THESIS

ONTOLOGY ENGINEERING: THE BRAIN GENE ONTOLOGY CASE STUDY

Submitted by

Yufei Wang

The Knowledge Engineering and Discovery Research Institute

(KEDRI)

In partial fulfillment of the requirements

for the degree of Master of Computer and Information Sciences

Auckland University of Technology

March 2007

Abstract

The emergence of ontologies has marked another stage in the evolution of knowledge engineering. In the biomedical domain especially, a notable number of ontologies have been developed for knowledge acquisition, maintenance, sharing and reuse from large and distributed databases in order to reach the critical requirements of biomedical analysis and application. This research aims at the development of a Brain Gene Ontology by adopting a constructive IS methodology which tightly combines the processes of ontology learning, building, reuse and evaluation together.

Brain Gene Ontology is a part of the BGO project that is being developed by KEDRI (Gottgtroy and Jain, 2005). The objective is to represent knowledge of the genes and proteins that are related to specific brain disorders like epilepsy and schizophrenia. The current stage focuses on the crucial neuronal parameters such as AMPA, GABA, CLC and SCN through their direct or indirect interactions with other genes and proteins. In this case, ontological representations were able to provide the conceptual framework and the knowledge itself to understand more about relationships among those genes and their links to brain disorders. It also provided a semantic repository of systematically ordered molecules concerned. The research adopts Protégé-Frames, which is an open source ontology tool suite for BGO development. Some Protégé plug-ins were also used to extend the applicable functions and improve knowledge representation.

Basically, the research discusses the availability and the framework of the constructive Information System research methodology for ontology development, it

also describes the process that bridges different notions of the brain, genes and proteins in various databases, and illustrates how to build and implement the ontology with Protégé-Frames and its plug-ins. The results of the BGO development proved that the constructive IS methodology does help to fill in the cognitive gap between domain users and ontology developers, the extensible, component-based architectures of Protégé-Frames significantly support the various activities in the ontology development process, and through explicitly specifying the meaning of fundamental concepts and their relations, ontology can actually integrate knowledge from multiple biological knowledge bases.

Acknowledgement

I wish to thank all the people who gave me the possibility to complete this thesis. Firstly I want to express my gratitude to the Knowledge Engineering and Discovery Research Institute (KEDRI) for providing me chance to commence this thesis, and necessary resources for the research work.

I would like to express once again my gratitude to the foundation director of KEDRI, Professor Nikola Kasabov and the program leader of the School of Computer and Information Sciences at AUT, Krassie Petrova who gave and confirmed the permission and encouraged me to go ahead with my thesis.

I deeply appreciate my supervisor, Paulo Gottgtroy for his stimulating suggestions and encouragement that helped me in all the time of research and writing of this thesis.

Also, I would like to thank my co-supervisor Vishal Jain who involved me in the KEDRI project in the first place. He answered me many questions at the biological aspect.

Especially, I would like to thank my family, thank my parents and my sisters who un conditionally support and encourage me to pursue my study and interests in these years. I am greatly indebted to my father, Zhichun Wang and my mother, Yuqin Lin whose personality always affects and benefits me.

Last but not least, let me say “thank you” to my girl friend Manqian Wang whose trust and love enabled me to complete this work.

Table of content

List of Figures:

List of Tables

List of Abbreviations

AMPA	=	amino-methylisoxazole-propionic acid
BGO	=	Brain Gene Ontology
CLC	=	chloride voltage-gated channel
GABA	=	gamma-aminobutyric acid
GO	=	Gene Ontology
IS	=	Information System
KCN	=	kalium (potassium) voltage-gated channel
KR	=	Knowledge Representation
NCBI	=	National Center of Biotehcnology Information
OBO	=	Open Biomedical Ontologies
OKBC	=	Open Knowledge Base Connectivity
OWL	=	Web Ontology Language
RDF	=	Resource Description Framework
SUO	=	the Standard Upper Ontology
UMLS	=	Unified Medical Language System

Chapter 1 - Introduction

1.1 Background Overview

In recent years, modern biology has been developing fast and producing daily a huge quantity of heterogeneous biomedical data. Biomedical knowledge can be represented as nucleic acid sequences, EEG signals or chemical properties etc. Furthermore, discovering knowledge and sharing it for various purposes has always been complex and crucial in the biomedical domain. Such an explosion of knowledge and such critical requirements bring challenges to the bioinformatical researchers on acquiring, maintaining, reusing and sharing knowledge from those large and distributed databases. Ontology has been proven as a successful way to face the challenge with a conceptual wise.

Ontology is not only a philosophical discipline. It has been widely adopted in the domains of knowledge engineering, computer science and artificial intelligence etc. The definition of ontology varies in different domains as Guarino and Giaretta (1995) suggested. Uschold and Jasper (1999, page 11-2) give the definition of Ontology as: "*An ontology may take a variety of forms, but it will necessarily include a vocabulary of terms and some specification of their meaning. This includes definitions and an indication of how concepts are interrelated which collectively impose a structure on the domain and constrain the possible interpretations of terms*." In the domain of computer and information science, people mainly develop ontologies to share a common understanding of the structures of information (Musen 1992; Gruber 1993); analyze domain knowledge (McGuinness et al. 2000); separate domain knowledge from operational knowledge (Rothenfluh et al. 1996); or to enable reuse of domain knowledge (Bontas et al, 2005). Recently, there has been a large amount of ontologies

developed by different research groups, and different techniques and methods. Similar to software engineering, ontology engineering is a subject related to the activities of ontology development process, languages, methodologies and applications, etc.

To compare with traditional Information Systems, ontologies have much stronger abilities to build the knowledge structure and share knowledge. Therefore, ontology systems can provide more supports for creating semantic webs or other artificial intelligence systems. The reverse between relational databases and ontologies, which enable the data-intensive web pages to emigrate their bases from relational databases to ontologies (Astrova, 2004), are becoming a contemporary topic.

1.2 Research Gaps and Objectives

Previous ontology engineering research provides rich experience on ontology development such as building process analysis (Uschold and King, 1995), identifying ontology applications (Gruninger and Fox, 1995), ontology reuse (Bernaras et al, 1996) and building domain ontologies (Swartout et al, 1997). Some biomedical ontologies have also been built. Nevertheless, the relationships between ontology evaluation, integration, merging and learning are still incomplete because ontology development lacks the comprehensive methodologies and effective tools to fill the cognitive gap between the requirements of domain experts and common information system developers.

This brain gene ontology engineering case is part of the BGO project at KEDRI. It aims at developing and representing the knowledge of genes and proteins that are related to specific brain disorders like epilepsy and schizophrenia. The current stage focuses on the important neuronal parameters such as AMPA, CLC, GABA, and

KCN etc. through their direct or indirect interactions with other genes and proteins. This ontology engineering case answers three research questions as follows:

1. Why the constructive IS research methodology, which has been widely adopted for traditional IS developments, is also suitable for ontology development?
2. How do we apply the selected research methodology in the process of building BGO with the specific tool suite Protégé and plug-ins?
3. Is this BGO system able to integrate the knowledge from multiple biological knowledge bases?

The BGO development process adopts a constructive methodology that builds the theoretical foundation on IS research methodologies, creates BGO through building domain concepts (classes), properties of the concepts (slots) and individuals (instances) with the ontology development tool suite Protégé-Frames.

Protégé-Frames is an ontology editor developed by Stanford Medical Informatics at Stanford University School of Medicine. It is able to provide a user interface and acknowledge server to carry out the main activities of the process of ontology learning, ontology alignment, and ontology evaluation. Consequently, the frame-based domain ontology: BGO, which is designed with the Protégé platform, has the mechanisms to enhance the relationships between all the important stages of ontology development.

Stored knowledge in BGO integrates the gene information from a number of distributed biomedical databases. The data collection and data entry of BGO are based on a series of pre-established criteria. Both the BGO developer and domain experts

build these criteria, which ensures the accuracy and authority of the stored gene knowledge.

1.3 Thesis organization

The following chapter of this thesis consists of a literature review. A number of academic papers have been reviewed to give an introductory overview of ontology and ontology engineering. The second chapter presents the case studies of the most outstanding ontologies and a survey of the literature, which analyzes and evaluates the AI tools in the ontology engineering domain and the unique features of AI tools. The last section of this chapter describes the adopted tool suite Protégé by introducing its architecture, knowledge model and extended plug-ins. Chapter three summarizes the drawbacks of previous approaches, and then constructs a new IS approach for BGO development. This is followed by the criteria used for determining the scope, ontology structure design, reuse of existing sources and BGO structure evaluation and biological data verification. The ontology development architecture and its detailed process are provided in the final section. In chapter four, the evolving BGO is described, including the class hierarchy, key features, and implementation of different plug-ins. The various types of slots and widgets are also introduced. Chapter 5 presents the results of this research, discusses the research efforts and its limitations. Finally, chapter six presents the conclusion and outlines future practice and research directions.

Chapter 2 - Literature Review

2.1 Ontology and Ontological Engineering

From ontology to ontology engineering has been a long process. In philosophy, ontology means the systematic explanation of being. In ancient Greece philosophers proposed problems which were concerned with finding the essence of things through change (Gomez-Perez et al, 2004a). These problems involve the basic concepts inside of ontology such as how can we classify the entities of the world? Is the concept outside a person's mind? And is there an essence that can remain inside the thing although some properties have changed? According to Marias (2001), universals became the key issue of ontology in the Middle Ages. The question was whether universals are actual things. Such "universals" are the basis of the classes or concepts in knowledge modelling. The discussion and analysis of universals also benefited the development of intelligent symbol management in modern information science.

Marias (2001) also noted the series of investigations and discussions in the modern age, which are strongly related to the foundation of ontology theory. In this period, the discussion had been focused on how people's minds capture reality and how people explain the mind structure that is used to capture the reality. There are three important theoretical efforts towards an answer of these questions.

1. Emmanuel Kant (1724-1804) stated the essence of a thing is not only determined by the thing itself but also determined by how one perceives and understands it. This is one of the fundamental theories of modern information systems, that is; the objects in an information system are not only dependent on reality but also dependent on the design of the information system. Kant proposed a framework to describe the mind structure that was used by a person to percieve reality. The

framework is based on the logic classification of judgments and can be organized into four classes: quality (negation, limitation and reality), quantity (unity, plurality and totality), modality (possibility, necessity and existence) and relation (inherence, causality and community). Through these patterns, people's minds can classify an object's existence, uniqueness and commonness etc. This provides the basis for how information systems capture and stores information about objects in realities.

2. Jose Ortega y Gasset (1883-1955) pointed out that the development of a knowledge base must respect the differences within other knowlege bases when they represent the same objects because the object is strongly dependent on the person who perceives it. In other words, different knowledge may perceive objects in different ways or with different features.
3. William James (1842-1910) demonstrated that the truth of the world is the most proper consequences people have considered. Following his theory, today's knowledge structure and databases in information systems are not developed to faithfully represent the world but to more efficiently serve the system itself.

Besides these discussions, the concept of frame was proposed in 1975 by Minsky, he explained that a frame in a frame system represents a concept or an objective. There is a collection of properties or concepts (slots) on the frame, and the slots are filled with initial values. Minsky further interpreted that the slots can be used to represent the possible raised questions in a hypothetical situation represented by the frame and changing the values of slots can make the frame correspond to the particular situation. Based on Minsky's concept, the frame system subsequently gained ground as a basic tool for knowledge representation. (See section 2.2.1)

With the rapid development of computer science, ontologies have emerged as a very important research field at the end of 20th century. Formal ontology is believed to be able to provide a theoretical foundation for the architecture of modern information systems. Nino Cocchiarella (1991) defined formal ontology as the formal and systematic development of the logic of all modes of being. This was agreed with by Nicola Guarino (1998) in his discussion, which regards the related terminologies of ontology and the relationship between formal ontology and information systems. He stated that the word, "formal ontology" normally belongs to the area of philosophical research, but ontologies are sometimes called "formal" when they are used for integration with knowledge bases. Sowa (1999) clarified the difference between "formal" and "informal" ontology: the informal ontology is specified by an undefined catalogue of types or only defined by the statements in natural language, whereas the formal ontology is specified by a collection of names that belongs to the concepts and the relation types are organized in a type-subtype relation.

Due to its long history and the various interpretations of ontology, it is important to clarify the ontological glossaries that are related to the knowledge engineering part of this paper. In fact, the discussions and the investigations on how to define modern ontology have continued since the 1990s. Neches et al (1991) first proposed the idea of building an ontology. They believed that an ontology is able to define concepts, the relationships between the concepts and the rules that combine the concepts and relationships together in a topic area. The most quoted definition was summarized by Studer et al (1998, page 185, para 3) as follows: "*An ontology is a formal, explicit specification of a shared conceptualization.…*" Basically, they merged Gruber (1993a)

and Borst (1997)'s definitions, and they also provided explanations for the words "formal", "explicit" and "conceptualization" etc. in detail.

During the same period, other ontology developers focused on the other aspects. Guarino worked on the terminological clarification for ontology domain. He and Giaretta (1995) collected and explained the seven definitions of ontology (Figure 2.1) in order to clarify the different meanings of ontology in different fields, ontology as a philosophical discipline, a formal semantic account or the vocabulary used by a logical theory etc. They formalized the idea of conceptualization and established the way an ontology developer builds an ontology by making a logical theory: "*A logical theory which gives an explicit, partial account of a conceptualization.*"

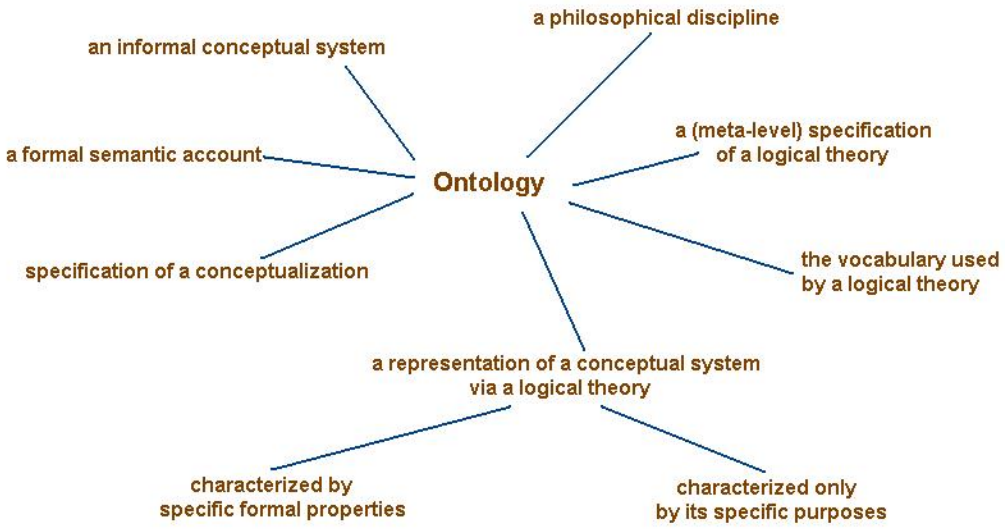


Figure 2.1: The seven definitions of Ontology

In 1996, Bernaras et al gave their ideas on the relationship between ontology and knowledge bases. In their project, which is named KACTUS, they illustrated that ontology provides the conceptualization behind the knowledge represented explicitly

in the knowledge base. This describes the process; an ontology can be extracted from a knowledge base, and it reflects the ideas which the ontology developer adopts to design the ontology, then as more knowledge base applications are built, the more general the ontology becomes.

