environment", Proceedings of the Seventeenth National Conference on Artificial Intelligence AAAI, Austin, Texas. July 30 - August 3, 2000

[77].   McGuinness, D.L, Fikes, R., Hendler, J., Stein, L.A.(2002),"DAML+OIL: An Ontology Language for the Semantic Web", In IEEE Intelligent Systems, 17(5): pp 72-80, September/October 2002.

[78].   Message Understanding Conference, http://www.itl.nist.gov/iaui/894.02/related_projects/muc/proceedings , accessed on 20 Jun. 2008

[79].   Miles-Board, T., Woukeu, A., Carr, L., Wills, G. and Hall, W. (2005) "Bringing the Semantic Web to the Office Desktop", In: 2005 ACM International Symposium on Document Engineering, 2 - 4 November 2005, Bristol, United Kingdom.

[80].   Mindswap,"SMORE: Semantic Markup, Ontology and RDF Editor", http://www.mindswap.org/~aditkal/editor.shtml accessed on 1 July 2004

[81].   mindswap,"SWOOP" , http://www.mindswap.org/2004/SWOOP/~adikal/editor.shtml accessed 1-4-2006

[82].   Netcraft, http:// www.netcraft.com, accessed on 14 Mar. 2008

[83].   Ogbuji, U. (2002) "The Language of the Semantic Web", June 2002. Available from: http://www.ddj.com/architect/184414544, accessed on 1st Sep. 2005

[84].   Online Computer Library Corporation, http:// www.oclc.org, accessed on 14 Mar. 2008

[85].   Ontotext, http://www.ontotext.com/, accessed on 20 Jun. 2008

[86].   Pascal challenge latest website: http://nlp.shef.ac.uk/pascal/

[87]. Popov B., Kirayakov A., Ognyanoff D., Manov D., Kirilov A (2004) "KIM - a semantic platform for information extraction and retrieval", Natural Language Engineering 10 (3/4), p 375-392.

[88]. Porter, M.(1980) "An algorithm for suffix stripping", Program, 14(3): 130-137, 1980. Available From: http://tartarus.org/~martin/PorterStemmer/def.txt

[89]. Price, C. and Spackman, K. (2000). "SNOMED clinical terms", BJHC&IM-British Journal of Healthcare Computing & Information Management (173): 27-31.

[90]. RdfDB , http://www.guha.com/rdfdb/, accessed on 20 Jun. 2008

[91]. Riloff, E., Phillips, W. (2004) "An Introduction to Sundance and AutoSlog System", University of Utah, UUCS-04-015, 2004

[92]. Sahuguet, A., Azavant, F. "WysiWyg Web Wrapper Factory (W4F)". Available from: http://db.cis.upenn.edu/DL/WWW8/ ,accessed 26 Feb.2008

[93]. Schofield, J. (2003)"The third era starts here". The Guardian, 20th May 2003. Available from: http://lkcn.net/bbs/lofiversion/index.php/t27174.html, accessed 1 March 2008

[94]. Schroeter, R., Hunter, J., Kosovic.,D. "Vannotea - A Collaborative Video Indexing, Annotation and Discussion System for Broadband Networks" Knowledge Markup and Semantic Annotation Workshop, K-CAP 2003. Sanibel, Florida. October 2003. Available from: http://meta-data.net/filmed/pub/Vannotea.pdf

[95]. Sem WebCentral, http://projects.sem Webcentral.org/projects/aeroswarm, accessed on 20 Jun. 2008

[96]. Semantic Web, "SemanticWord", http://annotation.semanticweb.org/Members/lago/AnnotationTool.2004-01-28.5203, accessed on 29 Nov. 2008

[97]. Shadbolt, N., Ciravegna, F., Domingue, J., Hall, W., Motta, E., O'Hara, K., Robertson, D., Sleeman, D., Tate, A., Wilks, Y. "Advanced Knowledge Technologies Interdisciplinary Research Collaboration" , Mid-term Review, September 2003. Available from: http://www.aktors.org/publications/Mid-Term Scientific Review.htm#_ftnref2, accessed on 1 July. 2008

[98]. Shadbolt, N.R. , Burton, M. (1990) "Knowledge elicitation", J.R. Wilson & E.N. Corlett, Eds, Evaluation of Human Work: A Practical Ergonomics Methodology, p.321-345. London: Taylor and Francis.

[99]. Singhal, A. (2001) "Modern Information Retrieval: A Brief Overview", Bulletin of the IEEE Computer Society Technical Committee on Data Engineering 24 (4): 35-43.

[100]. Southampton Univ., http://rdf.ecs.soton.ac.uk/ontology/ecs, accessed on 20 Jun. 2008

[101]. United Nations Standard Products and Services Code (UNSPSC). Available from: http://www.unspsc.org/, accessed on 20 Jun. 2008

[102]. University of Queensland, "Vannotea", Available from: http://www.itee.uq.edu.au/~eresearch/projects/vannotea/index.html, accessed on 20 Jun. 2008

[103]. University of Washington, "UW Mangrove Project". Available from: http://www.cs.washington.edu/research/semweb/annotation.html, accessed on 20 Jun. 2008

[104]. Uren, V., Cimiano, P. Iria, J., Handschuh, S., Vargas-Vera, M., Motta, E., Ciravegna, F.(2006) "Semantic Annotation for Knowledge Management: Requirements and a Survey of the State of the Art", Journal of Web Semantics, 2006, 4(1): pp14-28.

[105]. W3C, "Annotea Project", http://www.w3.org/2001/Annotea/, accessed on 20 Jun. 2008

[106]. W3C, "RDF Primer",2004. Available from: http://www.w3.org/TR/2004/REC-rdf-primer-20040210/, accessed on 3 Mar. 2008

[107]. W3C, "SPARQL Query Language for RDF", http://www.w3.org/TR/rdf-sparql-query/ accessed on 20 Jun. 2008

[108]. W3C, "Welcome to Amaya W3C's Editor/Browser". Available from : http://www.w3.org/Amaya/, accessed on 20 Jun. 2008

[109]. W3C,Amaya,"What is an Annotation". Available from: http://www.w3.org/Amaya/User/attaching_Annotations/what_is_an_Annotatio n.html, accessed on 20 June 2008

Appendices

Appendix A: A Summary of the Annotation Tools Investigated in this Work

Annotation Tool	Developer	Automation	Automation tool	Machine Learning	Support KB Services used	Multi-Lingual
Amaya	W3C	No	No	No		Yes
Mangrove	University Of Washington	No	No	No	Who's who and Calendar	No
Vannotea	University of Queensland	No	No	No	No	No
SMORE	University of Maryland	No	No	No	No	No
Melita	University of Sheffield	Yes	Amilcare	Supervised		
OntoMat	University of Karlsruhe	Yes	Amilcare, PANKOW	Genetic algorithm and Supervised		No
Armadillo	University of Sheffield	Yes	Amilcare, and other ML techniques		Structured data	No

Table 2: A Summary of the Annotation Tools Investigated in this Work

Appendix B: The Contents of the SWIS CD

The programs, JAPE Rules, Ontology file and all other developed or modified code for the purpose of this project are provided on the SWIS CD that accompanies this thesis. The content of the CD are arranged in several folders as follows:

Ontology: This directory contains a copy of the ontology file *Squ-Ontology .owl* together with the *dumpoint-output.html* which is the output produced from executing the *dumpoint2.java.*

GateProcesses: It includes three subdirectories and a file.

1. GateProcesses : The GateProcesses.java program executes all the processing resources needed by GATE pre-processing to produce the GP_out_toXML_1.xml together with other output files. The files produced in two different versions; version 1 with detailed annotation, and version 2 with selected annotation of interest.

2. Gazetteer Lists: several gazetteer list files are included together with a copy of the modified gazetteer definition list files lists.def.

3. JAPE rules: contains the rules written for the purpose of this work. In addition, we also included developed rules that can be used instead of the CluaseChunker that is currently used as part of the SwisProcesses.

4. The batch file set-classpath1.bat contains the commands needed to set the correct environment for Java, Jena and Oracle

Creating Jena Modules: This directory includes three sub-directories that are used to build the three different Jena modules used in this work. The same program has been

used in three different versions for building each of the three modules; they are separated here into sub-directories and different program names only to clarify the input files and the produced module name.

1. Control Knowledge: it includes the CreateCK.java program that has been developed to create the initial control knowledge. The data used to build the initial control knowledge is also included as SquRdfData.rdf file.

2. Ontology module SquOnt: the OntologyM.java program which is used to converts the ontology into Jena module on persistent storage.

3. Initial RDF Repository-squrdf1: the CreateSquRDF.java program creates the initial SQU RDF triple that includes the name and address of the university.

Annotated Pages: This directory contains sample biography documents in its annotated version. It also includes an Arabic documents annotated using the Ontomat annotator.

SwisProcesses: is a directory that contains the source code of the programs developed to implement the algorithm in Section 4.7.

1. Clause Chunker : shows the basic code for the ClauseChunker.java program that can be used to convert complex sentences into simple ones. JAPE rules were written at this stage to do the same task.

2. Update Verifiability table and RDF model using the following programs and module:

   ▪ SwisProc2.java: this program manages the control knowledge, updates the RDF repository and manages validation process

▪ SwisProc3.java: this module is used to interrogate and update the verifiability table

3. Creates the inferenced model using SwisProc4.java.

4. SemanticAalyser: This directory contains the SemanticAnalyser.java program that is used for relation and entity extraction. It also prepares for annotation generation.

SwisProcesses.java is the main or "container" program that is used to execute all the programs and modules that represent Semantic Analysis, and knowledge base management and Annotation.

Queering the RDF repository: the sample program SwisQuery1.java and its modified version SwisQuery2 are used to interrogate the inferenced model. The code can be used as a template for executing many other queries.