Although there are many possible definitions for ontology, any ontology or AI community will not misunderstand its use. The different definitions state the same fact. According to Gomez-Perez et al (2004a), ontologies aim to capture consensual knowledge in general, and they can be shared and recycled across software applications by different developers. Usually, they are developed by different ontology developers or knowledge base users in different places.

Recently the concept of ontology was introduced to modern applications related to bioinformatics, knowledge management, e-commerce and database development and integration. Ontology has been widely developed and used and many research groups and organizations are working on their ontology development using various methods. They adopt different application platforms, techniques and approaches. Gomez-Perez et al (2004a; page 5 paragraph 3) define ontological engineering as "*the activities that concern the ontology development process, the ontology lifecycle, and the methodologies, tools and language for building ontologies*". In other words, ontological engineering stands for all of the modelling processes that are able to capture realities.

Based on these works, Alphaworks (2007) clarified that how ontology is different from database and other AI technologies: Ontology represents a data model, not a data

repository as it is used to describe an interface with it through which data may be accessed. Furthermore, Paulo et al (2003) illustrated that an ontology, unlike traditional database, builds its fundamental asset on its relative independence of particular applications.

2.2 Ontology types and Domain Ontologies

This part surveys representative ontologies and describes the techniques for their design and features by their types. Such well-known ontologies as Knowledge Representation (KR) Ontologies, upper ontologies and linguistic ontologies are included. This literature survey covered the applied status of ontologies especially in the domain of biology.

2.2.1 Knowledge Representation Ontologies:

Ontologies can be classified by different criteria. Knowledge Representation ontologies are differentiated from other types of ontologies depending on the subject of conceptualization. According to Davis et al (1993), KR ontology reflects the conceptualizations that are underlying knowledge representation formalisms. Guarino and Boldrin (1993) also stated that KR Ontology does not make claims about the world but presents the representational framework for the world. Representative KR ontologies include the Frame Ontology, OKBC Ontology, RDF and RDF schema and DAML+ OIL etc. They all gather the modelling primitives; classes, attributes and relations to formalize knowledge in a knowledge representation paradigm.

- The Frame Ontology: The frame Ontology is one of the most representative of KR Ontologies. As the review in section 2.1 states, Minsky (1975) first proposed the definition of “frame”. Further research on Frame knowledge representation systems focused on the analysis of frame systems design. Fikes and Kehler (1985), Karp (1992) and Chaudhri et al (1998) provided their assumptions and practices on how to build a comprehensive frame system in order to represent knowledge more efficiently. Currently, the frame system technologies are boosted by the Knowledge Systems Laboratory at Stanford

University and aims at capturing Knowledge Representation conventions based on a frame approach in Protégé (Gennari et al, 2003).

The Frame Ontology allows both relational and object-centered styles of representation to coexist with the aid of KIF)(Knowledge Interchange Format) (Genesereth and Fikes, 1992) language. The significance of the Frame Ontology is that it basically unified the frequently used semantics of the primitives. In Frame Ontology, the classes are co-extensional with unary relations, so in the example of relational style (*-person Bred) is compatible with the object style (instance *-of Bred person). Furthermore, Frame ontology defines slots with a minimal commitment to slots and does not provide any commitment to make distinctions among slots, attributes and properties as local to class, so that its slots can be seen as unary functions and binary relations.

- OKBC ontology: Through analysis of the history of OKBC ontology, Gomez-Perez (2004b) discussed the relationship between OKBC (Open Knowledge Base Connectivity) ontology and Frame Ontology. Before the OKBC was developed in 1997, many ontology languages had their own KR ontology. These languages such as CycL (Lenat and Guha, 1989), OCML (Shadbolt et al, 1993), Ontolingua (Gruber, 1993b) and LOOM (MacGregor, 1991) etc are all frame-based and have similar features. In order to design a frame-based protocol for accessing knowledge bases stored in different languages, the Knowledge Systems Laboratory of Stanford University cooperated with the Artificial Intelligence Centre of SRI International and developed the OKBC Ontology.

The birth of OKBC ontology unified the standards of the various ontology languages. To compare it with a Frame Ontology, OKBC primitives are only concerned with classes, slots and frames. Currently, OKBC ontology is available in the Ontolingua server‘s library. It contains eight classes, 36 relations and three functions and has enhanced the performance of Frame Ontology.

- RDF and RDF schema KR Ontology: Resource Description Framework is a language, which is recommended by the World Wide Web Consortium (W3C) and is developed for Web information representation. According to the primer edited by Manola and Miller (2004), RDF is able to describe common resources from the World Wide Web, for example: the basic information about a web article (authors, article title, copyright etc). Conen and Klapsing (2001) stated that the data model of RDF is the same as the semantic network KR paradigm. They explained that the main components of the RDF data model: resource, properties and statements have a similar functionality with the semantic network’s node which represents concepts instances and attributes’ value, and edges which represent attributes and relations between concepts.

  In RDF Ontology, a statement can be displayed in any order. However, RDF Ontology cannot provide primitives to define the relationship between properties and resources. As a result, RDF schema (RDFS) was developed to solve this drawback (Brickley and Guha, 2004): RDF was extended by W3C with the frame-based primitives.

- DAML+OIL KR ontology: DAML+OIL KR ontology is developed by DAML+OIL which extends RDF(S). DAML+ OIL (Connolly et al, 2001) is known as a semantic makeup language with extended data types and nominals

(Horrocks et al, 1999). As a result, DAML+OIL KR Ontology extended RDF(S) with more modelling primitives: 14 classes, 38 properties and one instance. Gomez-Perez et al (2004b) listed the primitives (classes) that DAML+OIL KR ontology defines for extending RDF(S) ontology as shown in Figure 2.2.1.

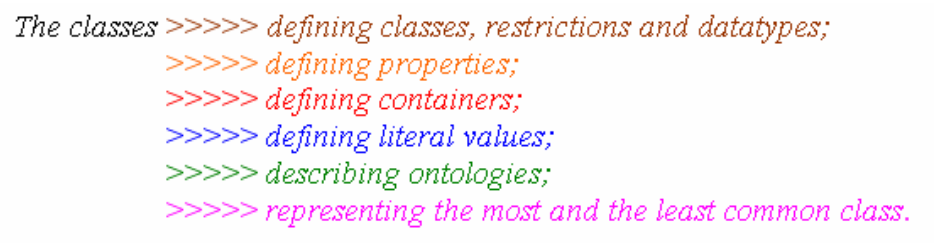


Figure 2.2.1: Class taxonomy of the DAML+OIL KR ontology

The class expressions of DAML+OIL are constructed with the properties of Knowledge Representation primitives. Besides this usage, Properties are also able to define other relationships among ontology components.

- OWL ontology: Web Ontology Language is built on top of RDF, and it has become the recommended language of the W3C for processing information on the semantic web since 2004. Basically, OWL is not the same as RDF, but has greater machine interpretability, a larger vocabulary and stronger syntax than RDF (Knublauch et al, 2004). There are three sub-language of OWL: OWL Lite, OWL DL and OWL Full. Protégé ontology editor also supports OWL ontology design.

2.2.2 Upper Ontologies

Ontology developers have been working for years to create Ontologies that can describe the concepts across domains in a very general way and provide links to all the terms in existing ontologies. This kind of Ontology is an Upper Ontology. Upper

Ontologies can also be called Foundation Ontologies or Top-level Ontologies. Figure 2.2.2 shows Sowa's (1999) upper level categories, which were derived from various sources such as logic, philosophy and artificial intelligence.

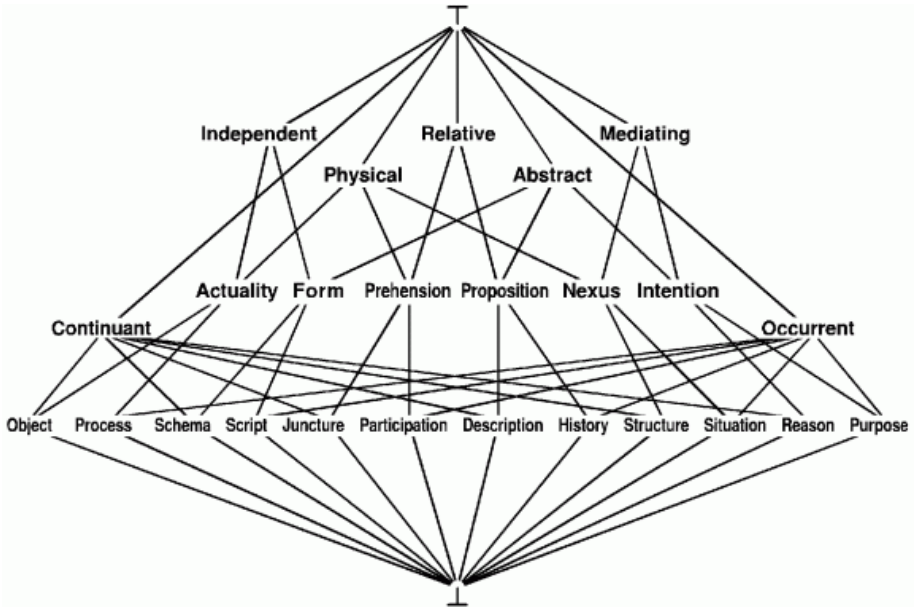


Figure 2.2.2: The top-level categories of Sowa's KR ontology (from Sowa, 1999)

The inner structure of this Upper Ontology reflects two characteristics: "universal" and "articulate". The upper part involves possible instances of the ontology and the lower part contains the subclasses of every concept of the taxonomy.

Such characteristics were addressed in the Cyc project. In 1989, Lenat and Guha emphasized that the Upper Ontology in the Cyc Knowldege Base should be universal: No matter what the background of the ontology development tools are, all the concepts can be linked to the upper level ontology in the correct places. In addition, Matuszek et al (2005) went further and summarized that the Upper Ontology should be articulate:

1) All the concepts have been justified.

2) The amount of concepts is enough for all kinds of applications, for example, knowledge sharing, and database operation.

Another well-known Ontology is SUO (the Standard Upper Ontology), which has been promoted by the IEEE SUO Group since 2000 and was approved as an IEEE standard project. Niles and Pease (2001) described the SUO as able to specify the syntax and semantics of an Upper level Ontology with a very general purpose, and at the same time build the structure for lower level domain ontologies. They stated that there are about one or two thousand terms with ten definitional statements that are the foundation for further (size) and wider (scope) ontology development.

2.2.3 Linguistic Ontologies

The Linguistic Ontologies are mainly related to the semantics of grammatical elements such as words, phrases or any nominal units, and are developed for describing semantic structures. The representative Linguistic Ontologies include WordNet (Miller, 1995), GUM (Generalized Upper Model) (Bateman et al, 1995), EuroWordNet (Vossen, 1998), Mikrokosmos (Mahesh and Nirenburg, 1995) and SENSUS (Swartout et al, 1997) etc. These ontologies are different in some aspects as shown in the Table 2.2.3. Some of them adopt words as their grammatical units, others phrases or the nominal elements that are longer than words. Moreover, they are dependent on language to different degrees, some are even language independent.

WordNet (Miller, 1995) was created by Princeton University and aims at the organization of lexical information in terms of word meaning. As of 2006 (George et al, 2006), it (Version 3.0) contains over 115,000 concepts and more than 150,000

words and has become a huge lexical database for English. The motivation of EuroWordNet is the same as WordNet, but it has European language versions: Dutch, Spanish, German, French, Estonian etc. The University of Amsterdam, UNED (Spain) and the University of Sheffield corporate on EuroWordNet and finished the project in 1999. Unlike other ontologies, Mikrokosmos is designed for a more practical purpose. It is part of a machine translation project in the knowledge merge and acquisition domain.

Name	***WordNet***	***EuroWordNet***	***GUM***	***Mikrokosmos***	***SENSUS***
Grammatical Unit	Word	Word	More than Word	Word	More than word
Language Dependency	Dependent on a single language	Partly dependent	Dependent on several languages	Language independent	Dependenton several languages
Motivations	Online lexical database	Online lexical database	Natural Language Generation	Machine translation	Machine translation
Developers	Princeton University	University of Amsterdam, UNED (Spain) and University of Sheffield	ISI, CNR and GMD	US goverment, the New Mexico State University and Carnegie Mellon University	ISI

Table 2.2.3: Comparison of several Linguistic Ontologies

GUM was firstly developed by the Penman text generation system (Bateman et al, 1995). There are two hierarchies inside GUM: concepts and relations. They have their own grammar, but this does not disturb their application theory. Three universities: ISI (the Information Sciences Institute), CNR (Italy) and GMD (German) took part in the development of GUM. Another similar project is SENSUS. According to Swartout and his colleagues (1997), there are over 70,000 nodes in the SENSUS for commonly encountered representation. That makes SENSUS able to provide a broad

conceptual structure for machine translation and obtain content to extract or merge information from different knowledge bases. It was designed by the Natural Language Group of ISI. Broadly speaking, both GUM and SENSUS can also be seen as Upper Ontology as well due to their abstract concepts.

2.2.4 Domain ontologies and Biomedical Ontologies:

Mizoguchi and his colleagues (1995) introduced the phrase domain ontologies from the perspective of applied ontologies. They stated that domain ontologies are able to capture the knowledge of specific subdomains by defining the reusable vocabularies in the domain concepts and the relationships among these concepts. This section focuses on the developing status and issues regarding the ontologies in the biomedical domain, the representative ontologies in other domains such as e-commerce, engineering and enterprise etc. are also addressed briefly.

According to Bodenreider and his colleagues (2003, page 562), biomedical ontologies organize the concepts "*involved in biological entities and processes in a system of hierarchical and associative relations that allows reasoning about biomedical knowledge.*" As an example, **OBO (Open Biomedical Ontologies)** provides an umbrella web address to organize the controlled vocabularies for sharing and reuse across the various biological and medical domain. Currently, the ontologies of OBO are available in its sourceforge site and can be viewed in the OBO tree browser or in OBO table form.

GO (Gene Ontology) and SO (Sequence Ontology) are both related projects of OBO and are well-known ontologies in the biomedical domain. GO is being developed by the Gene Ontology Consortium. In 2000, Ashburner and his colleagues stated that

their work on the development of the Gene Ontology would finally provide a dynamic, controlled vocabulary to share all knowledge about genes and proteins' roles in cells. They introduced the three sub-ontologies of GO: molecular function, biological process and cellular location and illustrated the importance of GO annotations with examples in GO. The original format of GO is OBO format. Ontological researchers continually developed the new formats of GO in the following years. In 2003, Yeh and his colleagues transferred the GO into Protégé 2000 in order to increase the accessibility of the dramatically expanding GO. Figure 2.2.4 shows the frame-based knowledge model of GO viewed in the OntoViz tab in Protégé 2000. This is also a test to manage and edit the knowledge base with the AI application tool. The GO subsequently provided an OWL version on the OBO website.