Appendix C: Class and Property Hierarchy- Output of dumpont.java

"null" mm null "http://localhost/Squ-Ontology.owl" mm2 http://localhost/Squ-Ontology.owl http://localhost/Squ-Ontology.owl using dumpont2.java

Class Hierarchy

- Country ()
- Degree (degree_date, degree_name, hasDegreeCountry, hasDegreeFrom, hasDegreeSubject)
- HomePage ()
- JobTitle ()
- Organization (city, country, hasSubOrg, org_address, org_name)
  - University ()
    - instance Sultan_Qaboos_University
  - College ()
  - Institute ()
  - Department ()
  - ResearchGroup ()
  - SocietyOrAssociation ()
- Person (address, birth_date, email, fax, first_name, full_name, hasDegree, hasHomePage, hasPJobTitle, hasPWorkFor, hasJobTitle, hasWorkFor last_name, middle_initial, phone, title)
  - Employee (squ_id)
    - AdministrativeStaff ()
      - ClericalStaff ()
      - SystemsStaff ()
    - Academic (hasCourseTeach, hasResearchTopic, has_Membership)
      - Faculty ()
        - FullProfessor ()
        - AssociateProfessor ()
        - AssistantProfessor ()
        - Chair ()
        - Dean ()
        - VisitingProfessor ()
      - Consultant ()
      - Lecturer ()
  - ExternalCooperator (hasCooperattionWith, hasSupervisee, hasSupervisor)
- Publication (hasPubAuthor, hasPubTopic, pub_year, pup_title)
  - Thesis ()
    - PhDThesis ()
    - MasterThesis ()
  - Article ()
    - ConferencePaper ()
    - JournalArticle ()
  - Book ()
  - Manual ()
- Student ()

- - UndergraduateStudent ()
  - GraduateStudent ()
  - ResearchAssistant (WorksFor)
- TeachingAssistant ()
- Topic (minor_Topic, topic_name)
- Work ()
  - Course ()
  - Research ()

Property Hierarchy

- WorksFor
- address
- birth_date
- city
- country
- degree_date
- degree_name
- email
- fax
- first_name
- full_name
- hasCooperattionWith
- hasCourseTeach
- hasDegree
- hasDegreeCountry
- hasDegreeFrom
- hasDegreeSubject
- hasHomePage
- hasJobTitle
- hasOrgAdmin
- hasPubAuthor
- hasPubTopic
- hasResearchTopic
- hasSubOrg
- hasSupervisee
- hasSupervisor
- hasWorkFor
- has_Membership
- last_name
- middle_initial
- minor_Topic
- org_address
- org_name
- phone
- pub_year
- pup_title
- research_interest
- squ_id
- teachingAssistantOf

- title
- topic_name

Produced from http://localhost/Squ-Ontology.owl using dumpont2.java

Appendix D: Sample RDF File of Extracted Information from SQU Data Base

```
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [
   <!ENTITY squ  'http://munahatam/Squ-Ontology.owl#'>
   <!ENTITY rdf  'http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
   <!ENTITY rdfs 'http://www.w3.org/2000/01/rdf-schema#'>
   <!ENTITY xsd  'http://www.w3.org/2001/XMLSchema#'>
   <!ENTITY owl  "http://www.w3.org/2002/07/owl#" >]>
<rdf:RDF       xmlns:rdf="&rdf;"       xmlns:rdfs="&rdfs;"       xmlns:xsd="&xsd;"
xmlns:owl="&owl;"
     xmlns="&squ;">


<Lecturer rdf:about="http://munahatam/person/5927">
     <address>
         Dept. of Computer Science, Sultan Qaboos University, PO. Box 36Postal Code
123, Muscat, Oman
     </address>
     <email>munahatam@squ.edu.om</email>
     <squ_Id>5927</squ_Id>
     <fax>(968) 24143415</fax>
     <first_Name>Mona</first_Name>
     <hasHomePage         rdf:resource="http://www.squ.edu.om/sci/Comp/FINAL
SHP/muna.html"/>
     <last_Name>Hatam</last_Name>
     <middle_Initial>S</middle_Initial>
     <full_Name>Mona Salman Hatam Al-Jiboori</full_Name>
     <phone>(968) 24142223</phone>
     <birth_date>14/11/1951</birth_date>
  </Lecturer>
<Lecturer rdf:about="http://munahatam/person/5927">
     <first_Name> Muna</first_Name>
      <last_Name>Hatem </last_Name>
</Lecturer>
<Lecturer rdf:about="http://munahatam/person/5927">
     <last_Name>Hattem </last_Name>
</Lecturer>
<Lecturer rdf:about="http://munahatam/person/5927">
<first_Name>منى</first_Name>
     <last_Name>حاتم </last_Name>
  <full_Name>Muna Salman Hatem Al-Jepoori</full_Name>
   <full_Name>منى سلمان حاتم الجبوري</full_Name>
     <hasDegree rdf:resource="http://munahatam/degree/5927_d1"/>
     <hasDegree rdf:resource="http://munahatam/degree/5927_d2"/>
 </Lecturer>
<Degree rdf:about="http://munahatam/degree/5927-d1">
     <degree_name>BSc</degree_name>
     <degree_date>1975</degree_date>
     <hasDegreeSubject rdf:resource="http://munahatam/topic/Physics"/>
```

```xml
        <hasDegreeFrom
rdf:resource="http://munahatam/organisationc/Baghdad_University"/>
        <hasDegreeCountry rdf:resource="http://munahatam/country/Iraq" />

 </Degree>
<Degree rdf:about="http://munahatam/degree/5927_d2">
        <degree_name>MSc</degree_name>
        <degree_date>1982</degree_date>
        <hasDegreeSubject rdf:resource="http://munahatam/topic/Computer_Science"/>
        <hasDegreeFrom
rdf:resource="http://munahatam/organisation/University_of_Aston_in_ Birmingham"/>
        <hasDegreeCountry rdf:resource="http://munahatam/country/UK"/>
</Degree>
<AssociateProfessor rdf:about="http://munahatam/973">
        <address>Dept. of Computer Science, Sultan Qaboos University, PO.
        Box 36Postal Code 123, Muscat, Oman   </address>
        <email>Haiderr@squ.edu.om</email>
        <fax> (968) 24413415 </fax>
        <first_Name> Haider </first_Name>
        <hasHomePage           rdf:resource="http://www.squ.edu.om/sci/Comp/FINAL
SHP/haider.html"/>
        <last_Name>Al-lawati  </last_Name>
        <middle_Initial>R   </middle_Initial>
        <full_Name>Haider Ramdan</full_Name>
        <phone>(968) 24141807</phone>
        <birth_date>01/03/1961</birth_date>
        <hasDegree rdf:resource="http://munahatam/degree/973_d1"/>
        <hasDegree rdf:resource="http://munahatam/degree/973_d2"/>
        <hasDegree rdf:resource="http://munahatam/degree/973_d3"/>
</AssociateProfessor>
<Degree rdf:about="http://munahatam/degree/973_d1" >
        <degree_name>BSc</degree_name>
        <degree_date>1985</degree_date>
        <hasDegreeSubject rdf:resource="http://munahatam/topic/Computer_Science"/>
        <hasDegreeFrom rdf:resource="http://munahatam/organisationc/North_Carolina
"/>
        <hasDegreeCountry rdf:resource="http://munahatam/country/USA" />

</Degree>
  <Degree rdf:about="http://munahatam/degree/973_d2" >
        <degree_name>MSc</degree_name>
        <degree_date>1988</degree_date>
        <hasDegreeSubject rdf:resource="http://munahatam/topic/Computer_Science" />
        <hasDegreeFrom rdf:resource="http://munahatam/organisationc/NorthCarolina
"/>
        <hasDegreeCountry rdf:resource="http://munahatam/country/USA" />
</Degree>
  <Degree rdf:about="http://munahatam/degree/973_d3" >
        <degree_name>PhD</degree_name>
        <degree_date>1993</degree_date>
```

```xml
        <hasDegreeSubject
rdf:resource="http://munahatam/topic/Artificial_Intelligence"/>
        <hasDegreeFrom
rdf:resource="http://munahatam/organisationc/University_of_Sussex "/>
        <hasDegreeCountry rdf:resource="http://munahatam/country/UK" />
</Degree>
<Professor rdf:about="http://munahatam/3899">
        <address>Dept. of Computer Science, Sultan Qaboos University, PO.
        Box 36Postal Code 123, Muscat, Oman   </address>
        <email>kday@squ.edu.om</email>
        <fax> (968) 24413415 </fax>
        <first_Name> Khaled  </first_Name>
        <hasHomePage          rdf:resource="http://www.squ.edu.om/sci/Comp/FINAL
SHP/khaled.html"/>
        <last_Name>Day</last_Name>
        <middle_Initial> N </middle_Initial>
        <full_Name>Khaled Day</full_Name>
        <birth_date>01/01/1963</birth_date>
        <phone>(968) 24141482</phone>
        <hasDegree rdf:resource="http://munahatam/degree/3899_d1"/>
        <hasDegree rdf:resource="http://munahatam/degree/3899_d2"/>
        <hasDegree rdf:resource="http://munahatam/degree/3899_d3"/>
</Professor>
<Degree rdf:about="http://munahatam/degree/3899_d1" >
        <degree_name>BSc</degree_name>
        <degree_date>1986</degree_date>
        <hasDegreeSubject rdf:resource="http://munahatam/topic/Computer_Science"/>
        <hasDegreeFrom
rdf:resource="http://munahatam/organisationc/Faculté_des_Sciences_de_Tunis "/>
        <hasDegreeCountry rdf:resource="http://munahatam/country/Tunisia"/>
</Degree>
<Degree rdf:about="http://munahatam/degree/3899_d2" >
        <degree_name>MSc</degree_name>
        <degree_date>1989</degree_date>
        <hasDegreeSubject rdf:resource="http://munahatam/topic/Computer_Science"/>
        <hasDegreeFrom
rdf:resource="http://munahatam/organisationc/University_of_Minnesota "/>
        <hasDegreeCountry rdf:resource="http://munahatam/country/USA"/>
</Degree>
 <Degree rdf:about="http://munahatam/degree/3899_d3" >
        <degree_name>PhD</degree_name>
        <degree_date>1992</degree_date>
        <hasDegreeSubject
rdf:resource="http://munahatam/topic/Parallel_and_Distributed_Computing"/>
        <hasDegreeFrom
rdf:resource="http://munahatam/organisationc/University_of_Minnesota "/>
        <hasDegreeCountry rdf:resource="http://munahatam/country/US"/>
</Degree>
</rdf:RDF>
```

Appendix E: Sample Document and its Annotated Version

E1: Sample Source Document

"Muna S. Hatem received MSc degree in Computer Science with Applications from the University of Aston in1982. She worked as a Programmer, System Analyst, Project leader and head of department of the Computer centres in different governmental governmental organizations. She was a lecturer at Al-Ahliyya Amman University in the Faculty of Engineering/ the Department of Computer Engineering for 5 year. Mrs. Hatem is currently a a lecturer at Sultan Qaboos University/ Department of Computer Science in Oman. Her main research interests are Semantic Web, NLP, Internet Programming, and Database Systems"