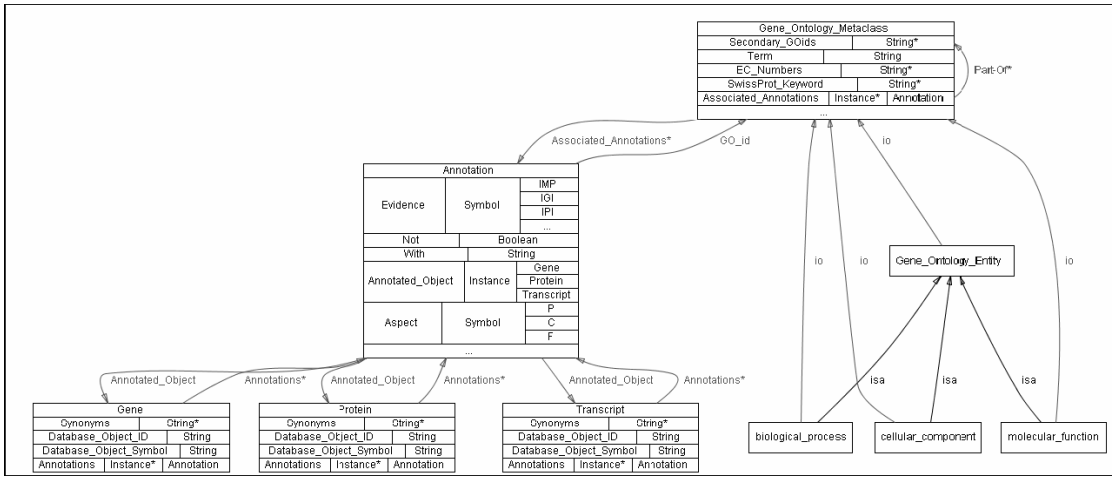


Figure 2.2.4: Frame-based knowledge model of GO (Yeh et al, 2003)

The Sequence Ontology (Eilbeck and Lewis, 2004; Eilbeck et al, 2005) defines the terms that can be used to describe the characteristics of the nucleotide or protein sequences. Such characteristics involve nucleotide similarity hits and gene models etc. Basically, SO is applied to provide the primary annotations for nucleic acid sequence; help the identification and the querying of aspecific gene whose transcript is modified and describe the mutations in the different levels of genomic databases. There are two

available versions of Sequence Ontology on the site: The full version is the normal meaning SO; the “SOFA” version provides the faster portal to the biological sequence.

UMLS (Unified Medical Language System) was developed by the US National Library of Medicine(2003). It is one of the largest knowledge sources for the integration of biomedical terms. UMLS involves three parts to allow researchers to access the knowledge base from the main page:

Metathesaurus: provides the information to describe all of the biomedical terms. There are over 1 million of biomedical terms and 5 million concept names in the Metathesaurus. These data are collected from over 100 controlled vocabularies and classification systems used in patient records, bibliographic and administrative health data and databases (Olivier, 2004).

Semantic Networks: define biomedical concepts and their relationships. In the 14th edition of UMLS (Olivier, 2004), there are totally 134 domain concepts in total and 54 relationships between them.

Specialist Lexicon: provides the information on biomedical terms in order for them to be used in natural language processing applications.

Besides the biomedical domain, ontologies also work actively at other domains. With the development of e-commerce, there is an increasing communication efficiency requirement for B2B applications. The new generation of ontological applications are being developed to classify services and products in vertical domains such as E-cl@ss, UNSPSC and NAICS etc (Gomez-Perez et al, 2004b). The Enterprise Ontology was addressed by Uschold and his colleagues (1998) for defining terms related to business. Aside from the Enterprise Ontology, ontological techniques are also adopted in

software engineering. Falbo et al (2002) explained their ontological approach (ODE) to domain engineering. Through these several projects, ontologies have shown their huge potential to be domain models.

2.3 Artificial Intelligence Tools for Knowledge Modelling

Just like the development of software engineering, the ontology building environment emerged in the mid-1990s to meet challenges in the ontology development process. An environment to provide a user interface was constructed in order to simplify activities in the ontology development process such as conceptualisation, consistency checking and implementation etc. Gomez-Perez (2002) summarized the six main functionalities of ontology building environment in his research, as follows:

1. Ontology Development.
2. Ontology merger and alignment
3. Ontology annotation
4. Ontology evaluation
5. Ontology query and inference
6. Ontology learning

The surveyed development tools suites normally consists of several functions from the above list. Gomez-Perez and his colleagues (2004c) investigated the evolutionary history of ontology tools and found a clear way of dividing the existing ontology development tools into two groups: Language-dependent ontology development tools and extensible language-independent ontology development tools and tool suites. The two groups of ontology tools represent the different stages of evolution. They are

introduced respectively in the following sections and the selected tool Protégé will be discussed in the next section.

2.3.1 Language-dependent ontology development tools

The first generation of ontology tools were developed from the middle of the 1990s. What they have in common is that they are strongly related to their own ontology language. Users are required to edit or browse the ontologies with their corresponding languages. The main tools are the Ontolingua server, WebOnto, Ontosaurus and OilEd. Their corresponding languages are shown in the Table 2.3.1.

Ontology tools	Language	
WebOnto	OCML	(Domingue, 1998)
Ontosaurus	LOOM	(Swartout et al, 1997)
OilEd	OIL	(Goble et al, 2001)
Ontolingua server	Ontolingua	(Farquhar et al, 1997)

Table 2.3.1: Ontology tools and their corresponding languages

As the earliest ontology tool, the Ontolingua server provides a web interface (shown in Figure 2.3.1) to ease the collaborative development of ontologies with the ontolingua language. The ontology editor is the most important toolset. Currently, other toolsets including Webster, OKBC server and Chimaera are available on the server as well. Webster provides the functionality to obtain term definitions; the OKBC server is not involved in the web interface but provides accessibility; and Chimaera is to analyze, merge and integrate ontologies.

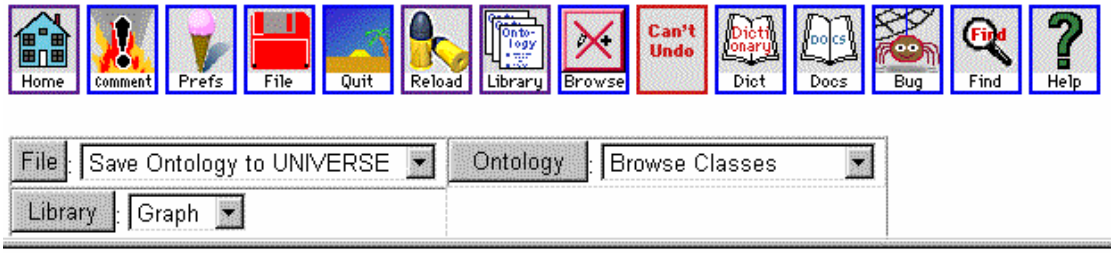


Figure 2.3.1: Web interface of Ontolingua server

The architecture of Ontosaurus is similar to the Ontolingua server. It includes two parts: an ontology server for a KR based system attached with LOOM language; a web interface for ontology editing and browsing. OilEd was designed by the University of Manchester in 2001 and grew with the European IST project On-To-Knowledge. The current version of the ontology editor of OilEd focuses on the DAML+OIL ontologies and does not provide a full ontology development environment. Therefore, the function of OilEd is restricted to an ontology editor only, which works like “NotePad”. WebOnto uses different types of ontology editors with Ontosaurus and Ontolingua. It is based on Java applets, which can provide stronger support for collaborative development of OCML ontologies and allows users to discuss the developing ontologies synchronously and asynchronously by group.

2.3.2 Extensible language-independent ontology development tool suites

There is a problem for ordinary users of the Language-dependent ontology development tools. If the users want to create a medium or large ontology with such ontology tools, they face a language problem; they need to write the language directly into the text box. Any kind of further operation such as writing the expression of an axiom or a relation constraint requires a comprehensive knowledge of the corresponding language. This also increases the difficulty of transferring the developed ontologies from one ontology editor to another.

The new generation of ontology building environments emphasizes extensibility, ontology language independence and the integration of the ontology services. The representative tools are Protégé (discussed in 2.4 in detail) (Noy et al, 2000), KAON tool suite (Maedche et al, 2003), OntoEdit (Sure et al, 2002) and WebODE (Arprez et al, 2003).

Extensible: The above tool suites have similar component-based architectures; the new modules can be deployed and implemented easily by the ontology developers to provide more comprehensive functions: Protégé supports extended plug-ins; WebODE has its own ontology access services; KAON provides an OI-modeler for ontology evolution mapping and generation etc.

Language independent: Their knowledge modules are language independent and support import and export of ontologies in multiple languages: Protégé supports FLogic, Jess, XML, Prolog, OIL etc; OntoEdit supports FLogic, XML, RDF(S) and DAML+OIL etc; WebODE supports CARIN, XML, FLogic, Jess and Prolog etc.

Integrated: Besides the extensible modules, these tools carry a set of ontology related services which integrate more functionality such as ontology annotation, evaluation, middleware service, query and inference etc.

WebODE is a scalable workbench for ontological engineering. It was developed by Universidad Politecnica de Madrid. The latest version of WebODE (2.0) includes an ontology editor, a semantic web portal generator, a web sources annotation tool, a semantic web services editing tool and a knowledge management system. One of the most significant features of the WebODE workbench is that the application server is able to specify the user or the user group to give access to a specific service.

KAON and OntoEdit are both designed by the Institute AIFB at the University of Karlsruhe. OntoEdit has been commercialized by Ontoprise GmbH. The professional version of OntoEdit has a similar structure to Protégé (easily extended with plug-ins) but is not open source. OntoEdit allows two ways to import and export SQL schemas: the functional module inside OntoEdit or the Java API for ontology access. KAON is still open source from sourceforge. KAON ontologies can be stored in files or relational databases. The web services of KAON allows users to access the APIs by using a web browser.

2.4 Protégé and the extended plug-ins

Protégé (Noy et al, 2000) is an open source ontology development platform and knowledge base framework. It is developed by the SMI (Stanford Medical Informatics) group at Stanford University. There are at least four reasons why this research chooses Protégé as the ontology engineering tool for BGO: Firstly, Protégé integrates the necessary and comprehensive tool suites for ontology development; secondly, the main characteristics of Protégé such as its extensibility and language independence ease the complexity of the development process; thirdly, Protégé is an open source platform and freely available for downloading under the Mozilla license; and finally, since the first Protégé tool was created in 1987, there is a significant amount of research and experiment that can be used as references and it has been proved as a high-performance tool in the biomedical domain (eg. the GO project as described in section 2.2.4).

The Protégé platform supports two ways of ontology development: Protégé-Frames editors and Protégé-OWL editors. They are suitable for the development of frame ontology and OWL ontology (see section 2.2.1) respectively. This section focuses on the Protégé-Frames editor, presents the architecture, knowledge model of Protégé-Frames, and then provides general information about the editor. It also includes the introduction of some important extended plug-ins that have been adopted for BGO development.

2.4.1 Architecture

Gennari et al (2003) reviewed the development of Protégé and evaluated its adopted architecture. As a Java-based standalone application, the essential part of Protégé is the extensible architecture as shown in Figure 2.4.1. On the top of the architecture, the ontology developers are allowed to customize the user interface or build a new interface to interact directly with the knowledge model. To do that, they must adopt two kinds of plug-ins:

Slot widget plug-ins: These are not used to modify or change the default user interface as a whole, but to edit the value type of a single element in the slot form. For example: show a slot value as a picture in GIF or JPEG format or a short video in AVI format etc.

Tab plug-ins: These modify the default user interface in order to add new functions that are not involved in the standard distribution of the ontology editor. They normally appear as additional tabs after Protégé reloads the UI and the ontology developer can access these new functions from the tab. The functions of some important plug-ins such as TGVizTab, UMLTab and PROMPT Tab are described in section 2.4.4.

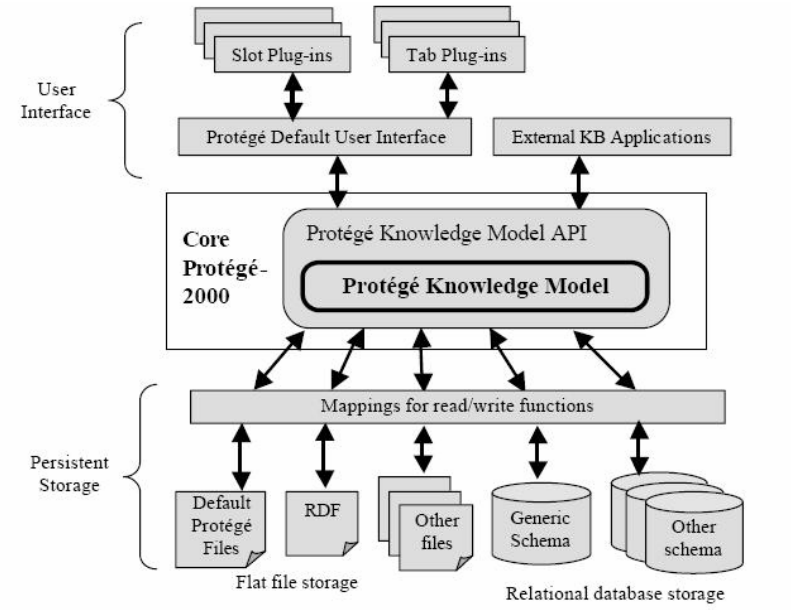


Figure 2.4.1: Protégé architecture (Gennari et al, 2003)

The core part is the ontology editor, which is described in section 2.4.3, and section 2.4.2 explains the Protégé Knowledge Model.

The lower part shows the persistent storage mechanisms of protégé. Protégé has a particular format; CLIPS to store the developed ontology. Otherwise, via the Backends plug-in, it supports ontology export and import in other formats such as, XML, XML schema, RDF, RDF schema etc. As a result, Protégé is able to store and retrieve ontologies from any JDBC compatible database.

2.4.2 Knowledge Model

The knowledge model of Protégé is frame-based and OKBC compatible. The main modelling components include classes, slots, facets and instances. According to Noy et al (2000):

Classes: stand for the domain concepts. They can be concrete or abstract in Protégé and are organized in class hierarchies where multiple inheritances are permitted. That is: If class A is the subclass of B and C, A will inherit all the attributes from both B

and C to be a super class. The root class in Protégé is the build-in class ":thing". Only concrete classes can have instances (individuals).

Slots: stand for the attributes or properties of the domain concepts. The name of a slot is unique in an ontology, but it is class independent: the slot with the same name in different classes does not stand for the same attributes. Slots can have values, for example, a person has a slot "name": Brad Pitt, then Brad Pitt is the value of the name.

Facets: stands for the constraints of slots. Facets define the restriction of value type (e.g. integer, string, float etc.) or the limitation of numeric value (100<slot value<10000) etc.

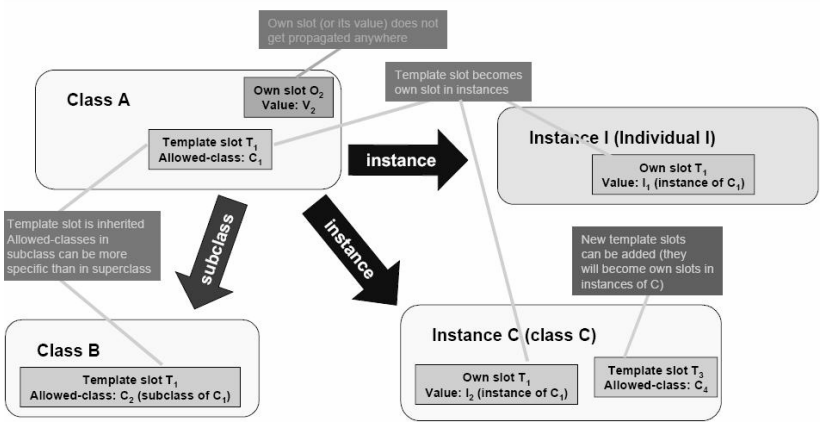


Figure 2.4.2: Propagation of template and own slots (Noy et al, 2000)

Noy et al (2000) mentioned that as in the OKBC ontology, Protégé distinguishes slots into two types: template slots and own slots. This can be seen in Figure 2.4.2. Own slots attach to the frames that describe the attributes of classes or instances when the classes do not get inherited by their subclasses or propagated to their instances. A template slot, by contrast, attaches to the class frames, which are inherited by their own subclasses. Once a template slot on a class frame is in the instances of that class, it will become a slot in its own right.

2.4.3 Ontology Editor

In Protégé, the default user interface of the ontology editor involves five tabs: Classes tab, Slots tab, Forms tab, Instances tab and Query tab. Noy and McGuinness (2001) provide a fundamental ontology development guide of the Wine and Food Ontology. They describe the functions of these default tabs as follows

The classes tab: is for class browsing and editing. It displays and edits the ontology's class taxonomy by adopting a tree structure (see Figure 2.4.3). On the right side of the screenshot, the information regarding the class can be filled in and the template slots are attached to classes. The whole interface supports copy-paste, drag-drop functions and various pop-up menus or windows for different types of ontology components.

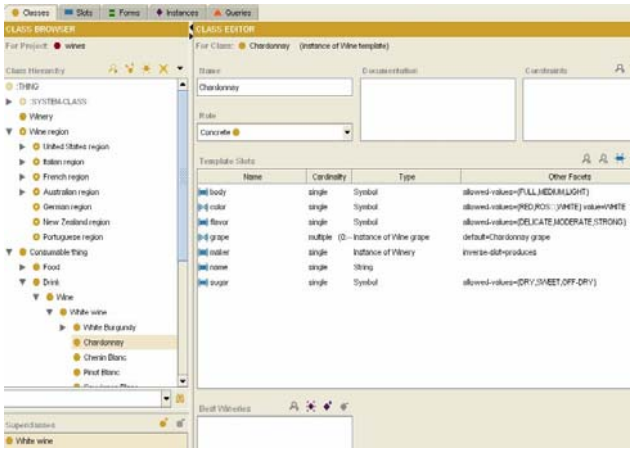


Figure 2.4.3: the screenshot of Classes tab

The slots tab: is for slots browsing and editing. It provides frames to show the slots hierarchy defined in the ontology and to edit slot information includingthe slot's name, value type etc.

The forms tab: is to customize the layout used to create instances. The ontology developers are able to define the type, location and size of the forms.

The instances tab: is for instance browsing and editing by predefined class hierarchy. The layout of the slot widgets can be modified in the forms tab.

The queries tab: is to provide a common search function for the ontology editor. The developer can create queries to search the instances, which have a specific slot value or the slot value in a range. The created queries can be stored, retrieved and combined from the query library.

2.4.4 Significant plug-ins

The architecture of Protégé simplified the new extensions' (plug-ins) creation and integration. The new plug-ins can provide the functions that are not involved in the standard distribution of Protégé. There is a significant number of such plug-ins developed by different research groups in the Protégé plug-in library. This section introduces TGVizTab, UMLS tab and PROMPT tab.

TGVizTab (Alani, 2003) is a new Protégé plug-in to graphically represent the ontologies in Protégé. It's based on the technology of Touchgraph that provides a java library for expressing networks as interactive graphs.

It can be seen from the Figure 2.4.4.1, TGVizTab displays classes and instances. The different classes or instances are distinguished as the different nodes in the graph. The users can navigate the connected sub-graphs by increasing the radius value. Search and zoom functions are provided as well. The generated graphs can be saved as XML, and then be opened by other Toughgraph applications.

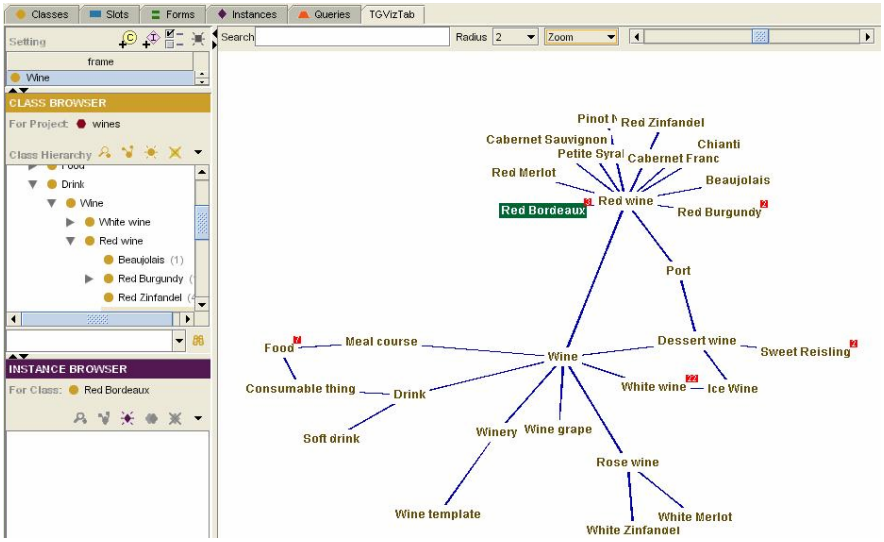


Figure 2.4.4.1: Visual Representation of the Wine and Food Ontology with TGVizTab

UMLTab (Shankar et al, 2002**)** is a Protégé plug-in developed by the University of Stanford. It provides an interface that allows users to query and retrieve domain concepts, terms and even semantic types from the knowledge sources of UMLS (introduced in section 2.2.4).

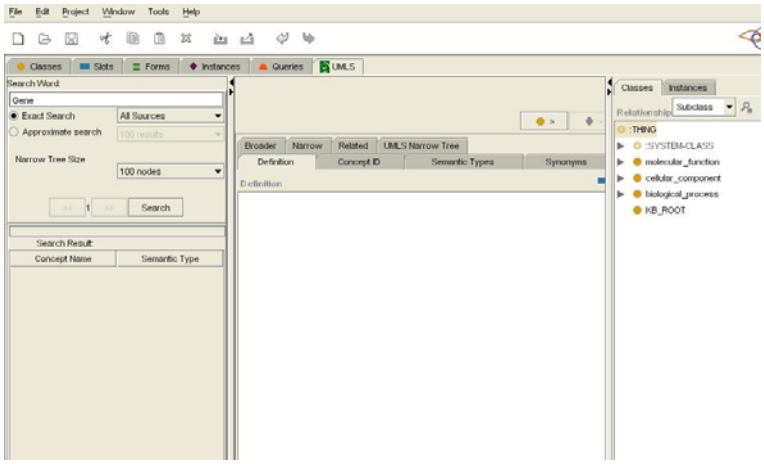


Figure 2.4.4.2: The UMLS tab interface

As shown in Figure 2.4.4.2, UMLSTab has the following functions:

- Connects the UMLS server with Protégé: the users can access the UMLS resources from the UMLS tab interface.

- Supports a query from UMLS sources with term names: users can search specific terms by inputting key words in the search component. (Login UMLS needed)
- Creats classes and instances in Protégé with UMLS content: users can copy or build anontology hierarchy based on information from UMLS.
- Auto-imports the slots value when classes are created from the search results: copying a class or an instance from UMLS ontologies involves their slot value.

PROMPT tab is powerful plug-ins for multiple ontology management. The current version of PROMPT tab provides four optional functions:

1. Compare the current ontology with a different version of the same ontology
2. Move frames between two ontologies including projects in an ontology project.
3. Merge an ontology with a current ontology.
4. Extract part of an ontology and move it to a current ontology.

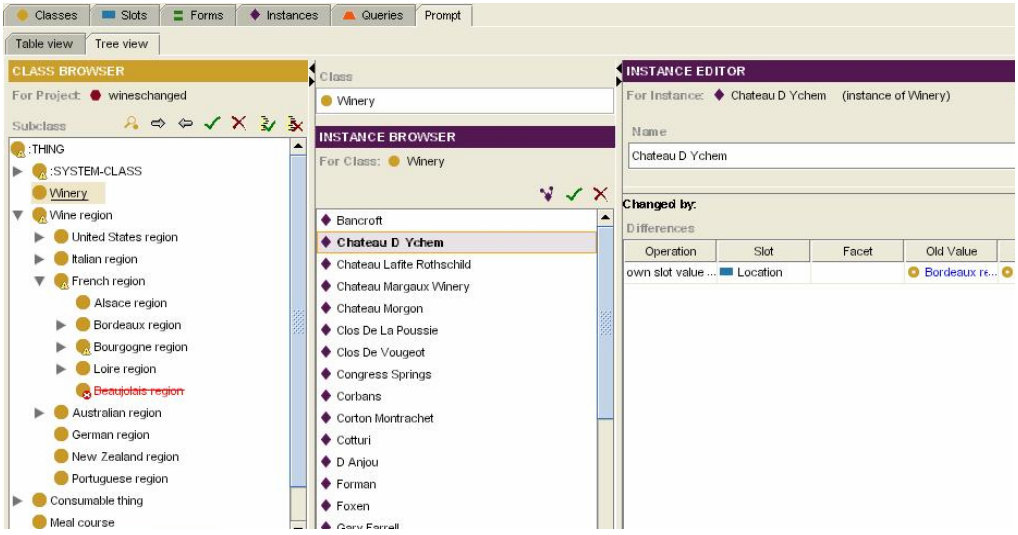


Figure 2.4.4.3: A comparison of different versions of the Wine and Food Ontology in the PROMPT tab

The Figure 2.4.4.3 shows the results of comparing two versions of the Wine and Food Ontology in the PROMPT tab. All the operations regarding the modification of the ontology being compared: the creation, deletion and the value change etc are all

shown clearly. Through the functions of the PROMPT tab, the ontology developers ease the process greatly when they detect the progress of other co-operators, integrate their own work with others, correct incorrect operations that have been saved or focus on only a small part of the ontology.

2.5 Bioinformedical Databases

The gene information of the BGO is extracted from large distributed databases. In order to choose credible sources, it is necessary to analyse these databases. This section introduces three representative bioinformedical databases that provided the authorized data for the instance creations in BGO.

NCBI: (National Center of Biotehcnology Information) NCBI was established in 1988. It has now become one of the most well known molecular biology information centres. The main page of NCBI provides entries for databases of both genome sequencing data and biomedical research literature. NCBI focuses on the understanding of molecular processes affecting human health and disease, and develops sequence search engine and software tools for analysing genome data and biomedical information. As a nucleotide sequence database, GenBank (Mizrachi, 2004) has been supported by NCBI and grown fast since 1992. The 134th release of GenBank contained more than 29.3 billion nucleotide bases in overy 23 million sequences. Now it has other collaborative databases such as the Molecular Modeling Database (3D protein structures), the Unique Human Gene Sequence Collection, a Gene Map of the Human genome etc. (Another important part of NCBI: PubMed is introduced later)

Gene Cards (Chalifa-Caspi et al, 2004; Lancet et al, 2005)**:** Gene Cards is one of the main knowledge sources for BGO development. It integrates the knowledge of human genes and the related annotations in a comprehensive topic range. It also provides the links about specific gene to more than 50 links to other databases. Basically, most known information about a human gene such as utomatically-mined genomic, proteomic and disease relationships can be found with Gene cards. The search engine of Gene Cards supports both simple search and advanced search.

Swiss-Prot (Apweiler, 2001): Further BGO development required gene information on all organisms. Swiss-Prot provides this. It is not only an annotated protein sequence database, but also presents other biochemical information. The annotations that Swiss-Prot provides for sequences are the result of a labor-intensive process that includes assessment information from published articles along with use of a variety of programs and algorithms. By 2000, SWISS-PROT had over 95,000 nucleic acid sequences, and the amount keeps increasing dramatically these days.

PubMed (Canese et al, 2003)**:** Most of the annotation links in BGO are searched through PubMed. PubMed is a biomedical journal database of the U.S. National Library of Medicine. It was built in the1950s and up to now, has included over 16 million biomedical citations and other life science journals. Through PubMed query, researchers can find full text articles and other related resources. An ordinary bioinformatical researcher can gain access from the NCBI home page.

Other used but less important resources in BGO are introduced briefly in the source table in section 3.4.2.

Chapter 3 - Methods and Methodologies

This BGO research aims at the development of the Brain Gene Ontology. According to Gomez-Perez et al, (1996) ontology engineering requires a comprehensive, highly integrated series of techniques and methods to perform the processes of ontology learning, building, evaluating, merging etc. In this chapter, section 3.1 discusses the features and gaps of the existing approaches, and section 3.2 explains why a new approach attached with Protégé was chosen and how this approach was applied. Section 3.3 and 3.4 describes the details of BGO development and the criteria used for building and evaluating BGO respectively.

3.1 Discussions about existing approaches

In the domain of computer and information science, there has been a growing interest in participating in ontology development since the 1990s. As a result, ontology research groups have provided a series of methods and methodologies for ontological engineering. These methodologies defined their activities in the ontology development process, and were used to develop ontologies from scratch, or by reusing other ontologies.

Uschold and King's method

One of the earliest methods for building ontology was summarized by Uschold and King in 1995. Based on their rich experience of enterprise ontologies, they defined the classical four steps that are necessary in ontology development: identify the purpose; build the ontology; evaluate the ontology and document the ontology. However, the conceptualization process, which provides a primary domain model, was not included.

This creates a gap between the understanding of the domains and the implementation of the ontology.

Gruninger and Fox's methodology

By contrast, Gruninger and Fox's methodology (1995) focuses on identifying the possible applications of the ontology, and then determines the scope of the ontology by using a set of competency questions. They extract the concepts, the properties of the concepts, the relationship between concepts and the formal axioms of the ontology from the answers of these competency questions. Gruninger and Fox's method first introduced the first-order logic and provided a guide to create computable models from natural language questions. On the one hand, based on the experience of the TOVE project, Gruninger and Fox's methodology pays more attention to ontology development and management. On the other hand, other important processes like ontology evaluation and integration are not involved.

Amaya Bernaras et al's approach

Amaya Bernaras et al's approach (1996) is based on the KACTUS project. This approach unites ontology development to application development. Once an application is developed, the required term's list will be provided and be reflected to corresponding terms in the top-level ontological categories. The design process will search developed ontologies to refine and extend the new ontology. Finally, the structure of the new ontology will be refined as well. Bernaras et al's approach provides a way of building an ontology by reusing a developed ontology, but it still lacks methods to evaluate the ontology.