E2: The Intelligent Version of the Document

```
<?xml version="1.0" ?>
<!DOCTYPE rdf:RDF [
<!ENTITY squ   'http://munahatam/Squ_Ontology.owl#'>
      <!ENTITY dc "http://purl.org/dc/elements/1.1/">
      <!ENTITY dct "http://purl.org/dc/terms/">
      <!ENTITY owl "http://www.w3.org/2002/07/owl#">
      <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#">
      <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#">
      <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#">]>

<rdf:RDF xmlns:dc="&dc;"
      xmlns:dct="&dct;"
      xmlns:owl="&owl;"
      xmlns:rdf="&rdf;"
      xmlns:rdfs="&rdfs;"
  xmlns="&squ;">



<Person rdf:about="http://munahatam/Person/5927 ">
<first_Name>Muna</first_Name>
<middle_Initial>S.</middle_Initial>
<last_name >Hatem</last_name>
</Person>

<Degree rdf:about="http://munahatam/Degree/5927_d2" >
<hasDegreeName rdf:resource="http://munahatam/Degree/MSc"/>
<hasDegreeSubject rdf:resource="http://munahatam/topic/Computer_Science_with_
Applications"/>
<hasDegreeFrom rdf:resource="http://munahatam/organisationc/University_of_Aston
"/>
<degree_date>1982</degree_date>
```

```
</Degree>
<Person rdf:about="http://munahatam/Person/5927 ">
<hasDegree rdf:resource="http://munahatam/Degree/5927_d2" />
</Person>

<Person rdf:about="http://munahatam/Person/5927">
<hasPJobTitle rdf:resource="http://munahatam/JobTitles/Programmer"/>
<hasPJobTitle rdf:resource="http://munahatam/JobTitles/System_Analysts"/>
<hasPJobTitle rdf:resource="http://munahatam/JobTitles/Project_Leder"/>
</Person>

<University rdf:about="http://munahatam/Org/Al_Ahliyya_Amman_University">
<org_name>Al-Ahliyya Amman University in </org_name>
<hasSubOrg  rdf:resource="http://munahatam/Org/FoE"/>
</University>

<College rdf:about="http://munahatam/Org/FoE">
 <org_name>Faculty of  Engineering/</org_name>
 <hasSubOrg  rdf:resource="http://munahatam/Org/DCE"/>
 </College>

<Department rdf:about="http://munahatam/Org/DCE">
 <org_name> Department of Computer Engineering</org_name>
</Department>

<Person rdf:about="http://munahatam/Person/5927">
<hasPJobTitle rdf:resource="http://munahatam/JobTitles/Lecturer"/>
<hasPWorkedFor
rdf:resource="http://munahatam/Org/Al_Ahliyya_Amman_University/FoE/DEC"/>
</Person>


<University rdf:about="http://munahatam/Org/Sultan_Qaboos_University">
<org_name>Sultan Qaboos University</org_name>
<hasSubOrg  rdf:resource="http://munahatam/Org/Sultan_Qaboos_University/CoS"/>
<country>Oman</country>
</University>

<College rdf:about="http://munahatam/Org/Sultan_Qaboos_University/CoS">
 <org_name>College of Science/</org_name>
 <hasSubOrg
rdf:resource="http://munahatam/Org/Sultan_Qaboos_University/CoS/DCS"/>
 </College>

<Department rdf:about="http://munahatam/Org/Sultan_Qaboos_University/CoS/DCS">
 <org_name> Department of Computer Science</org_name>
</Department>

<Person rdf:about="http://munahatam/Person/5927 ">
<hasJobTitle rdf:resource="http://munahatam/JobTitles/Lecturer"/>
```

```
<hasWorkFor
rdf:resource="http://munahatam/Org/Sultan_Qaboos_University/CoS/DCS"/>
</Person>

<Lecturer rdf:about="http://munahatam/Person/5927">
<hasResearchTopic rdf:resource="http://munahatam/Topics/Semantic_Web"/>
<hasResearchTopic rdf:resource="http://munahatam/Topics/NLP"/>
<hasResearchTopic rdf:resource="http://munahatam/Topics/Internet_Programming"/>
<hasResearchTopic rdf:resource="http://munahatam/Topics/Database_Systems"/>
</Lecturer>

</rdf:RDF>
```

Appendix F: Arabic Document and its Annotated Version

F1: Processing Sample Document in Arabic using OntoMat

F2: Screen Image of the Generated Annotation

F3: The Annotated Version of the document using OntoMat

```
<rdf:RDF
    xmlns:rss="http://purl.org/rss/1.0/"
    xmlns:squ="http://www.squ.edu.om/sw/Squ-Ontology#"
    xmlns:jms="http://jena.hpl.hp.com/2003/08/jms#"
    xmlns:ontomat="http://Annotation.semanticweb.org/ontologies/cream/ontomat#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns="http://yournamespace.org#"
    xmlns:vcard="http://www.w3.org/2001/vcard-rdf/3.0#"
    xmlns:daml="http://www.daml.org/2001/03/daml+oil#"
    xmlns:dc="http://purl.org/dc/elements/1.1/">
  <owl:Ontology rdf:about="http://yournamespace.org#">
    <owl:imports
rdf:resource="http://Annotation.semanticweb.org/ontologies/cream/ontomat#"/>
    <owl:imports rdf:resource="http://www.squ.edu.om/sw/Squ-Ontology#"/>
  </owl:Ontology>
  <ontomat:ReificationDataProperty>
    <ontomat:AboutProperty>http://www.squ.edu.om/sw/Squ-
Ontology#fax</ontomat:AboutProperty>
    <ontomat:AboutIndividual>http://www.squ.edu.om/sw/Squ-
Ontology#Kday</ontomat:AboutIndividual>
    <ontomat:CreationSource>http://yournamespace.org#xpointer(//point()[98]/range-
to(//point()[106])</ontomat:CreationSource>
    <rdfs:label>about:        http://www.squ.edu.om/sw/Squ-Ontology#Kday        -
http://www.squ.edu.om/sw/Squ-Ontology#fax</rdfs:label>
  </ontomat:ReificationDataProperty>
  <ontomat:ReificationDataProperty>
    <ontomat:AboutIndividual>http://www.squ.edu.om/sw/Squ-
Ontology#Kday</ontomat:AboutIndividual>
    <rdfs:label>about:        http://www.squ.edu.om/sw/Squ-Ontology#Kday        -
http://www.squ.edu.om/sw/Squ-Ontology#first_name</rdfs:label>
    <ontomat:CreationSource>http://yournamespace.org#xpointer(//point()[22]/range-
to(//point()[26])</ontomat:CreationSource>
    <ontomat:AboutProperty>http://www.squ.edu.om/sw/Squ-
Ontology#first_name</ontomat:AboutProperty>
  </ontomat:ReificationDataProperty>
  <ontomat:ReificationDataProperty>
    <rdfs:label>about:        http://www.squ.edu.om/sw/Squ-Ontology#Kday        -
http://www.squ.edu.om/sw/Squ-Ontology#phone</rdfs:label>
    <ontomat:AboutIndividual>http://www.squ.edu.om/sw/Squ-
Ontology#Kday</ontomat:AboutIndividual>
    <ontomat:AboutProperty>http://www.squ.edu.om/sw/Squ-
Ontology#phone</ontomat:AboutProperty>
    <ontomat:CreationSource>http://yournamespace.org#xpointer(//point()[68]/range-
to(//point()[89])</ontomat:CreationSource>
  </ontomat:ReificationDataProperty>
```

```xml
    <ontomat:ReificationDataProperty>
    <ontomat:AboutProperty>http://www.squ.edu.om/sw/Squ-
Ontology#last_name</ontomat:AboutProperty>
    <rdfs:label>about:          http://www.squ.edu.om/sw/Squ-Ontology#Kday          -
http://www.squ.edu.om/sw/Squ-Ontology#last_name</rdfs:label>
    <ontomat:CreationSource>http://yournamespace.org#xpointer(//point()[27]/range-
to(//point()[30])</ontomat:CreationSource>
    <ontomat:AboutIndividual>http://www.squ.edu.om/sw/Squ-
Ontology#Kday</ontomat:AboutIndividual>
  </ontomat:ReificationDataProperty>
  <squ:AssociateProfessor          rdf:about="http://www.squ.edu.om/sw/Squ-
Ontology#Kday">
    <rdfs:label>Kday</rdfs:label>
    <squ:last_name>داي</squ:last_name>
    <squ:email> kday@squ.edu.om</squ:email>
    <squ:first_name>خالد</squ:first_name>
    <squ:fax>24413415</squ:fax>
    <squ:full_name>خالد داي</squ:full_name>
    <squ:phone>: 24413333 1418 داخلي</squ:phone>
    <squ:address> قسم الحاسب الالي
      كلية العلوم
جامعة السلطان قابوس
    مسقط   عمان
مكتب بريد 123 الخوض   صندوق بريد 36</squ:address>
  </squ:AssociateProfessor>
  <ontomat:ReificationDataProperty>
    <ontomat:AboutIndividual>http://www.squ.edu.om/sw/Squ-
Ontology#Kday</ontomat:AboutIndividual>
    <ontomat:CreationSource>http://yournamespace.org#xpointer(//point()[250]/range-
to(//point()[266])</ontomat:CreationSource>
    <rdfs:label>about:          http://www.squ.edu.om/sw/Squ-Ontology#Kday          -
http://www.squ.edu.om/sw/Squ-Ontology#email</rdfs:label>
    <ontomat:AboutProperty>http://www.squ.edu.om/sw/Squ-
Ontology#email</ontomat:AboutProperty>
  </ontomat:ReificationDataProperty>
  <ontomat:ReificationDataProperty>
    <ontomat:AboutProperty>http://www.squ.edu.om/sw/Squ-
Ontology#full_name</ontomat:AboutProperty>
    <rdfs:label>about:          http://www.squ.edu.om/sw/Squ-Ontology#Kday          -
http://www.squ.edu.om/sw/Squ-Ontology#full_name</rdfs:label>
    <ontomat:AboutIndividual>http://www.squ.edu.om/sw/Squ-
Ontology#Kday</ontomat:AboutIndividual>
    <ontomat:CreationSource>http://yournamespace.org#xpointer(//point()[22]/range-
to(//point()[30])</ontomat:CreationSource>
  </ontomat:ReificationDataProperty>
  <ontomat:ReificationDataProperty>
    <ontomat:CreationSource>http://yournamespace.org#xpointer(//point()[128]/range-
to(//point()[228])</ontomat:CreationSource>
    <ontomat:AboutIndividual>http://www.squ.edu.om/sw/Squ-
Ontology#Kday</ontomat:AboutIndividual>
```

```
    <ontomat:AboutProperty>http://www.squ.edu.om/sw/Squ-
Ontology#address</ontomat:AboutProperty>
    <rdfs:label>about:        http://www.squ.edu.om/sw/Squ-Ontology#Kday        -
http://www.squ.edu.om/sw/Squ-Ontology#address</rdfs:label>
  </ontomat:ReificationDataProperty>
</rdf:RDF>
```

Appendix G : The Contents of the Control Knowledge

G1: The contents of SquCK displayed as RDF XML output

```
 <rdf:Description rdf:about="http://munahatam/degree/3899_d2">
   <hasDegreeCountry rdf:resource="http://munahatam/country/USA"/>
   <hasDegreeFrom
rdf:resource="http://munahatam/organisationc/University_of_Minnesota "/>
   <hasDegreeSubject rdf:resource="http://munahatam/topic/Computer_Science"/>
   <degree_date>1989</degree_date>
   <hasDegreeName rdf:resource="http://munahatam/Degree/MSc"/>
   <rdf:type rdf:resource="http://munahatam/Squ-Ontology.owl#Degree"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://munahatam/degree/5927_d2">
   <hasDegreeCountry rdf:resource="http://munahatam/country/UK"/>
   <hasDegreeFrom
rdf:resource="http://munahatam/organisation/University_of_Aston_in_ Birmingham"/>
   <hasDegreeSubject rdf:resource="http://munahatam/topic/Computer_Science"/>
   <degree_date>1982</degree_date>
   <hasDegreeName rdf:resource="http://munahatam/Degree/MSc"/>
   <rdf:type rdf:resource="http://munahatam/Squ-Ontology.owl#Degree"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://munahatam/3899">
   <birth_date>01/01/1963</birth_date>
   <rdf:type rdf:resource="http://munahatam/Squ-Ontology.owl#Professor"/>
   <phone>(968) 24141482</phone>
   <full_Name>Khaled Day</full_Name>
   <first_Name> Khaled </first_Name>
   <email>kday@squ.edu.om</email>
   <middle_Initial> N </middle_Initial>
   <fax> (968) 24413415 </fax>
   <hasDegree rdf:resource="http://munahatam/degree/3899_d1"/>
   <hasDegree rdf:resource="http://munahatam/degree/3899_d2"/>
   <last_Name>Day</last_Name>
   <hasHomePage          rdf:resource="http://www.squ.edu.om/sci/Comp/FINAL
SHP/khaled.html"/>
   <address>Dept. of Computer Science, Sultan Qaboos University, PO.
      Box 36Postal Code 123, Muscat, Oman   </address>
   <hasDegree rdf:resource="http://munahatam/degree/3899_d3"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://munahatam/person/5927">
   <phone>(968) 24142223</phone>
   <address>
        Dept. of Computer Science, Sultan Qaboos University, PO. Box 36Postal Code
123, Muscat, Oman
     </address>
   <middle_Initial>S</middle_Initial>
   <first_Name> Muna</first_Name>
   <full_Name>Mona Salman Hatam Al-Jiboori</full_Name>
   <last_Name>Ï¡Ï°Ï┘ à </last_Name>
```

```
    <rdf:type rdf:resource="http://munahatam/Squ-Ontology.owl#Lecturer"/>
    <last_Name>Hatem </last_Name>
    <hasHomePage              rdf:resource="http://www.squ.edu.om/sci/Comp/FINAL
SHP/muna.html"/>
    <full_Name>Muna Salman Hatem Al-Jepoori</full_Name>
    <last_Name>Hatam</last_Name>
    <hasDegree rdf:resource="http://munahatam/degree/5927_d2"/>
    <email>munahatam@squ.edu.om</email>
    <first_Name>Mona</first_Name>
    <last_Name>Hattem </last_Name>
    <fax>(968) 24143415</fax>
    <squ_Id>5927</squ_Id>
    <hasDegree rdf:resource="http://munahatam/degree/5927_d1"/>
    <first_Name>⌐ à⌐ å⌐ ë</first_Name>
    <birth_date>14/11/1951</birth_date>
    <full_Name>⌐ à⌐ å⌐ ë Ï | ⌐ ä⌐ àÏ°⌐ å Ï¡Ï°Ï—⌐ à Ï°⌐ äÏ¼Ï¿⌐ êÏ░⌐ è</full_Name>
  </rdf:Description>
  <rdf:Description rdf:about="http://munahatam/degree/3899_d3">
    <hasDegreeCountry rdf:resource="http://munahatam/country/US"/>
    <hasDegreeFrom
rdf:resource="http://munahatam/organisationc/University_of_Minnesota "/>
    <hasDegreeSubject
rdf:resource="http://munahatam/topic/Parallel_and_Distributed_Computing"/>
    <degree_date>1992</degree_date>
    <hasDegreeName rdf:resource="http://munahatam/Degree/PhD"/>
    <rdf:type rdf:resource="http://munahatam/Squ-Ontology.owl#Degree"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://munahatam/degree/973_d2">
    <hasDegreeCountry rdf:resource="http://munahatam/country/USA"/>
    <hasDegreeFrom rdf:resource="http://munahatam/organisationc/North Carolina "/>
    <hasDegreeSubject rdf:resource="http://munahatam/topic/Computer_Science"/>
    <degree_date>1988</degree_date>
    <hasDegreeName rdf:resource="http://munahatam/Degree/MSc"/>
    <rdf:type rdf:resource="http://munahatam/Squ-Ontology.owl#Degree"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://munahatam/degree/3899_d1">
    <hasDegreeCountry rdf:resource="http://munahatam/country/Tunisia"/>
    <hasDegreeFrom
rdf:resource="http://munahatam/organisationc/Faculty ├®_des_Sciences_de_Tunis "/>
    <hasDegreeSubject rdf:resource="http://munahatam/topic/Computer_Science"/>
    <degree_date>1986</degree_date>
    <hasDegreeName rdf:resource="http://munahatam/Degree/BSc"/>
    <rdf:type rdf:resource="http://munahatam/Squ-Ontology.owl#Degree"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://munahatam/degree/5927-d1">
    <hasDegreeCountry rdf:resource="http://munahatam/country/Iraq"/>
    <hasDegreeFrom
rdf:resource="http://munahatam/organisationc/Baghdad_Universty"/>
    <hasDegreeSubject rdf:resource="http://munahatam/topic/Physics"/>
    <degree_date>1975</degree_date>
```

```xml
    <hasDegreeName rdf:resource="http://munahatam/Degree/BSc"/>
    <rdf:type rdf:resource="http://munahatam/Squ-Ontology.owl#Degree"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://munahatam/degree/973_d3">
    <hasDegreeCountry rdf:resource="http://munahatam/country/UK"/>
    <hasDegreeFrom
rdf:resource="http://munahatam/organisationc/University_of_Sussex "/>
    <hasDegreeSubject rdf:resource="http://munahatam/topic/Artificial_Intelligence"/>
    <degree_date>1993</degree_date>
    <hasDegreeName rdf:resource="http://munahatam/Degree/PhD"/>
    <rdf:type rdf:resource="http://munahatam/Squ-Ontology.owl#Degree"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://munahatam/degree/973_d1">
    <hasDegreeCountry rdf:resource="http://munahatam/country/USA"/>
    <hasDegreeFrom rdf:resource="http://munahatam/organisationc/North_Carolina "/>
    <hasDegreeSubject rdf:resource="http://munahatam/topic/Computer_Science"/>
    <degree_date>1985</degree_date>
    <hasDegreeName rdf:resource="http://munahatam/Degree/BSc"/>
    <rdf:type rdf:resource="http://munahatam/Squ-Ontology.owl#Degree"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://munahatam/973">
    <rdf:type rdf:resource="http://munahatam/Squ-Ontology.owl#AssociateProfessor"/>
    <email>Haiderr@squ.edu.om</email>
    <hasDegree rdf:resource="http://munahatam/degree/973_d2"/>
    <middle_Initial>R   </middle_Initial>
    <full_Name>Haider Ramdan</full_Name>
    <hasDegree rdf:resource="http://munahatam/degree/973_d1"/>
    <last_Name>Al-lawati  </last_Name>
    <first_Name> Haider </first_Name>
    <birth_date>01/03/1961</birth_date>
    <fax> (968) 24413415 </fax>
    <hasHomePage             rdf:resource="http://www.squ.edu.om/sci/Comp/FINAL
SHP/haider.html"/>
    <hasDegree rdf:resource="http://munahatam/degree/973_d3"/>
    <phone>(968) 24141807</phone>
    <address>Dept. of Computer Science, Sultan Qaboos University, PO.
       Box 36Postal Code 123, Muscat, Oman   </address>
  </rdf:Description>
</rdf:RDF>
```

G2: The contents of SquCK displayed as RDF Triples (Subject, Predicate, Object)

This is a fragment of the displayed list; the full list is on SWIS CD

http://munahatam/973
  http://munahatam/Squ-Ontology.owl#email
    "Haiderr@squ.edu.om ".

http://munahatam/973
  http://munahatam/Squ-Ontology.owl#hasDegree
    http://munahatam/degree/973_d2.

http://munahatam/973
  http://munahatam/Squ-Ontology.owl#middle_Initial
    "R.".