SENSUS method

The SENSUS method (Swartout et al, 1997) builds the skeleton of a domain ontology. As introduced in section 2.2.3, SENSUS concentrates on the domain of machine translation. It presents a method to link domain specific terms to a large-scale ontology and prune the tree in the large-scale ontology. The process to obtain the ontological skeleton is shown in Figure 3.1. All the steps need to be done manually and some new nodes also need to be added if the ontology developer requires some understanding of the domain. One problem raised here is perhaps the complexity of classifying whether the nodes in all of the sub-trees are useful or not, especially when the process is manually undertaken.

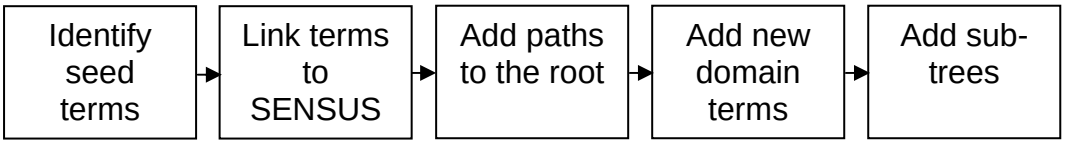


Figure 3.1: The process of building ontological skeleton in SENSUS

On-To-Knowledge methodology

The On-To-Knowledge methodology (Staab et al, 2001) has a group of techniques and methods to develop ontologies in order to improve the quality of knowledge management in distributed organizations. It firstly presents a feasibility study to make the adopted terms appropriate to the completed application, and then based on this feasibility study, executes a kickoff program to describe the competency questions. Finally it refines the terms to obtain the application oriented “target ontology”. As a knowledge engineering methodology, On-To-Knowledge also provides a method to learn, evaluate and maintain ontologies.

Thus, the ontology development methods and methodologies above cover almost all existing approaches. It is hard to compare the value of each because they all work

effectively for different objectives. They provide suitable guidelines for ontology development at different stages of the whole development process. However, most of the approaches focus only on development activities and ignore the other important respects such as evaluation, integration, merging and learning ontologies. The BGO development requires a more comprehensive approach that can provide a tool or tool suite that is able to provide technical support to cover all the activities necessary in the ontology engineering process. Therefore, the developing BGO employed constructive methodology, which takes advantage of Protégé to execute the development process.

3.2 A Constructive approach

The constructive methodology was firstly addressed by Nunamaker et al in 1991, and was concerned with conducting information system research that incorporated theory building, system development, experimentation and observation (see **Figure 3.2**). In the BGO development project, a constructive approach can take advantage of the selected development tool: Protégé. According to Cornford and Smithson (1996, page 44), constructive methodology is an information system development methodology, suitable for building a research framework, undertaking technical development and refining domain concepts. It "*does not describe any existing reality, but rather helps to create a new one, and does not necessarily have any 'physical' realization.*" The development of the BGO is new and important in its field, and it cannot be proven mathematically and evaluated empirically although there are some previous ontology cases. So the best way is to construct a new method under the theoretical framework of constructive methodology.

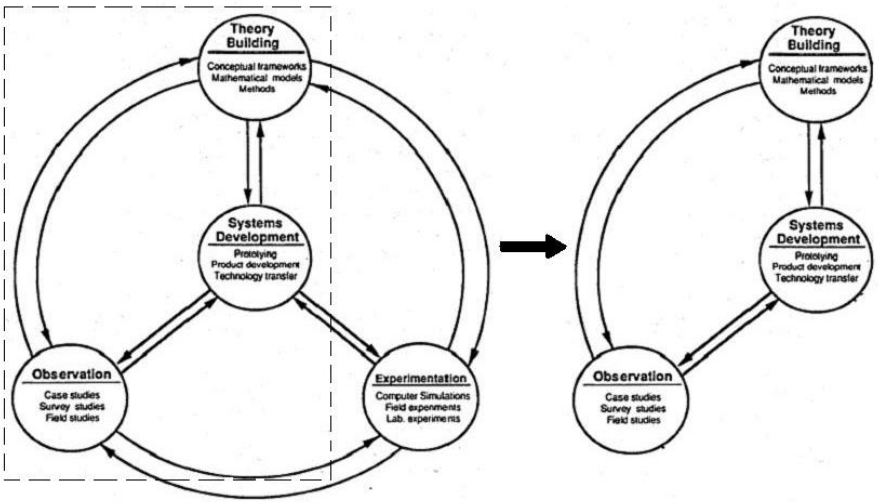


Figure 3.2: A multimethodological approach to IS research (Nunamaker et al, 1991)

Based on Nunamaker and his colleagues' multi-methodological approach, this research was executed through the following five basic phases not including experimentation:

1. **Construct a conceptual framework**: In this phase the research questions were justified. After broad literature research and analysis, the proposed solution was specified as building BGO by using Protégé. As a result, the technical features and advantages of Protégé were reviewed in order to demonstrate the validity of the solution. This phase included the investigation and the understanding of Protégé functions, additional plug-ins, BGO requirements and building procedures etc. Some other disciplines related to additional approaches were also explored.
2. **Build architecture**: This provided the road map for the BGO development process. The architecture specified the employed functions of Protégé components and the relationships among the components. It also considered the extensibility and modularity of Protégé.

3. **Analyze and design the system**: This stage was mainly for brain gene domain studies, diseases analysis and database creation. The domain studies included the application of the related scientific and technical knowledge. As an ontological engineering project, the data structures and databases were determined at this stage. The final solution was selected from the alternatives.
4. **Develop the system prototype**: This phase was building the BGO and gain insight as to the feasibility and usability of the BGO.
5. **Observe and evaluate the system**: Depending on the statement regarding the requirements of the previous phase, the developed BGO was tested in order to measure its performance, and at the same time, the impacts that each operation brought to various terms were observed. Finally, the evaluation for the whole BGO provided based on the conceptual framework.

As Figure 3.2 shows, the experimentation part is not involved in the research framework because this Brain Gene Ontology Engineering case mainly focuses on the BGO design, implementation and observation as stated above, and experimentation will be a necessary component of the research framework in further research. This approach basically covers the efforts of previous research and integrates a comprehensive and flexible tool to resolve technical gaps. The next main issue is the criteria used in the BGO development process.

3.3 Criteria

The criteria in this section were used for research restrictions, the performance measurements of the BGO and design problem identification.

3.3.1 The Criteria for Determining Scope

Nunamaker et al (1991) emphasize that every researcher of IS engineering firstly needs to determine and limit the scope of their research. This is also very important in the ontology engineering process. On the one hand, ontology is a way of modeling reality, but this does not mean it includes everything in the domain. In the process of building the BGO, the following questions were used to guide many of the modeling decisions down the road:

1. What is the knowledge that BGO will cover?
2. What is the BGO going to be used for?
3. What types of questions should the BGO provide answers for?
4. Who will use and maintain the BGO?

These questions help limit the scope of the BGO. The simple answers for these questions are: 1.) BGO represents information about brain genes, proteins, diseases and the relationships between them. 2.) BGO is used to help the study of brain genes and diseases, emphasizing which genes are highly expressed in a brain disease. 3.) The concepts of BGO describe the functions, the molecular length, weight and important annotations of significant genes. 4.) Biomedicine students and researchers are the potential users and maintainers of the BGO.

The BGO should provide enough information and certain specific levels of detailed information about brain genes, proteins and diseases. A **Competency questions** (Gruninger and Fox, 1995) list was made to describe the information that should be included in the BGO:

- What is the name of Gene/protein related to brain disease?
- Is that a protein complex?

- What is the chromosomal location of gene?
- What is the molecular length of gene/ protein?
- What is the molecular weight of gene/ protein?
- Which protein does this gene produce?
- What are the reference articles for this gene/protein in PubMed?
- What is the function of the gene/protein
- Are there any other comments?
- How is the gene expressed in its expression map?
- What is the orientation of protein/gene?
- What is the source of the gene/protein? (Human/animal)

These questions are not comprehensive, but are useful in representing some typical information in the BGO. Through answering these competency questions, BGO would include knowledge that biomedicine students and researchers need.

3.3.2 The Criteria for Ontology design

The design of BGO obeyed the basic principles that have been proven valuable in previous ontology development. These design principles can also be seen as the objective criteria for guiding ontology design and are explained as follows:

The standardization of names (Arpirez et al, 1998): Besides the naming rules in Protégé, the naming conventions of BGO are the same for all the name related terms in order to make the BGO easily understood. For example: *EO_Creator_Scheme* and *EO_Annotation_Scheme* follow the same naming conventions; *EO_Creator_Scheme* and *Annotation_SchemeInEO* do not.

Minimize the syntactic distance between sibling concepts (Arpirez et al, 1998): The creation of sibling concepts in BGO is based on the same pattern, the representations of these sibling concepts use the same primitives if possible. This improves the comprehensibility and reusability of the BGO.

The representation of disjointed and exhaustive knowledge (Arpirez et al, 1998): Generally, subclasses are disjointed if they do not have any common instances. Two subclasses in BGO are defined as disjointed decomposition when using the system in Protégé if they do not have common instances.

Clarity (Gruber, 1993b): All the definitions (classes) in BGO are objective. They are stated and documented with natural language in BGO.

Extendibility (Gruber, 1993b): Future users of the BGO are able to define new terms for some extended usage, for example, the four classes in the red box can be created for the introduction of the BGO and related information storage (see **Figure 3.3.2**). This extended content is based on existing vocabulary and did not require revision of existing definitions.

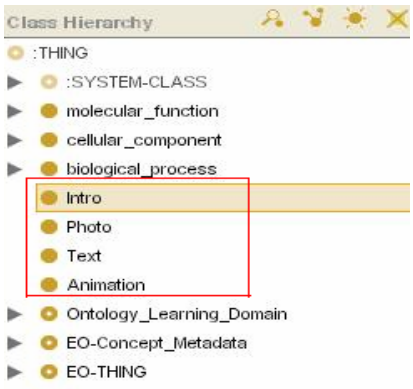


Figure 3.3.2: Extended classes

Coherence (Gruber, 1993b): The slots and facets of the definitions in BGO are limited to a reasonable range based on the axioms. Anything that causes a definition or instance given contradicts the axioms has been fixed.

Minimal ontological commitments (Gruber, 1993a): To support knowledge sharing, BGO specifies the weakest theory and defines only the terms that are crucial to the communication of knowledge consistent with the theory. This minimizes ontological commitments and increases the reusability of the definitions of the BGO in different systems.

3.3.3 The Criteria for Reusing the Existing Sources

As is stated earlier, there are two ways to build an ontology, by reusing a developed ontology or from scratch. Noy and McGuinness (2001) emphasized that it is always worth considering refining or extending existing ontologies that have been created by others with a similar purpose or in the same domain. The biomedical field has produced some large standardized and structured vocabularies. This brain gene ontology case imports and reuses some structures and information from GO (Gene Ontology) (Ashburner et al, 2000) and UMLS (Unified Medical Language System) (Humphreys and Lindberg, 1993). Information about specific genes and proteins was also extracted from well-known bioinformatical databases. Some criteria were defined to guide the importing process:

Comparative selection: The knowledge model of BGO took experiences from both GO and UMLS. Through importing some of the class hierarchies with attributes from these classical domain ontologies, BGO gained a first pass of the classification of genetic knowledge and feature selection. The selected structures or knowledge trees were compared before merging them in order to improve their suitability and accuracy. Almost all of the data has multiple sources from different genetic databases. These

were marked and linked to the annotation part of BGO. The different sources on the same unit were also recorded in the BGO.

The adoption of Plug-ins: Importing and comparison were done by Protégé plug-ins. There are two plug-ins in the importing process: PROMOPT Tab and UMLSTab. UMLSTab was adopted to access the UMLS knowledge base and PROMOPT Tab was used to compare and merge the imported parts with BGO and track any changes that happened in the whole process.

Database evaluation: A list of biological sources covering the whole domain was evaluated by domain experts. The brain gene data and information were collected from large-scale, stable and credible databases. There is also an annotation feature in BGO where domain experts can share and annotate information in the field. Each piece of data was included from more than one source. Different results were stored as different categories by source.

3.3.4 Criteria for BGO Structure Evaluation and Biological Data Verification

Domain experts (Vishal Jain and Lubica Benuskova) supervised the BGO structure evaluation and biological data verification in this research. There are two reasons why domain experts are required to supervise these aspects:

1. **The knowledge limitation of an IS developer:** Although ontology engineering case study is IS research, it requires the BGO developer to have a deep understanding of the structure of biological knowledge and vast experience of biological data investigation, in which an ordinary IS researcher

may not be competent. Domain experts and supervisors were also the initial users of the BGO and the cooperators for BGO development.

2. **The importance of data resource verification and evaluation work:** There are a huge numbers of biological and bioinformatical resources that are available from the World Wide Web. However, not all of them are credible and comprehensive. These resources and data need to be verified by biological experts before being adopted. Moreover, the potential users of the BGO are biological students and the knowledge base for biological researchers. The knowledge content and structure must reach a high standard of accuracy and authority.

3.4 Building Architecture

Figure 3.4 shows the entire research framework. In the whole process of BGO development, domain experts provided basic ideas on the requirements for ontology development and reflected on the problems that the ontology had from the point of view of biology. At the same time, the ontology development started to build the architecture with the blocks of literature research, methodology analysis and biological knowledge resources were being developed. Excluding the preparation phase, BGO development involved four basic stages: building the conceptual framework, designing the BGO system, BGO assessment and BGO system refinement.

The building of the conceptual framework has been stated in detail in section 3.2. This section focuses on the ontology development process and describes the whole process including ontology design; materials, techniques, samples and data that were used in the BGO development.