Appendix H : The Contents of the SquRDF

H1: The contents of SquRDF displayed as RDF XML output

```
<rdf:Description
rdf:about="http://munahatam/Org/Sultan_Qaboos_University/CoS/DCS">
   <org_name> Department of Computer Science</org_name>
   <rdf:type rdf:resource="http://munahatam/Squ-Ontology.owl#Department"/>
 </rdf:Description>
 <rdf:Description rdf:about="http://munahatam/Person/5927 ">
   <hasWorkFor
rdf:resource="http://munahatam/Org/Sultan_Qaboos_University/CoS/DCS"/>
   <hasJobTitle rdf:resource="http://munahatam/JobTitles/Lecturer"/>
   <hasDegree rdf:resource="http://munahatam/Degree/5927_d2"/>
   <last_name>Hatem</last_name>
   <middle_Initial>S.</middle_Initial>
   <first_Name>Muna</first_Name>
   <rdf:type rdf:resource="http://munahatam/Squ-Ontology.owl#Person"/>
 </rdf:Description>
 <rdf:Description rdf:about="http://munahatam/Org/Al_Ahliyya_Amman_University">
   <hasSubOrg rdf:resource="http://munahatam/Org/FoE"/>
   <org_name>Al-Ahliyya Amman University in </org_name>
   <rdf:type rdf:resource="http://munahatam/Squ-Ontology.owl#University"/>
 </rdf:Description>
 <rdf:Description rdf:about="http://munahatam/Org/Sultan_Qaboos_University/CoS">
   <hasSubOrg
rdf:resource="http://munahatam/Org/Sultan_Qaboos_University/CoS/DCS"/>
   <org_name>College of Science/</org_name>
   <rdf:type rdf:resource="http://munahatam/Squ-Ontology.owl#College"/>
 </rdf:Description>
 <rdf:Description rdf:about="http://munahatam/Degree/5927_d2">
   <degree_date>1982</degree_date>
   <hasDegreeFrom  rdf:resource="http://munahatam/organisationc/University_of_Aston
"/>
   <hasDegreeSubject    rdf:resource="http://munahatam/topic/Computer_Science_with_
Applications"/>
   <hasDegreeName rdf:resource="http://munahatam/Degree/MSc"/>
   <rdf:type rdf:resource="http://munahatam/Squ-Ontology.owl#Degree"/>
 </rdf:Description>
 <rdf:Description rdf:about="http://munahatam/Org/FoE">
   <hasSubOrg rdf:resource="http://munahatam/Org/DCE"/>
   <org_name>Faculty of  Engineering/</org_name>
   <rdf:type rdf:resource="http://munahatam/Squ-Ontology.owl#College"/>
 </rdf:Description>
 <rdf:Description rdf:about="http://munahatam/Person/5927">
   <hasPJobTitle rdf:resource="http://munahatam/JobTitles/Programmer"/>
   <hasPJobTitle rdf:resource="http://munahatam/JobTitles/Project_Leader"/>
   <rdf:type rdf:resource="http://munahatam/Squ-Ontology.owl#Lecturer"/>
   <hasPJobTitle rdf:resource="http://munahatam/JobTitles/Lecturer"/>
   <hasResearchTopic rdf:resource="http://munahatam/Topics/NLP"/>
```

```
        <hasPJobTitle rdf:resource="http://munahatam/JobTitles/System_Analysts"/>
        <rdf:type rdf:resource="http://munahatam/Squ-Ontology.owl#Person"/>
        <hasResearchTopic rdf:resource="http://munahatam/Topics/Database_Systems"/>
        <hasPWorkedFor
rdf:resource="http://munahatam/Org/Al_Ahliyya_Amman_University/FoE/DEC"/>
        <hasResearchTopic rdf:resource="http://munahatam/Topics/Semantic_Web"/>
        <hasResearchTopic
rdf:resource="http://munahatam/Topics/Internet_Programming"/>
      </rdf:Description>
      <rdf:Description rdf:about="http://munahatam/Org/Sultan_Qaboos_University">
        <country>Oman</country>
        <hasSubOrg rdf:resource="http://munahatam/Org/Sultan_Qaboos_University/CoS"/>
        <org_name>Sultan Qaboos University</org_name>
        <rdf:type rdf:resource="http://munahatam/Squ-Ontology.owl#University"/>
      </rdf:Description>
      <rdf:Description rdf:about="http://munahatam/Org/DCE">
        <org_name> Department of Computer Engineering</org_name>
        <rdf:type rdf:resource="http://munahatam/Squ-Ontology.owl#Department"/>
      </rdf:Description>
</rdf:RDF>
```

H2: The contents of SquRDF displayed as RDF Triples

A fragment of the displayed list is show below. Appendix B describes the full list the on

SWIS CD

http://munahatam/Org/Al_Ahliyya_Amman_University
  http://munahatam/Squ-Ontology.owl#hasSubOrg
   http://munahatam/Org/FoE .

http://munahatam/Org/Al_Ahliyya_Amman_University
  http://munahatam/Squ-Ontology.owl#org_name
   "Al-Ahliyya Amman University in " .

http://munahatam/Org/Al_Ahliyya_Amman_University
  http://www.w3.org/1999/02/22-rdf-syntax-ns#type
   http://munahatam/Squ-Ontology.owl#University .

http://munahatam/Org/DCE
  http://munahatam/Squ-Ontology.owl#org_name
   " Department of Computer Engineering " .

Appendix I : Communication with The GATE Support Team

I1: The Message Sent to GATE Support Team
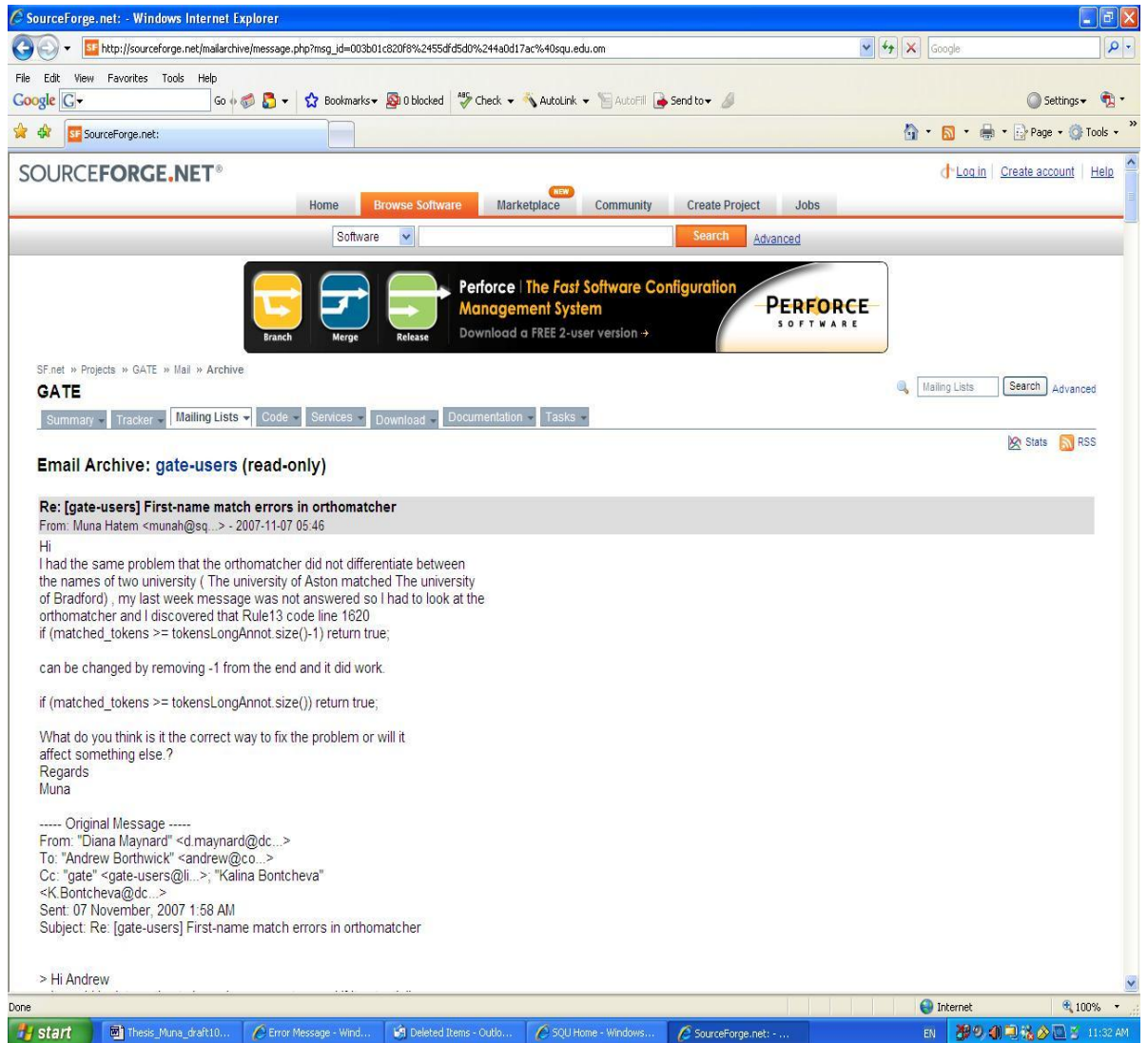
I2: The Message received from GATE Support Team
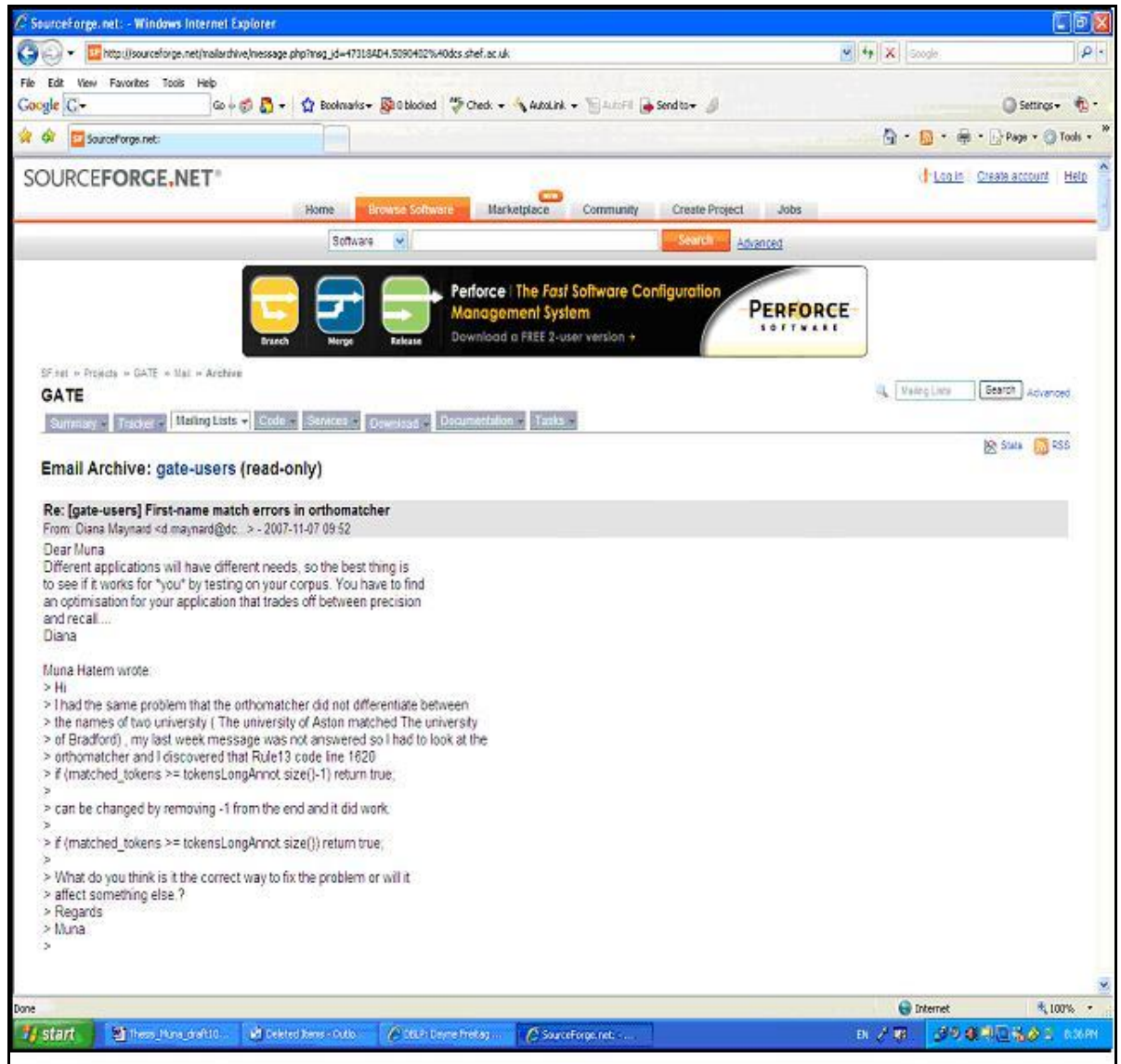
Appendix J: Sample of the new JAPE Rules and Gazetteer lists

J1:Sample JAPE rule: DegreeFinder.jape

```
Phase:  degreeFinder

Input: Lookup

Rule:DegreeConverter

({Lookup.majorType == "Degree"}):match

-->

:match.Degree = {rule = "Degree"},

{

  Annotation lookupAnn = (Annotation)

    ((AnnotationSet) bindings.get("match"))

    .iterator().next();

  inputAS.remove(lookupAnn);

}
```

J2:2Sample Gazetteer list: Degree.lst

```
BSc

MSc

PhD

others
```

Appendix K: Sample XML document produced by GateProcesses

K1: Sample Source Document

> Muna S. Hatem received her MSc degree in Computer Science with Applications from the University of Aston in Birmingham/UK in 1982. She worked as a programmer, System Analyst, Project leader and Computer department manger in different governmental organizations. She was a lecturer in the Faculty of Engineering at Al-Ahliyaa Amman University for 5 year. Mrs. Hatem joined Sultan Qaboos University/ Department of Computer Science in January 2002. Her main research interests are Semantic Web, Internet Programming, and Database Systems. Currently Mrs. Hatem is doing PhD research on Semantic Web at the University of Bradford / UK . Dr.Daniel Neagu is supervising her research. her email address is: munah@squ.edu.om

K2: Annotated Version of the Document in XML form

```xml
- <paragraph xmlns:gate="http://www.gate.ac.uk" gate:gateId="0"
  gate:annotMaxId="545">
- <Sentence gate:gateId="293">
- <Person gate:gateId="327" gender="female" rule1="PersonFull"
  rule="PersonFinal"
  gate:matches="339;334;327;539;540;541;542;543;544">
  <Token gate:gateId="1" category="NNP" string="Muna"
  stem="muna" kind="word" orth="upperInitial"
  length="4">Muna</Token>
  <Token gate:gateId="3" category="NNP" string="S" stem="s"
  kind="word" orth="upperInitial" length="1">S</Token>
  <Token gate:gateId="4" category="." string="." stem="."
  kind="punctuation" length="1">.</Token>
  <Token gate:gateId="6" category="NNP" string="Hatem"
  stem="hatem" kind="word" orth="upperInitial"
  length="5">Hatem</Token>
  </Person>
- <Token gate:gateId="8" category="VBD" string="received"
  stem="receiv" kind="word" orth="lowercase" length="8">
  <VG gate:gateId="355" voice="active" tense="SimPas"
  type="FVG">received</VG>
  </Token>
- <Token gate:gateId="10" category="PRP$" string="her"
  stem="her" kind="word" orth="lowercase" length="3">
  <Person gate:gateId="541"
  gate:matches="339;334;327;539;540;541;542;543;544"
  ENTITY_MENTION_TYPE="PRONOUN"
  antecedent_offset="0">her</Person>
  </Token>
- <Token gate:gateId="12" category="NNP" string="MSc" stem="msc"
  kind="word" orth="mixedCaps" length="3">
  <Degree gate:gateId="347" rule="Degree">MSc</Degree>
  </Token>
  <Token gate:gateId="14" category="NN" string="degree"
  stem="degre" kind="word" orth="lowercase"
```

```xml
    length="6">degree</Token>
    <Token gate:gateId="16" category="IN" string="in" stem="in"
    kind="word" orth="lowercase" length="2">in</Token>
-   <Subject gate:gateId="353" rule="Subject">
    <Token gate:gateId="18" category="NNP" string="Computer"
    stem="comput" kind="word" orth="upperInitial"
    length="8">Computer</Token>
    <Token gate:gateId="20" category="NNP" string="Science"
    stem="scienc" kind="word" orth="upperInitial"
    length="7">Science</Token>
    <Token gate:gateId="22" category="IN" string="with"
    stem="with" kind="word" orth="lowercase"
    length="4">with</Token>
    <Token gate:gateId="24" category="NNS" string="Applications"
    stem="applic" kind="word" orth="upperInitial"
    length="12">Applications</Token>
    </Subject>
    <Token gate:gateId="26" category="IN" string="from"
    stem="from" kind="word" orth="lowercase"
    length="4">from</Token>
    <Token gate:gateId="28" category="DT" string="the" stem="the"
    kind="word" orth="lowercase" length="3">the</Token>
-   <Organization gate:gateId="328" rule1="BaseofOrg"
    rule2="OrgFinal">
    <Token gate:gateId="30" category="NNP" string="University"
    stem="univers" kind="word" orth="upperInitial"
    length="10">University</Token>
    <Token gate:gateId="32" category="IN" string="of" stem="of"
    kind="word" orth="lowercase" length="2">of</Token>
    <Token gate:gateId="34" category="NNP" string="Aston"
    stem="aston" kind="word" orth="upperInitial"
    length="5">Aston</Token>
    </Organization>
    <Token gate:gateId="36" category="IN" string="in" stem="in"
    kind="word" orth="lowercase" length="2">in</Token>
-   <Token gate:gateId="38" category="NNP" string="Birmingham"
    stem="birmingham" kind="word" orth="upperInitial" length="10">
    <Location gate:gateId="329" rule2="LocFinal"
    rule1="InLoc1">Birmingham</Location>
    </Token>
    <Token gate:gateId="39" category="NN" string="/" stem="/"
    kind="punctuation" length="1">/</Token>
-   <Token gate:gateId="40" category="NNP" string="UK" stem="uk"
    kind="word" orth="allCaps" length="2">
    <Location gate:gateId="330" rule1="Location1" rule2="LocFinal"
    gate:matches="341;330" locType="country">UK</Location>
    </Token>
    <Token gate:gateId="42" category="IN" string="in" stem="in"
    kind="word" orth="lowercase" length="2">in</Token>
-   <Token gate:gateId="44" category="CD" string="1982"
    stem="1982" kind="number" length="4">
    <Date gate:gateId="331" rule2="DateOnlyFinal"
    rule1="YearContext1" kind="date">1982</Date>
    </Token>
    <Token gate:gateId="45" category="." string="." stem="."
```

```
        kind="punctuation" length="1">.</Token>
      </Sentence>
-     <Sentence gate:gateId="294">
-     <Token gate:gateId="47" category="PRP" string="She" stem="she"
      kind="word" orth="upperInitial" length="3">
      <Person gate:gateId="544"
      gate:matches="339;334;327;539;540;541;542;543;544"
      ENTITY_MENTION_TYPE="PRONOUN"
      antecedent_offset="0">She</Person>
      </Token>
-     <Token gate:gateId="49" category="VBD" string="worked"
      stem="work" kind="word" orth="lowercase" length="6">
      <VG gate:gateId="356" voice="active" tense="SimPas"
      type="FVG">worked</VG>
      </Token>
      <Token gate:gateId="51" category="IN" string="as" stem="as"
      kind="word" orth="lowercase" length="2">as</Token>
      <Token gate:gateId="53" category="DT" string="a" stem="a"
      kind="word" orth="lowercase" length="1">a</Token>
      <Token gate:gateId="55" category="NN" string="programmer"
      stem="programm" kind="word" orth="lowercase"
      length="10">programmer</Token>
      <Token gate:gateId="56" category="," string="," stem=","
      kind="punctuation" length="1">,</Token>
      <Token gate:gateId="58" category="NNP" string="System"
      stem="system" kind="word" orth="upperInitial"
      length="6">System</Token>
      <Token gate:gateId="60" category="NNP" string="Analyst"
      stem="analyst" kind="word" orth="upperInitial"
      length="7">Analyst</Token>
      <Token gate:gateId="61" category="," string="," stem=","
      kind="punctuation" length="1">,</Token>
      <Token gate:gateId="63" category="NNP" string="Project"
      stem="project" kind="word" orth="upperInitial"
      length="7">Project</Token>
-     <Token gate:gateId="65" category="NN" string="leader"
      stem="leader" kind="word" orth="lowercase" length="6">
      <JobTitle gate:gateId="326" rule="JobTitle1">leader</JobTitle>
      </Token>
      <Token gate:gateId="67" category="CC" string="and" stem="and"
      kind="word" orth="lowercase" length="3">and</Token>
-     <Organization gate:gateId="332" rule2="OrgFinal"
      rule1="OrgXBase">
      <Token gate:gateId="69" category="NNP" string="Computer"
      stem="comput" kind="word" orth="upperInitial"
      length="8">Computer</Token>
      <Token gate:gateId="71" category="NN" string="department"
      stem="depart" kind="word" orth="lowercase"
      length="10">department</Token>
      </Organization>
      <Token gate:gateId="73" category="NN" string="manger"
      stem="manger" kind="word" orth="lowercase"
      length="6">manger</Token>
      <Token gate:gateId="75" category="IN" string="in" stem="in"
      kind="word" orth="lowercase" length="2">in</Token>
```

```
<Token gate:gateId="77" category="JJ" string="different"
stem="differ" kind="word" orth="lowercase"
length="9">different</Token>
<Token gate:gateId="79" category="JJ" string="governmental"
stem="government" kind="word" orth="lowercase"
length="12">governmental</Token>
<Token gate:gateId="81" category="NNS" string="organizations"
stem="organ" kind="word" orth="lowercase"
length="13">organizations</Token>
<Token gate:gateId="82" category="." string="." stem="."
kind="punctuation" length="1">.</Token>
</Sentence>
<Sentence gate:gateId="295">
<Token gate:gateId="84" category="PRP" string="She" stem="she"
kind="word" orth="upperInitial" length="3">
<Person gate:gateId="540"
gate:matches="339;334;327;539;540;541;542;543;544"
ENTITY_MENTION_TYPE="PRONOUN"
antecedent_offset="0">She</Person>
</Token>
<Token gate:gateId="86" category="VBD" string="was" stem="was"
kind="word" orth="lowercase" length="3">
<VG gate:gateId="357" voice="active" tense="SimPas"
type="FVG">was</VG>
</Token>
<Token gate:gateId="88" category="DT" string="a" stem="a"
kind="word" orth="lowercase" length="1">a</Token>
<Token gate:gateId="90" category="NN" string="lecturer"
stem="lectur" kind="word" orth="lowercase"
length="8">lecturer</Token>
<Token gate:gateId="92" category="IN" string="in" stem="in"
kind="word" orth="lowercase" length="2">in</Token>
<Token gate:gateId="94" category="DT" string="the" stem="the"
kind="word" orth="lowercase" length="3">the</Token>
<Token gate:gateId="96" category="NN" string="Faculty"
stem="faculti" kind="word" orth="upperInitial"
length="7">Faculty</Token>
<Token gate:gateId="98" category="IN" string="of" stem="of"
kind="word" orth="lowercase" length="2">of</Token>
<Token gate:gateId="100" category="NNP" string="Engineering"
stem="engin" kind="word" orth="upperInitial"
length="11">Engineering</Token>
<Token gate:gateId="102" category="IN" string="at" stem="at"
kind="word" orth="lowercase" length="2">at</Token>
<Organization gate:gateId="333" rule2="OrgFinal"
rule1="OrgXBase">
<Token gate:gateId="104" category="NNP" string="Al-" stem="al-
" kind="word" orth="upperInitial" length="3">Al-</Token>
<Token gate:gateId="105" category="NNP" string="Ahliyaa"
stem="ahliyaa" kind="word" orth="upperInitial"
length="7">Ahliyaa</Token>
<Token gate:gateId="107" category="NNP" string="Amman"
stem="amman" kind="word" orth="upperInitial"
length="5">Amman</Token>
<Token gate:gateId="109" category="NNP" string="University"
```

```xml
  stem="univers" kind="word" orth="upperInitial"
  length="10">University</Token>
  </Organization>
  <Token gate:gateId="111" category="IN" string="for" stem="for"
  kind="word" orth="lowercase" length="3">for</Token>
  <Token gate:gateId="113" category="CD" string="5" stem="5"
  kind="number" length="1">5</Token>
  <Token gate:gateId="115" category="NN" string="year"
  stem="year" kind="word" orth="lowercase"
  length="4">year</Token>
  <Token gate:gateId="116" category="." string="." stem="."