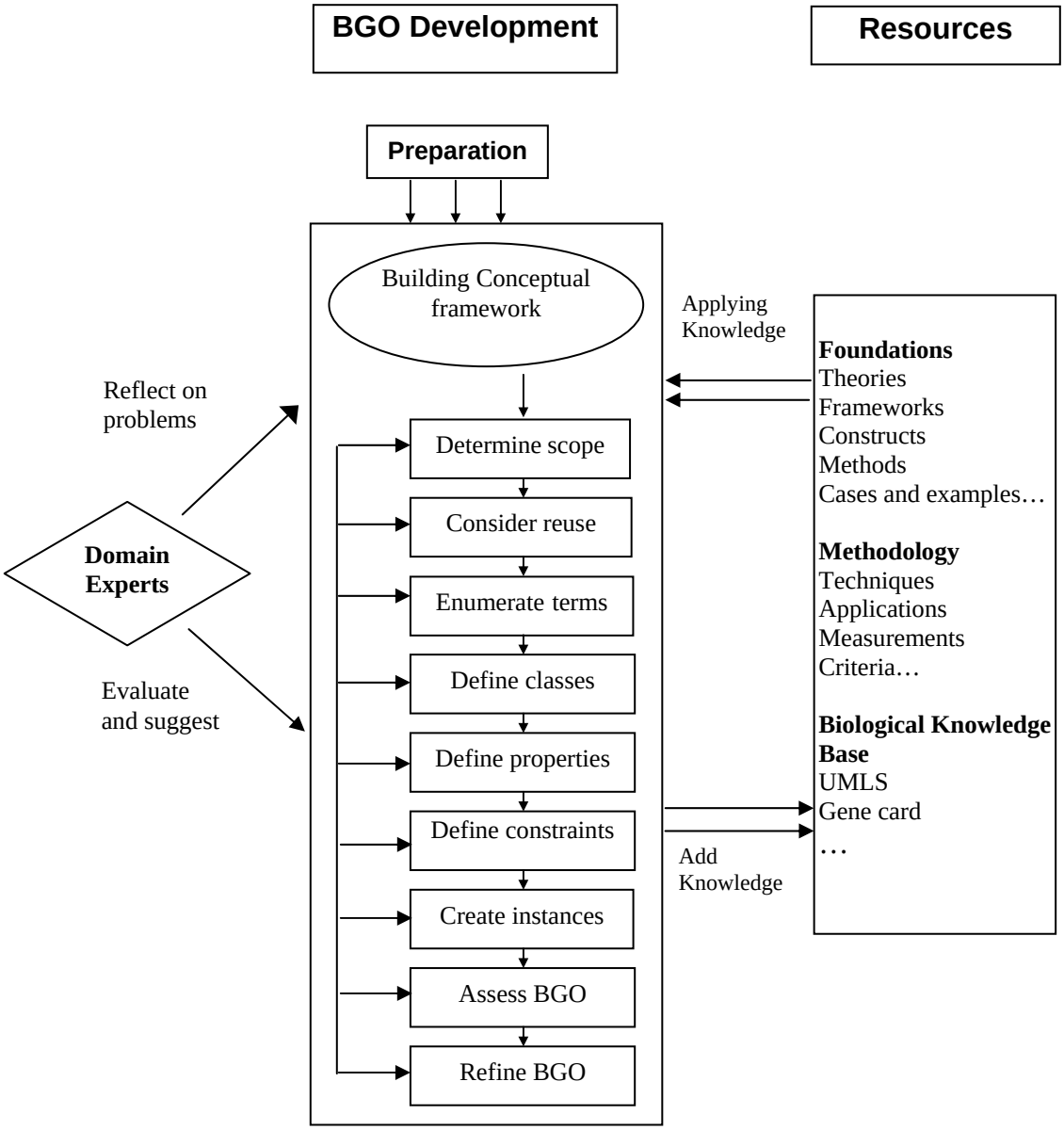


Figure 3.4: The conceptual framework for BGO development

3.4.1 Preparation

To follow the normal research procedure sequence, the preparation phase was organized into the following steps:

Step 1. Determine the research scope and the research topic.

Step 2. Decide on the research questions and the expected research outcomes.

Step 3. Review the literature related to work done by other researchers in the field in order to identify their achievements and drawbacks.

Step 4. Select the techniques and platforms for BGO development based on the previous step.

Step 5. Decide on the research methodologies and confirm the plan with the supervisor.

Step 6. Analyze, make, evaluate and update the criteria.

Step 8. Write a plan for the practical phase and identify the contribution of each stage.

Step 9. Discuss and review the outcomes of the preparation phase with the supervisor.

3.4.2 BGO Development Environment and Data

Techniques and Platform: A stable version of Protégé (version 3.1) was adopted as the BGO development platform (See section 2.4). There were three adopted Protégé plug-ins in the development process in total: UMLSTab, PROMPT, and TGVizTab. PROMPT was mainly used to update, compare and merge ontologies; the UMLSTab was used to import and reuse; the TGVizTab was for ontology visualization and structure handling (See section 2.4.4).

Biological Data: Through a series of web searches with the supervisor and domain experts' recommendations, the candidate biological knowledge bases were selected (shown in **Table 3.4.2**).

Data regarding 93 significant brain genes, 70 proteins and 14 protein complexes with a number of attributes were collected from these online knowledge bases. Table 3.4.2.2 shows a part of the gene information gathered for the subunits of the AMPAR

(amino-methylisoxazole-propionic acid) receptor. The data collected from different sources were compared, identified and stored with annotations.

Name	*URL Address*	Type of the Knowledgebase
Gene Card	*http://www.genecards.org/*	An integrated database of human genes
Genbank	*http://www...Genbank/index.html*	An NIH genetic sequence database
Genes and Disease	*http://www.ncb...=gnd.chapter.75*	A collection of articles that discuss genes and the diseases that they are connected to.
KEGG	*http://www.genome.jp/kegg/*	An Encyclopedia of genes and genomes
Allen Brain Atlas	*http://www.brain-map.org/*	An interactive, genome-wide image database of gene expression in the mouse brain
Swiss-Prot	*http://us.expasy.org/sprot/*	A curated protein sequence database
Ensembl	*http://www.ensembl.org/index.html*	A project which produces and maintains automatic annotation on eukaryotic genomes
Gene Ontology	*http://www.geneontology.org/*	A controlled vocabulary to describe gene and gene product attributes in any organism
GNF	*http://expression.gnf...bin/index.cgi*	An RNA expression database
GeneLoc	*http://bioinfo2.weizmann...index.shtml*	An integrated map for each human chromosome
GeneNote	*http://bioinfo2.wei.../home_page.pl*	A database of human genes and their expression profiles in healthy tissue

Table 3.4.2.1: Candidate biological knowledge bases for data collection

Protein Name	Gene Name	Function or Comments	Molecular Weight; Length	References at NCBI (if available)
Glutamate receptor 1	GRIA1	L-glutamate acts as an excitatory neurotransmitter at many synapses in the central nervous system. The postsynaptic actions of Glu are mediated by a variety of receptors that are named according to their selective agonists.	101536 Da; 906 AA	PubMed=1311100 [NCBI] PubMed=1320959 [NCBI] PubMed=1652753 [NCBI]
Glutamate receptor 2	GRIA2	Receptor for glutamate. L-glutamate acts as an excitatory neurotransmitter at many synapses in the central nervous system. The postsynaptic actions of Glu are mediated by a variety of receptors that are named according to their selective agonists. This receptor binds AMPA(quisqualate) > glutamate > kainate. Interacts with PRKCABP, GRIP1 and GRIP2 (By similarity).	98821 Da; 883 AA	PubMed=8003671 [NCBI] PubMed=12477932 [NCBI] PubMed=7523595 [NCBI]
Glutamate receptor 3	GRIA3	Receptor for glutamate. L-glutamate acts as an excitatory neurotransmitter at many synapses in the central nervous system. The postsynaptic actions of GLU are mediated by a variety of receptors that are named according to their selective agonists. Interacts with PRKCABP, GRIP1 and GRIP2 (By similarity).	101023 Da; 894 AA	PubMed=7918660 [NCBI] PubMed=10602120 [NCBI]
Glutamate receptor 4	GRIA4	L-glutamate acts as an excitatory neurotransmitter at many synapses in the central nervous system. The postsynaptic actions of Glu are mediated by a variety of receptors that are named according to their selective agonists.	100809 Da; 902 AA	PubMed=8589990 [NCBI]

Table 3.4.2.2: Part of the gene table for the subunits of the AMPAR (amino-methylisoxazole-propionic acid) receptor

3.4.3 BGO Development Process

Considering the features of the Protégé ontology editor and its possible functional plug-ins, one of the best solutions to building a brain gene ontology is to begin with a rough first pass of the ontology, and then assess and refine the evolving ontology, finally inserting the data and details.

Therefore, BGO development in Protégé is actually an iterative process. **Figure** shows the possible ontology development process with Protégé. Theoretically, the

most direct way to build the BGO has to involve the basic elements in **Figure** and follow the steps below:

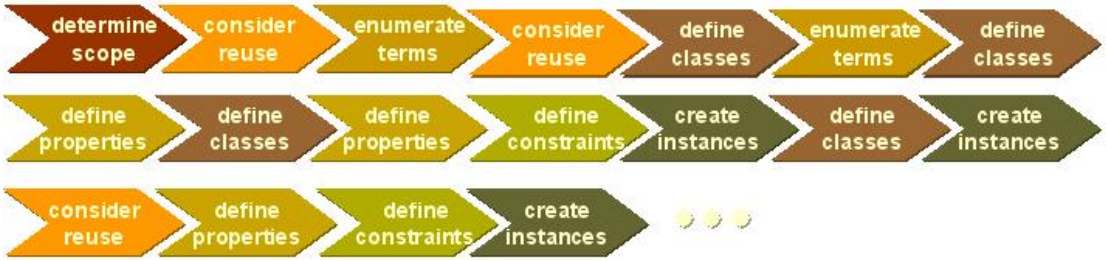


Figure 3.4.3.1: The iterative ontology building process in Protégé (Noy & McGuinness, 2001)

Step 1. Determine the scope of the BGO: As discussed in section 3.3.1, the knowledge that BGO covers was restricted to a reasonable range and while still being able to provide the necessary level of detail on brain genes, proteins and the relationships between them.

Step 2. Consider reusing existing ontologies: There are two biological knowledge structures that were able to be used as a reference by the BGO; UMLS and Gene Ontology (see section 2.2.4). Gene Ontology has XML and OWL versions which are both supported by Protégé. In order to import the terms from these two knowledge bases into Protégé, The plug-in: UMLSTab was used. **UMLSTab** is able to browse and import the knowledge tree from the UMLS knowledge base. (See section 2.4.4)

Step 3. Enumerate important terms in the ontology: This step requires a list to remind the developer which terms are necessary to the BGO. These terms include the basic domain concepts such as: gene, protein, protein complex etc， and the properties

of the concepts such as gene function, gene expression map, molecular length, molecular weight etc.

Step 4. Define classes and class hierarchy: The BGO adopts the top-down method **(Uschold & Gruninger, 1996).** This method starts building the class hierarchy with the most general concepts in the domain, and then creates and categorizes the sub-concepts of the concepts. For example: BGO requires a class to represent brain gene annotations and the class "EO_Annotation_Scheme" was created, then further categorized the classes into subclass "GO_Annotation" to represent the Annotations in Gene Ontology and "EO_External_Database" to represent the Annotations in other knowledge bases. **(Figure 3.4.3.2)**

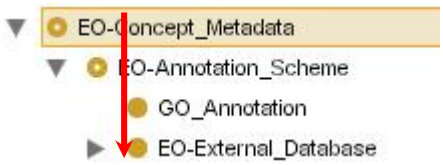


Figure 3.4.3.2: Top-down method

Step 5. Define the properties of classes—slots: Slots can be created with Protégé editors. They represent the properties of the domain concepts. After completing the class hierarchy, slots were created to describe the internal structures of the BGO classes. There are various types of slots. For example: molecular length is an"intrinsic" property, name is an"extrinsic" property and organism represents the relationship between "gene" and "organism" etc. The properties of the class can be inherited by its subclasses.

Step 6. Define the constraints of the slots: Facets restrict a series of features such as: value type, value range and the number of values etc that a slot value can have. For example, the orientation of a gene can only be "plus strand" or "minus strand", so the

value type of the slot “orientation” is symbol and the two values are the only allowable values.

Step 7. Create instances: The instances represent the individual cases in the created classes. Individuals of the BGO classes were created last. This requires specifying a class, creating an instance for the class and filling in the slots of the instance.

Step 8. Assess the BGO: The first version of the BGO was rough and easily made errors. This research adopted two methods to find the problem. One was the suggestions and feedback from the domain experts and supervisors as mentioned in section 3.3.4. As well as this, this research integrated the ideas of Noy and McGuinness (2001) to provide a guideline to check and advise the ontologies built with Protégé. Some of the rules were adopted for BGO assessment and listed as follows:

- All the instances of a subclass are also the instances of the parent classes.
- All the subclasses of a subclass are also the subclasses of the parent classes.
- One of the most common errors is a class cycle: A is the subclass of B and B is the subclass of A.
- Normally, the amount of the subclasses of a class should be a natural number that is greater than 1 and smaller than twelve. Otherwise, the ontology is not complete and some intermediate categories need to be added.
- A class can be the subclass of several classes, which also means it inherits all of the slots from all these classes.
- New classes should be introduced when an ontology has a very flat structure; a few classes and a large number of information in each slot; classes should be

reduced when an ontology has an extremely nested hierarchy with many extraneous classes.

- Instances must be the most specific concepts. Otherwise they should be a class.
- Inverse slots decide each other's value. For example, if Gene A produces protein B, then protein B is produced by Gene A. From the knowledge-acquisition perspective, this is not redundant, and both pieces of information should be explicitly available.

Step 9. Refine BGO: Once any problem has been found, go back to the step where it was created and fix it.

Chapter 4 – Brain Gene Ontology

The Brain Gene Ontology version as of June 14, 2006 contains 286 concepts, stores information about 93 brain genes, 90 proteins, 14 protein complexes and 228 related references from PubMed. Significant neuronal parameters such as AMPA, GABA, NMDA, SCN, KCN and CLC were created with detailed descriptions. In addition, their direct or indirect interactions with several other genes/proteins, and their expression levels were also indicated. **Figure 4.0.1** shows the ClassesTab of the BGO in Protégé-Frames. The class hierarchy can be seen clearly from the CLASS BROWSER on the left. The class: THING is the root class and the parent class of all of the classes in the BGO and SYSTEM_CLASS is used by Protégé-Frames for defining the structure of various Protégé forms. Several classes were created for BGO learnings and presentations including the classes, Intro, Photo, Text and Ontology_learning_domain. The rest of the classes describe the roles of brain gene products and the relationships between the most important terms with annotations.