  kind="punctuation" length="1">.</Token>
  </Sentence>
- <Sentence gate:gateId="296">
- <Person gate:gateId="334" gender="female" rule1="PersonTitle"
  rule="PersonFinal"
  gate:matches="339;334;327;539;540;541;542;543;544">
  <Token gate:gateId="118" category="NNP" string="Mrs"
  stem="mrs" kind="word" orth="upperInitial"
  length="3">Mrs</Token>
  <Token gate:gateId="119" category="." string="." stem="."
  kind="punctuation" length="1">.</Token>
  <Token gate:gateId="121" category="NNP" string="Hatem"
  stem="hatem" kind="word" orth="upperInitial"
  length="5">Hatem</Token>
  </Person>
- <Token gate:gateId="123" category="VBD" string="joined"
  stem="join" kind="word" orth="lowercase" length="6">
  <VG gate:gateId="358" voice="active" tense="SimPas"
  type="FVG">joined</VG>
  </Token>
- <Organization gate:gateId="335" rule2="OrgFinal"
  rule1="joinOrg">
  <Token gate:gateId="125" category="NNP" string="Sultan"
  stem="sultan" kind="word" orth="upperInitial"
  length="6">Sultan</Token>
  <Token gate:gateId="127" category="NNP" string="Qaboos"
  stem="qaboo" kind="word" orth="upperInitial"
  length="6">Qaboos</Token>
  <Token gate:gateId="129" category="NNP" string="University"
  stem="univers" kind="word" orth="upperInitial"
  length="10">University</Token>
  </Organization>
  <Token gate:gateId="130" category="NN" string="/" stem="/"
  kind="punctuation" length="1">/</Token>
- <Organization gate:gateId="336" rule2="OrgFinal"
  rule1="GazOrganization">
  <Token gate:gateId="132" category="NNP" string="Department"
  stem="depart" kind="word" orth="upperInitial"
  length="10">Department</Token>
  <Token gate:gateId="134" category="IN" string="of" stem="of"
  kind="word" orth="lowercase" length="2">of</Token>
- <Subject gate:gateId="354" rule="Subject">
  <Token gate:gateId="136" category="NNP" string="Computer"
  stem="comput" kind="word" orth="upperInitial"
```

```xml
length="8">Computer</Token>
<Token gate:gateId="138" category="NNP" string="Science"
stem="scienc" kind="word" orth="upperInitial"
length="7">Science</Token>
</Subject>
</Organization>
<Token gate:gateId="140" category="IN" string="in" stem="in"
kind="word" orth="lowercase" length="2">in</Token>
<Date gate:gateId="337" rule2="DateOnlyFinal" rule1="DateName"
kind="date">
<Token gate:gateId="142" category="NNP" string="January"
stem="januari" kind="word" orth="upperInitial"
length="7">January</Token>
<Token gate:gateId="144" category="CD" string="2002"
stem="2002" kind="number" length="4">2002</Token>
</Date>
<Token gate:gateId="145" category="." string="." stem="."
kind="punctuation" length="1">.</Token>
</Sentence>
<Sentence gate:gateId="297">
<Token gate:gateId="147" category="PRP$" string="Her"
stem="her" kind="word" orth="upperInitial" length="3">
<Person gate:gateId="542"
gate:matches="339;334;327;539;540;541;542;543;544"
ENTITY_MENTION_TYPE="PRONOUN"
antecedent_offset="354">Her</Person>
</Token>
<Token gate:gateId="149" category="JJ" string="main"
stem="main" kind="word" orth="lowercase"
length="4">main</Token>
<Token gate:gateId="151" category="NN" string="research"
stem="research" kind="word" orth="lowercase"
length="8">research</Token>
<Token gate:gateId="153" category="NNS" string="interests"
stem="interest" kind="word" orth="lowercase"
length="9">interests</Token>
<Token gate:gateId="155" category="VBP" string="are"
stem="are" kind="word" orth="lowercase" length="3">
<VG gate:gateId="359" voice="active" tense="SimPre"
type="FVG">are</VG>
</Token>
<Topic gate:gateId="349" rule="Topic">
<Token gate:gateId="157" category="NNP" string="Semantic"
stem="semant" kind="word" orth="upperInitial"
length="8">Semantic</Token>
<Token gate:gateId="159" category="NNP" string="Web"
stem="web" kind="word" orth="upperInitial"
length="3">Web</Token>
</Topic>
<Token gate:gateId="160" category="," string="," stem=","
kind="punctuation" length="1">,</Token>
<Topic gate:gateId="350" rule="Topic">
<Token gate:gateId="162" category="NN" string="Internet"
stem="internet" kind="word" orth="upperInitial"
length="8">Internet</Token>
```

```xml
<Token gate:gateId="164" category="NN" string="Programming"
stem="program" kind="word" orth="upperInitial"
length="11">Programming</Token>
</Topic>
<Token gate:gateId="165" category="," string="," stem=","
kind="punctuation" length="1">,</Token>
<Token gate:gateId="167" category="CC" string="and" stem="and"
kind="word" orth="lowercase" length="3">and</Token>
<Organization gate:gateId="338" rule2="OrgFinal"
rule1="TheOrgXKey">
<Topic gate:gateId="351" rule="Topic">
<Token gate:gateId="169" category="NNP" string="Database"
stem="databas" kind="word" orth="upperInitial"
length="8">Database</Token>
<Token gate:gateId="171" category="NNPS" string="Systems"
stem="system" kind="word" orth="upperInitial"
length="7">Systems</Token>
</Topic>
</Organization>
<Token gate:gateId="172" category="." string="." stem="."
kind="punctuation" length="1">.</Token>
</Sentence>
<Sentence gate:gateId="298">
<Token gate:gateId="174" category="RB" string="Currently"
stem="current" kind="word" orth="upperInitial"
length="9">Currently</Token>
<Person gate:gateId="339" gender="female" rule1="PersonTitle"
rule="PersonFinal"
gate:matches="339;334;327;539;540;541;542;543;544">
<Token gate:gateId="176" category="NNP" string="Mrs"
stem="mrs" kind="word" orth="upperInitial"
length="3">Mrs</Token>
<Token gate:gateId="177" category="." string="." stem="."
kind="punctuation" length="1">.</Token>
<Token gate:gateId="179" category="NNP" string="Hatem"
stem="hatem" kind="word" orth="upperInitial"
length="5">Hatem</Token>
</Person>
<VG gate:gateId="360" voice="active" tense="PreCon"
type="FVG">
<Token gate:gateId="181" category="VBZ" string="is" stem="is"
kind="word" orth="lowercase" length="2">is</Token>
<Token gate:gateId="183" category="VBG" string="doing"
stem="do" kind="word" orth="lowercase"
length="5">doing</Token>
</VG>
<Token gate:gateId="185" category="NNP" string="PhD"
stem="phd" kind="word" orth="mixedCaps" length="3">
<Degree gate:gateId="348" rule="Degree">PhD</Degree>
</Token>
<Token gate:gateId="187" category="NN" string="research"
stem="research" kind="word" orth="lowercase"
length="8">research</Token>
<Token gate:gateId="189" category="IN" string="on" stem="on"
kind="word" orth="lowercase" length="2">on</Token>
```

```xml
- <Topic gate:gateId="352" rule="Topic">
  <Token gate:gateId="191" category="NNP" string="Semantic"
  stem="semant" kind="word" orth="upperInitial"
  length="8">Semantic</Token>
  <Token gate:gateId="193" category="NNP" string="Web"
  stem="web" kind="word" orth="upperInitial"
  length="3">Web</Token>
  </Topic>
  <Token gate:gateId="195" category="IN" string="at" stem="at"
  kind="word" orth="lowercase" length="2">at</Token>
  <Token gate:gateId="197" category="DT" string="the" stem="the"
  kind="word" orth="lowercase" length="3">the</Token>
- <Organization gate:gateId="340" rule1="BaseofOrg"
  rule2="OrgFinal">
  <Token gate:gateId="199" category="NNP" string="University"
  stem="univers" kind="word" orth="upperInitial"
  length="10">University</Token>
  <Token gate:gateId="201" category="IN" string="of" stem="of"
  kind="word" orth="lowercase" length="2">of</Token>
  <Token gate:gateId="203" category="NNP" string="Bradford"
  stem="bradford" kind="word" orth="upperInitial"
  length="8">Bradford</Token>
  </Organization>
  <Token gate:gateId="206" category="NN" string="/" stem="/"
  kind="punctuation" length="1">/</Token>
- <Token gate:gateId="208" category="NNP" string="UK" stem="uk"
  kind="word" orth="allCaps" length="2">
  <Location gate:gateId="341" rule1="Location1" rule2="LocFinal"
  gate:matches="341;330" locType="country_abbrev">UK</Location>
  </Token>
  <Token gate:gateId="210" category="." string="." stem="."
  kind="punctuation" length="1">.</Token>
  </Sentence>
- <Sentence gate:gateId="299">
- <Person gate:gateId="342" rule="PersonFinal"
  rule1="PersonTitle" gender="male">
  <Token gate:gateId="212" category="NNP" string="Dr" stem="dr"
  kind="word" orth="upperInitial" length="2">Dr</Token>
  <Token gate:gateId="213" category="." string="." stem="."
  kind="punctuation" length="1">.</Token>
  <Token gate:gateId="214" category="NNP" string="Daniel"
  stem="daniel" kind="word" orth="upperInitial"
  length="6">Daniel</Token>
  <Token gate:gateId="216" category="NNP" string="Neagu"
  stem="neagu" kind="word" orth="upperInitial"
  length="5">Neagu</Token>
  </Person>
- <VG gate:gateId="361" voice="active" tense="PreCon"
  type="FVG">
  <Token gate:gateId="218" category="VBZ" string="is" stem="is"
  kind="word" orth="lowercase" length="2">is</Token>
  <Token gate:gateId="220" category="VBG" string="supervising"
  stem="supervis" kind="word" orth="lowercase"
  length="11">supervising</Token>
  </VG>
```

```
- <Token gate:gateId="222" category="PRP$" string="her"
  stem="her" kind="word" orth="lowercase" length="3">
  <Person gate:gateId="543"
  gate:matches="339;334;327;539;540;541;542;543;544"
  ENTITY_MENTION_TYPE="PRONOUN"
  antecedent_offset="546">her</Person>
  </Token>
  <Token gate:gateId="224" category="NN" string="research"
  stem="research" kind="word" orth="lowercase"
  length="8">research</Token>
  <Token gate:gateId="225" category="." string="." stem="."
  kind="punctuation" length="1">.</Token>
  </Sentence>
- <Sentence gate:gateId="300">
- <Token gate:gateId="227" category="PRP$" string="her"
  stem="her" kind="word" orth="lowercase" length="3">
  <Person gate:gateId="539"
  gate:matches="339;334;327;539;540;541;542;543;544"
  ENTITY_MENTION_TYPE="PRONOUN"
  antecedent_offset="546">her</Person>
  </Token>
  <Token gate:gateId="229" category="NN" string="email"
  stem="email" kind="word" orth="lowercase"
  length="5">email</Token>
  <Token gate:gateId="231" category="NN" string="address"
  stem="address" kind="word" orth="lowercase"
  length="7">address</Token>
- <Token gate:gateId="233" category="VBZ" string="is" stem="is"
  kind="word" orth="lowercase" length="2">
  <VG gate:gateId="362" voice="active" tense="SimPre"
  type="FVG">is</VG>
  </Token>
  <Token gate:gateId="234" category=":" string=":" stem=":"
  kind="punctuation" length="1">:</Token>
- <Address gate:gateId="343" rule2="EmailFinal"
  rule1="Emailaddress1" kind="email">
  <Token gate:gateId="236" category="NN" string="munah"
  stem="munah" kind="word" orth="lowercase"
  length="5">munah</Token>
  <Token gate:gateId="237" category="IN" string="@" stem="@"
  kind="punctuation" length="1">@</Token>
  <Token gate:gateId="238" category="NN" string="squ" stem="squ"
  kind="word" orth="lowercase" length="3">squ</Token>
  <Token gate:gateId="239" category="." string="." stem="."
  kind="punctuation" length="1">.</Token>
  <Token gate:gateId="240" category="NN" string="edu" stem="edu"
  kind="word" orth="lowercase" length="3">edu</Token>
  <Token gate:gateId="241" category="." string="." stem="."
  kind="punctuation" length="1">.</Token>
  <Token gate:gateId="242" category="NN" string="om" stem="om"
  kind="word" orth="lowercase" length="2">om</Token>
  </Address>
  </Sentence>
  </paragraph>
```

Appendix L: Sample output of the Semantic Analyser-part 1 when processing the text document in Appendix E1

```
stat GI    Name          Stem      Voice     start End   gate: matches                              UURI

1    327   Person                            0     13    339;334;327;539;540;541;542;543;544        http://munahatam/Person/5927
1    355   VG            receiv    active    14    22    null
1    541   Person                            23    26    339;334;327;539;540;541;542;543;544        http://munahatam/Person/5927
1    347   Degree                            27    30    null
1    16    pp                                38    40    null
1    353   Subject                           41    57    null
1    26    pp                                76    80    null
1    328   Organization                      85    104   null
1    36    pp                                105   107   null
1    329   Location                          108   118   null
1    330   Location                          119   121   341;330                                    http://munahatam/country/UK
1    42    pp                                122   124   null
1    331   Date                              125   129   null
2    544   Person                            131   134   339;334;327;539;540;541;542;543;544        http://munahatam/Person/5927
2    356   VG            work      active    135   141   null
2    51    pp                                142   144   null
2    326   JobTitle                          183   189   null
2    332   Organization                      194   213   null
2    75    pp                                221   223   null
3    540   Person                            262   265   339;334;327;539;540;541;542;543;544        http://munahatam/Person/5927
3    357   VG            was       active    266   269   null
3    92    pp                                281   283   null
3    98    pp                                296   298   null
3    102   pp                                311   313   null
3    333   Organization                      314   341   null
3    111   pp                                342   345   null
4    334   Person                            354   364   339;334;327;539;540;541;542;543;544        http://munahatam/Person/5927
4    358   VG            join      active    365   371   null
4    335   Organization                      372   396   null
4    354   Subject                           412   428   null
4    140   pp                                429   431   null
4    337   Date                              432   444   null
5    542   Person                            446   449   339;334;327;539;540;541;542;543;544        http://munahatam/Person/5927
5    359   VG            are       active    474   477   null
5    349   Topic                             478   490   null
5    350   Topic                             492   512   null
5    351   Topic                             518   534   null
6    339   Person                            546   556   339;334;327;539;540;541;542;543;544        http://munahatam/Person/5927
6    360   VG            do        active    557   565   null
6    348   Degree                            566   569   null
6    189   pp                                579   581   null
6    352   Topic                             582   594   null
6    195   pp                                595   597   null
6    340   Organization                      602   624   null
6    341   Location                          628   630   341;330                                    http://munahatam/country/UK
7    342   Person                            633   648   null
7    361   VG            supervis  active    649   663   null
7    543   Person                            664   667   339;334;327;539;540;541;542;543;544        http://munahatam/Person/5927
8    539   Person                            678   681   339;334;327;539;540;541;542;543;544        http://munahatam/Person/5927
8    362   VG            is        active    696   698   null
8    343   Address                           700   716   null
```

Appendix M: List of Abbreviations

ACE	Automatic Content Extraction
ANNIE	A Nearly-New Information Extraction
API	Application Programming Interface
B2B	Business to Business
CK	Control Knowledge
CO	COreference
CPSL	Common Patten Specification Language
CREOLE	Collection of Reusable Objects for Language Engineering
DAML	DARPA Agent Markup Language
DARPA	Defence Advanced Research Projects Agency
GATE	General Architecture for Text Engineering
HCI	Human Computer Interaction
HLT	Human Language Technology
IE	Information Extraction
IR	Information Retrieval
JAPE	Java Annotation Pattern Engine
KA	Knowledge Acquisition
LHS	Left Hand Side
LMS	Learning Management System
LP^2	Learning Patterns by Language Processing
MUC	Message Understanding Conference
NE	Named Entity
KB	Knowledge Based
OCLC	Online Computer Library Corporation
OIL	Ontology Interchange Language
OWL	Web Ontology Language
POS	Part Of Speech
RDF	Resource Description Framework
RHS	Right Hand Side
RDQL	RDF Data Query Language
RQL	RDF Query Language
SPARQL	RDF Query Language and Protocol
SQU	Sultan Qaboos University
SSE	Semantic Search Engine
ST	Scenario Template
SWIF	Semantic Web Implementation Framework
SWIS	Semantic Web Implementation System
TE	Template Element
TR	Template Relation
UURI	Unique Uniform Resource Identifier
W3C	World Wide Web Consortium (W3C)
XML	Extensible Markup Language