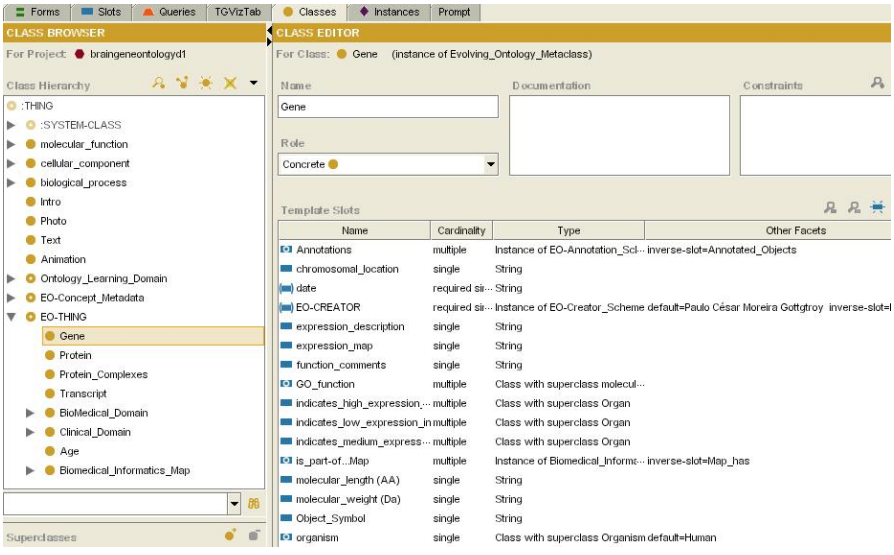


Figure 4.0.1: BGO in the Classes Tab

The CLASS EDITOR on the right shows the information in the class and the template slots that the selected class has. The slots can be viewed and edited by double clicking in CLASS EDITOR without going to the Slots Tab. All individuals including the brain genes, proteins and annotations etc can be queried from both the Queries Tab and the Instances Tab by entering their symbols or any slots value that the queried individual has. A detailed description of each term can be seen in the sample script in **Figure 4.0.2** which is taken from the XML version of a BGO file.

```
<class>
        <name>Gene</name>
        <type>Evolving_Ontology_Metaclass</type>
        <own_slot_value>
                <slot_reference>EO-SOURCE</slot_reference>
                <value value_type="simple_instance">infogenev1_00072</value>
        </own_slot_value>
        <own_slot_value>
                <slot_reference>EO-CREATOR</slot_reference>
                <value value_type="simple_instance">evolvingontology2_Instance_8</value>
        </own_slot_value>
        <own_slot_value>
                <slot_reference>EO-DATE_CREATED</slot_reference>
                <value value_type="simple_instance">BGOmap_Instance_60000</value>
        </own_slot_value>
        <own_slot_value>
                <slot_reference>:ROLE</slot_reference>
                <value value_type="string">Concrete</value>
        </own_slot_value>
        <superclass>EO-THING</superclass>
        <template_slot>is_part-of...Map</template_slot>
        <template_slot>Object_Symbol</template_slot>
        <template_slot>Synonyms</template_slot>
        <template_slot>indicates_high_expression_in</template_slot>
        <template_slot>organism</template_slot>
        <template_slot>chromosomal_location</template_slot>
        <template_slot>GO_function</template_slot>
```

<template_slot>Orientation</template_slot>

<template_slot>Annotations</template_slot>

<template_slot>produce</template_slot>

<template_slot>indicates_low_expression_in</template_slot>

<template_slot>expression_description</template_slot>

<template_slot>function_comments</template_slot>

<template_slot>indicates_medium_expression_in</template_slot>

<template_slot>expression_map</template_slot>

<template_slot>molecular_length (AA)</template_slot>

<template_slot>molecular_weight (Da)</template_slot>

<template_facet_value>

<slot_reference>is_part-of...Map</slot_reference>

<facet_reference>:VALUE-TYPE</facet_reference>

<value value_type="string">Instance</value>

<value value_type="class">Biomedical_Informatics_Map</value>

</template_facet_value>

<template_facet_value>

<slot_reference>organism</slot_reference>

<facet_reference>:DEFAULTS</facet_reference>

<value value_type="class">Human</value>

</template_facet_value>

<template_facet_value>

<slot_reference>organism</slot_reference>

<facet_reference>:VALUE-TYPE</facet_reference>

<value value_type="string">Class</value>

<value value_type="class">Organism</value>

</template_facet_value>

<template_facet_value>

<slot_reference>GO_function</slot_reference>

<facet_reference>:VALUE-TYPE</facet_reference>

<value value_type="string">Class</value>

<value value_type="class">molecular_function</value>

</template_facet_value>

<template_facet_value>

<slot_reference>Annotations</slot_reference>

<facet_reference>:VALUE-TYPE</facet_reference>

<value value_type="string">Instance</value>

<value value_type="class">EO-Annotation_Scheme</value>

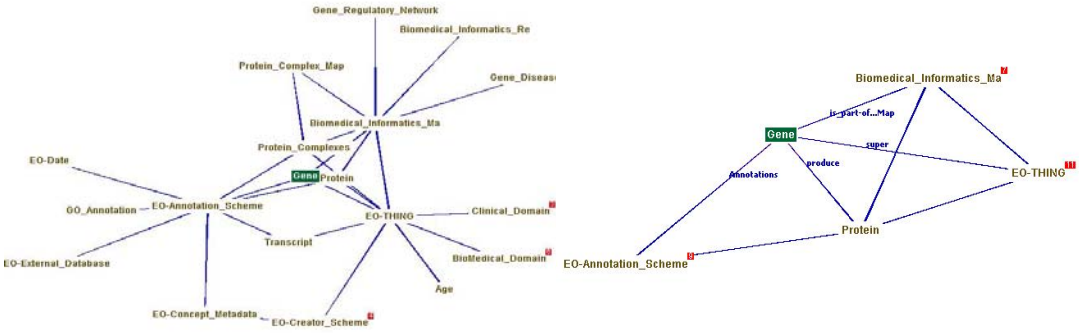
```
                </template_facet_value>
                <template_facet_value>
                        <slot_reference>produce</slot_reference>
                        <facet_reference>:VALUE-TYPE</facet_reference>
                        <value value_type="string">Instance</value>
                        <value value_type="class">Protein</value>
                </template_facet_value>
</class>
```

Figure 4.0.2: A detailed script of a created Class: Gene (XML version)

The creation of the gene class includes identifying different types of part–whole relationships with different slots, and defining slots and facets for slots. The following sections focuses on how BGO works under the extended functional APIs and enumerates the Widget types in BGO and their different applicability.

4.1 BGO visualizations with TGVizTab

Sometimes, the relations and interactions among the terms in BGO can be very complex. There are methods to view and navigate such terms and relations more clearly and effectively in Protégé-Frames. **Figure 4.1A and 4.1B** show the visualization maps for the class: Gene with TGVizTab. Through map A, one can see the closest terms of Gene in the first two levels of the whole BGO hierarchy, and through the enlarged map B, one can navigate the exact relations among Gene and other terms. For example: Gene produces Protein; Gene is the super class of EO-THING; Gene is in part of the Biomedical_Informatics_Map; Gene is annotated by EO-Annotation_Scheme etc.

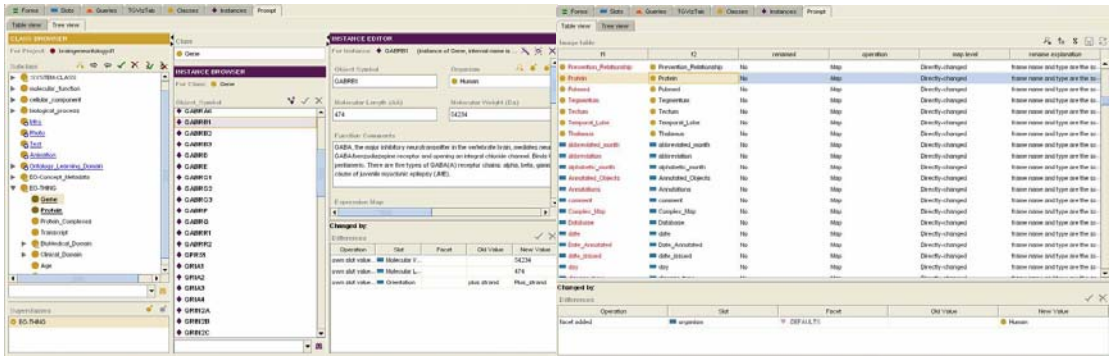


(A) **(B)**

Figure 4.1: Gene in visualization maps.
(A) The closest terms in the first two hierarchies. (B) Navigating exact relations.

4.2 Tracking changes in the BGO in PROMPT

The engineering process of BGO is endless and requires improvements and maintenance by various developers and users. Therefore it must have a function to track changes in order to know where the changes are, who made the changes and what they changed. As can be seen in **Figure 4.2A and 4.2B**, the plug-in, PROMPT, provides two kinds of forms to compare BGO versions from June 14, 2006 and June 2, 2006.



(A) **(B)**

Figure 4.2: Tracking changes in BGO with PROMPT.
(A) Tree view, (B) Table view

The tree view form follows the visualized style of the BGO class hierarchy faithfully. One can find details of the changes in class, slots and instances easily as they have been marked. All the differences in the selected term are shown on the bottom right. The table view form provides a list of which terms in BGO have been changed and what kind of changes they are, and shows the differences in the term at the bottom.

4.3 The storage and display of slots in the BGO

In BGO, almost every class has a number of attributes that are very important for brain gene research. **Figure 4.3** shows the attributes panel of the brain gene: GABRA1. These attributes have various value types and are displayed in different forms of widget in the Instances Tab (See section 2.4.1).

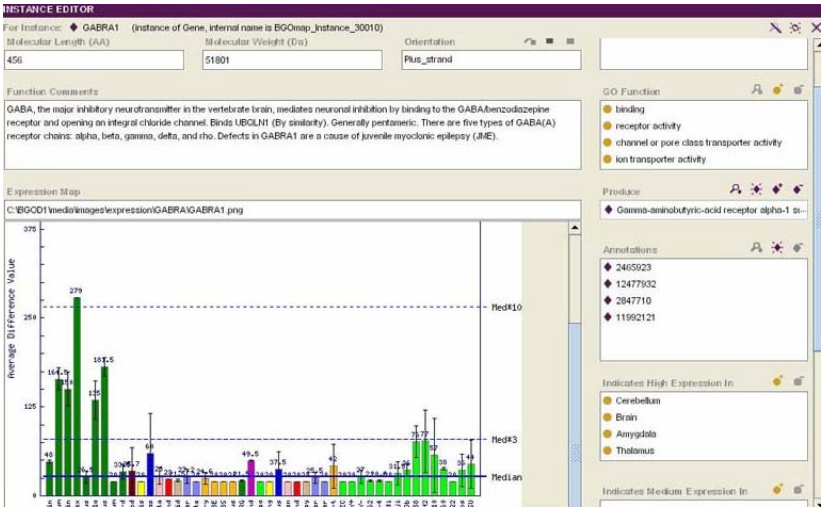


Figure 4.3: The various types of slot widgets

There is no restrictive rule for choosing widget types for the slots. Nevertheless, the value type, value cardinality and the capacities of the widgets should be considered. The gene slots can be examples to prove the conditions for which widgets are suitable.

TextFieldWidget: Chromosomal Location, Molecular length and Weight, Object Symbol

Chromosomal Location, Molecular length and Weight are the internal attributes of brain genes and the values for them are all short numeric. Object symbol is the symbol name of brain genes in biology. Their values can all be short strings, and normally they only have a single value.

StringListWidget: Synonyms

One gene may have multiple synonyms identified in different knowledge bases. For example, GRIA1 has the Synonyms: GLUR1 and GLUH1 in SWISS-PROT. The value type is short strings.

SymbolListWidget: Orientation

The orientation of a brain gene has only two possible values: PLUS or MINUS. The value type is short strings.

TextAreaWidget: Function comments and Expression descriptions

The function comment and the expression description of a brain gene are all paragraphs of descriptive words. The value type is long strings.

ClsFieldWidget: Organism

A gene may belong to multiple organisms, but the current version of BGO only supports brain genes in human organisms. Human is a class.

ClassListWidget: GO Functions, Indicates High, Medium and Low Expression In

A brain gene may have more than one function comment in GO. The same gene can indicate different levels of expression in multiple organs of the brain. The value type is “classes”.

ImageWidget: Expression Map

Images of the gene expression map can only be displayed in BGO with ImageWidget. Other multimedia files, such as videos and audios, are displayed as URL links.

InstanceFieldWidget: Produce, Creators and Is Part-of...Map

A brain gene can produce a protein which is an instance of the class Protein, and is part of a protein complex map which is the also an instance of the class Protein_Complex_Map. The creator of this gene is an instance of the class Ontology_Engineer.

InstanceListWidget: Annotations

Every annotation in BGO has a unique ID number and the annotations of genes are instances of the class: EO-Annotation_Scheme. Since the annotations are displayed in widgets as ID numbers, their value type is short strings.

Chapter 5 - Result and Discussion

The Brain Gene Ontology development case study is implemented to answer the questions is Information System development methodology suitable for ontology development, how does Protégé-Frames and its extended functional APIs perform the BGO development process, and if the BGO system can integrate gene data from multiple biological knowledge bases with BGO? The present results are evidence that their research basically reached these targets and resolved the research problems.

Firstly, the BGO engineering case study was done under the guidance of the constructive approach. The conceptual BGO development framework followed fundamental IS research theory. The architecture of the research also gained benefits from the IS approach since it enhanced the interactions and relations among the ontological engineering process components. The ontology building, merging, learning and evaluation of the ontology were sufficiently combined together as a whole and this actually filled in the gaps of previous research. All of the above indicates that IS methodology can properly be applied to biomedical research and is capable of conceptual development and implementation.

Secondly, Protégé-Frames was well utilized in the whole ontology building process. The main functional tabs tightened the relationships among the classes, class hierarchy, slots and instances. The developed BGO is able to describe the role of significant brain genes, proteins and their relationships. The extended plug-ins were helpful in improving BGO visualization. The visualization plug-ins helped to enhance the presentation of the relationships among BGO terms and reduced the cognitive gap between ontology users and developers. To increase the reusability of the existing resources and the efficiency of this reuse, the knowledge-acquiring plug-ins were

implemented. This brought to the BGO not only ideas for constructing an ontology, but also the information under the expert system supervised. To reinforce the functions of the ontology editor, plug-ins for tracking changes, ontology merger and comparison were employed. This dramatically enhanced the performance of ontology merger, modification and update. Therefore, the selected tool suite is proven to be suitable and flexible enough for the research task. The open strategy of BGO is also based on functional support from Protégé.

Finally, the research extracted the information from several well-known biomedical knowledge bases to build its own knowledge base. BGO gathered genomic data and protein annotations from many sources. The most important issues at this stage are data verification and database evaluation. Since the research received supervision from domain experts, it gained significantly in the area of criteria making, these issues were resolved smoothly.

Chapter 6 Conclusion

6.1 Conclusion

This ontology engineering research represents brain gene and disease information by adopting a constructive approach, improving discovery, sharing and reuse of information about crucial neuronal parameters through their interactions with each other. The constructive approach takes advantage of the IS development methodology. It builds a comprehensive development framework to enhance the process of ontology learning, design, evaluation and refinement so that the adopted tool, Protégé-Frames can be efficiently used. To compare the BGO development process with previous research, the IS development methodology provides more ideas to improve the communications between the ontology developers and the domain users.

In addition, the implementation of an open source extended application provided more flexibility and reusability for ontological visualizations and refinements. The multiple tools are applied to the development process. This actually brings benefits for both the developing ontology and the developed Brain Gene Ontology. Furthermore, the BGO annotation system is based on multiple notations from different biomedical knowledge base. It significantly increases the credibility of the BGO information and improves the accuracy of the biological data in the system.

A difficulty in BGO development is always making criteria because there is no single rule to cover all the issues in ontology development. Assumptions on class hierarchy design, knowledge reuse and verification must have the support of detailed criteria. The criteria making process of this research was based on theoretical foundations and

the experience of bioinformatical experts who were also in the BGO project team. All the steps were evaluated by them, which ensured the developing process was on the right track.

This BGO development case is also a foundation stone for further Kedri project: Brain Gene and Simulation System (BGOS) (Kasabov et al, 2007a), which represents stronger integration of Bioinformatics and Neuroinformatics Data in order to improve the understanding of Brain (Kasabov et al, 2007b).

6.2 Future research

The research results indicate that the BGO engineering case study significantly improved knowledge representation of the brain genes. Future research on brain gene ontology is proposed as follows:

1. Ontology Visualizations: Although there are already some opportunities to visually represent the relationships and interactions between important terms in BGO, the visualization of BGO is unfinished. Further work is detailed in another paper (Wang et al, 2006). Further improvements in the visualized plug-in were introduced. Hyper graph techniques such as Large Graphical Layout, Walrus and Hypergraph were presented and discussed as new and plausible techniques for ontology visualization and their application in the BGO project.
2. Ontology driven knowledge discovery: Gottgtroy et al (2004) demonstrated a theory, which combines ontology and machine learning techniques and will be applied to the BGO system. The core of “Ontology driven knowledge discovery” is Onto4KDD4Onto. On the one hand, the adoption of the ontology

application can be used to improve the KDD process. On the other hand, the mining technologies of KDD can also be employed to learn and build an ontology. Based on this theory, Wang et al (2006) worked on a method for mapping interesting ontology knowledge, by means of visualization, into a format able to be further analyzed by data mining workbenches, with the possibility of importing knowledge generated by machine learning algorithms into ontology representation and visualization.

3. Evolving Ontology: Further research on the evolving ontology may integrate more information and annotations from more knowledge bases dealing with brain disorders and other brain diseases in order to enhance the understanding of the relationships between the genes and diseases.

References:

Alani, H. (2003) *TGVizTab: An Ontology Visualisation Extension for Protégé*. Proceedings of the Knowledge Capture (K-Cap'03), Workshop on Visualization Information in Knowledge Engineering, Sanibel Island, Florida, USA.

Alphaworks. (2006). *What is ontology? Frequently asked questions.* Retrived from http://www.alphaworks.ibm.com/contentnr/semanticsfaqs?open&S_TACT=106AH21W&S_CMP=AWLP in March, 2007, IBM.

Apweiler, R. (2001). *Functional information in SWISSPROT: The basis for large-scale characterisation of protein sequences.* HENRY STEWART PUBLICATIONS 1467-5463. BRIEFINGS IN BIOINFORMATICS. vol 2. NO1: 9-18.

Arpirez, J.C., Corcho, O., Fernndez-Lopez, M., Gomez-Perez, A. (2003). *WebODE in a nutshell*. AI Magazine 24(3):37-48.

Ashburner, M and 19 colleagues in the Gene Ontology Consortium. (2000). *Gene ontology: tool for the unification of biology*. Nature Genet, vol 25, page: 25–29. http://www.nature.com/ng/journal/v25/n1/full/ng0500_25.html

Aspirez, J., Gomez-Perez, A., Lozano, A and Pinto, S. (1998). *(onto)2agent: An ontology-based www broker to select ontologies.* In Proceedings of the Workshop on Applications of Ontologies and Problem-Solving Methods, ECAI'98, Brighton, England, pages 16-24.

Astrova, I. (2004). *Extracting Ontologies from Relational Databases*. Databases and Applications 2004: 56-61.

Bateman, J. A., Fabris, G and Magnini, B. (1995). *The generalized upper model knowledge base: Organization and use.* In Proceedings of KB-KS-95, Twente, The Netherlands. Second International Conference on Building and Sharing of Very Large-Scale Knowledge Bases.

Bernaras, A., Laresgoiti, I. and Corera, J. (1996). *Building and reusing ontologies for electrical network applications.* In: Wahlster, W. (ed) European Conference on Artificial Intelligence (ECAI'96). Budapest, Hungary. John Wiley and Sons, Chichester, United Kingdom, pp 298-302.

Bodenreider, O., Mitchell, J. A and McCray, A. T. (2003). *Biomedical ontologies*. Proceedings of the Pacific Symposium on Biocomputing, vol. 8, page: 562-564.

Bontas, E. P., Mochol, M. and Tolksdorf, R. (2005). *Case Studies on Ontology Reuse*. Proceedings of I-KNOW '05 Graz, Austria. http://i-know.know-center.tugraz.at/content/download/414/1619/file/Paslaru_paper.pdf

Borst, W. N. (1997). *Construction of Engineering Ontologies*. Centre for Telematica and Information Technology, University of Twente. Enschede, The Netherlands.

Brickley, D and Guha, RV. (2004). *RDF vocabulary Description Language: RDF Schema.* W3C Recommendation 10 February 2004. W3C. http://www.w3.org/TR/rdf-schema/

Canese, K., Jentsch, J and Myers, C. (2003). *PubMed: The Bibliographic Database.* The NCBI Handbook, Part 1. The Databases.

Chalifa-Caspi, V., Yanai, I., Ophir, R., Rosen, N., Shmoish, M., Benjamin-Rodrig, H., Iny Stein, T., Shmueli, O., Safran, M. and Lancet, D. (2004). *GeneAnnot: comprehensive two-way linking between oligonucleotide array probesets and GeneCards genes*. Bioinformatics 20, 9: 1457-1458.

Chaudhri, V., Farquhar, A., Fikes, R., Karp, P and Rice, J. (1998). *OKBC: A Programmatic Foundation for Knowledge Base Interoperability,* Proceedings of AAAI-98, July 26-30, Madison, WI.

Cocchiarella, N. B. (1991). *Formal Ontology*. In: Burkhardt H. and Smith, B. (eds) Handbook of Metaphysics and Ontology. Philosophia Verlag, Munich, pp: 640-647

Conen, W and Klapsing, R. (2001). *Logical Interpretations of RDFS – A compatibility Guide.* Working paper version: 1.0. http://citeseer.ist.psu.edu/635990.html

Connolly, D., Harmelen, F. V., Horrocks, I., McGuinness, D. L., Patel-Schneider, P. F. and Stein L. A. (2001). *DAML+OIL Reference Description*, W3C Note 18 December 2001. W3C. http://www.w3.org/TR/daml+oil-reference

Cornford, T and Smithson, S. (1996) *Project Research in Information Systems*. Basingstoke UK: Macmillan, p 44.

Davis, R., Shrobe, H., and Szolovits, P. (1993). *What is a knowledge representation?* AI Magazine, Spring:17–33.

Domingue, J. (1998). *Tadzebao and WebOnto: Discussing, Browsing, and Editing Ontologies on the Web.* Proceedings of the 11th Knowledge Acquisition for Knowledge Based Systems Workshop, Banff, Canada.

Eilbeck, K and Lewis, S. E. (2004). *Sequence Ontology Annotation Guide*. Comparative and Functional Genomics. 2004 (5):642-647

Eilbeck, K., Lewis, S. E., Mungall, C. J., Yandell, M., Stein, L., Durbin, R., Ashburner, M. (2005). *The Sequence Ontology: A tool for the unification of genome annotations*. Genome Biology 2005(6):R44

Falbo, R. A., Guizzardi, G and Duarte, K. C. (2002). *An Ontological Approach to Domain Engineering*. http://citeseer.ist.psu.edu/dealmeidafalbo02ontological.html

Farquhar, A., Fikes, R and Rice, J. (1997). *The Ontolingua server: a tool for collaborative ontology construction,* International Journal of Human-Computer Studies, 46, page: 707-727.

Fikes, R and Kehler, T. (1985). *The Role of Frame-Based Representation in Reasoning*, CACM 28(9): 904-920.

Genesereth, MR and Fikes, RE. (1992). *Knowledge Interchange Format Version 3 Reference Manual*, Logic-92-1, Stanford University Logic Group. http://citeseer.ist.psu.edu/genesereth92knowledge.html

Gennari, J. H., Musen, M. A., Fergerson, R. W., Grosso, W. E., Crubezy, M., Eriksson, H., Noy, N. F. and Tu, S. W. (2003). *The Evolution of Protege: An Environment for Knowledge-Based Systems Development*. International Journal of HumanComputer Studies, 58(1), page: 89--123.

George MA., Fellbaum, C and Miller, KJ. (2006). *Five Papers on WordNet.* retrieved December, 2006. ftp://ftp.cogsci.princeton.edu/pub/wordnet/5papers.ps

Goble, C. A., McGuinness, D. L., Moller, R and Patel-Schneider, P. F. (2001). *OilEd: a reasonable ontology editor for the semantic web.* Proceedings of the Intl. Description Logics Workshop, volume 49 of CEUR Workshop.

Gomez-Perez, A., Fernandez-Lopez, M and de Vicente, A. (1996). *Towards a method to conceptualize domain ontologies.* In Proceedings of the ECAI'96 Workshop on Ontological Engineering. van der Vet. P(ed). Budapest, Hungary, page: 41-52.

Gomez-Perez, A. (2002). *A Survey on Ontology Tools*. Proceedings of the OntoWeb Ontology-based information exchange for knowledge management and electronic commerce, deliverable D1.3. IST-2000-29243.

Gomez-Perez, A., Lopez, M. F and Corcho, O. (2004a). *Theoretical Foundations of Ontologies.* Chapter 1. Page 1-44. Ontology Engineering. Springer-Verlage London Limited, London, UK.

Gomez-Perez, A., Lopez, M. F. and Corcho, O. (2004b). *The Most Outstanding Ontologies*. Chapter 2. Page 47-105. Ontology Engineering. Springer-Verlage London Limited, London, UK.

Gomez-Perez, A., Lopez, M. F and Corcho, O. (2004c).*Ontology Tools*. Chapter 5. Page 293-361. Ontology Engineering. Springer-Verlage London Limited, London, UK.

Gottgtroy, p and Jain, V. (2005). *Linking brain diseases through gene networks using ontology maps*. Proceedings of NZ Bio 2005, Auckland, New Zealand.

Gottgtroy, P., Kasabov, N and MacDonell, S. (2004). *An ontology driven approach for knowledge discovery in Biomedicine*. Proceeding in VIII Pacific Rim Intl. Conference. AI (PRICAI) – LNAI 3157, pp. 53-67, Auckland, Newland.

Gottgtroy, P., Kasabov, N. & MacDonell, S. (2003). *An ontology engineering approach for Knowledge Discovery from data in evolving domains*. Proceedings of the Data mining IV. Southampton, Boston: WIT press.

Gruber, T. R. (1993a). *A translation approach to portable ontology specification.* Knowledge Acquisition 5(2). pp: 199-220.

Gruber, TR. (1993b). *Toward principles for the design of ontologies used for knowledge sharing.* Technical Report KSL 93-04, Stanford University. Substantial revision of paper presented at the International Workshop on Formal Ontology, Knowledge Acquisition, 5:199-220. Padova, Italy.

Gruninger, M and Fox, M. S. (1995). *Methodology for the design and evaluation of ontologies.* In Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing, International Joint Conference on Artificial Intelligence (IJCAI-95), Montreal, Canada.

Guarino, N. (1998). *Formal Ontology in Information Systems*. Proceedings of FOIS'98, Trento, Italy, 6-8 June. Amsterdam, IOS Press, pp. 3-15.

Guarino, N and Boldrin, L. (1993). *Ontological requirements for knowledge sharing.* proceedings of the IJCAI Workshop for Knowledge Sharing and Information Interchange, Chambery, France.

Guarino, N and Giaretta, P. (1995). *Ontologies and knowledge bases: towards a terminological clarification.* In N. Mars (Ed.), Towards Very Large Knowledge Bases (pp. 25 - 32). Amsterdam: IOS Press.

Horrocks, I., Sattler, U and Tobies, S. (1999). *Practical reasoning for expressive description logics*. Proceedings of the 6th Int. Conf. on Logic for Programming and Automated Reasoning (LPAR'99), No. 1705 in Lecture Notes in Artificial Intelligence, pp. 161–180.

Humphreys, B. L and Lindberg D. A. (1993). *The UMLS project: making the conceptual connection between users and the information they need*. Bulletin of the Medical Library Association, 81(2): 170.

Karp, P. (1993). *The design space of frame knowledge representation systems*, Technical Report 520, SRI International AI Center. http://www.cs.umbc.edu/771/papers/karp-freview.pdf

Kasabov, N, Jain. V., Gottgtroy, P., Benevuska, L. & Joseph, F. (2007). *Evolving Brain-Gene Ontology and Simulation System (BGOS):Towards Integrating Bioinformatics and Neuroinformatics Data,*Information and Knowledge to Facilitate Discoveries. Special Issue of Neural Networks. In Press.

Kasabov, N, Jain. V., Gottgtroy, P., Benevuska, L. & Joseph, F. (2007). *Brain gene ontology and simulation system (BGOS) for a better understanding of the brain.* Special Issue ofCBS (Cybernetics and Systems: An International Journal)-Knowledge Management and Ontologies. Cybernetics and Systems, June 2007, Vol. 38 (5), pp 495-508.

Knublauch, H., Fergerson, R. W., Noy, N. F and Musen, M. A. (2004). The Protégé OWL Plugin: An Open Development Environment for SemanticWeb Applications.

Stanford Medical Informatics, Stanford School of Medicine. http://protege.stanford.edu/plugins/owl/publications/ISWC2004-protege-owl.pdf

Lancet, D., Ron, S., Shmoish, M., Madi, A., Sirota, A., Noy, K., Rosen, N., Greenshpan, O., Shmueli, O., Safran, M., Aumann, Y and Strichman-Almashanu, L. (2005). *GeneDecks: A Systems Biology Facilitator with Combinatorial GeneCards Display*. Transcriptome 2005. http://www.genecards.org/papers/transcriptome_2005.pdf

Lenat, DB., and Guha, RV. (1989). *Building large knowledge-based systems: Representation and inference in the Cyc project*, Addison-Wesley, Reading, MA.

MacGregor, R. (1991). *Inside the LOOM chasifier*. SIGART bulletin 2(2). Page: 70-76.

Maedche, A., Motik, B., Stojanovic, L., Studer, R. and Volz, R. (2003). *Ontologies for enterprise knowledge management*. IEEE Intelligent Systems, January/February

Mahesh, K and Nirenburg, S. (1995). *Semantic Classification for Practical Natural Language Processing*. http://citeseer.ist.psu.edu/mahesh95semantic.html

Manola, F and Miller, E. (2004). *Resource Description Framework (RDF) Primer*. W3C Recommendation 10 February 2004. W3C. http://www.w3.org/TR/rdf-primer/

Marias, J. (2001). *Historia de la filosofia*, 4th edn, Filosofia y Pensamiento, Alianza Editorial, Madrid, Spain.

Matuszek, C. M. Witbrock, R. Kahlert, J. Cabral, D. Schneider, P. Shah and D. Lenat. (2005). *Searching for Common Sense: Populating Cyc from the Web.* In Proceedings of the Twentieth National Conference on Artificial Intelligence, Pittsburgh, Pennsylvania.

McGuinness, D. L., Fikes, R., Rice, J. and Wilder, S. (2000). *An Environment for Merging and Testing Large Ontologies. Principles of Knowledge Representation and Reasoning.* Proceedings of the Seventh International Conference on Knowledge Representation(KR2000). San Francisco, CA.

Miller, G. A. (1995). *WordNet: a lexical database for English*. Communications of the ACM 38(11), page: 39-41.

Minsky, M. (1975). *A Framework for Representing Knowledge*, in Patrick Henry Winston (ed.), The Psychology of Computer Vision, McGraw-Hill, New York.

Mizoguchi, R., Welkenhuysen, J and Ikeda, M. (1995). *Task ontology for reuse of problem-solving knowledge*. In N.J.I. Mars, (ed), Proceedings of the 2nd International Conference on Knowledge Building and Knowledge Sharing, Twente, The Netherlands, pages 46—57.

Mizrachi, I. (2004). *GenBank: The Nucleotide Sequence Database.* The NCBI Handbook , Part 1: The Databases.

Musen, M. A. (1992). *Dimensions of knowledge sharing and reuse*. Computers and Biomedical Research. Pp.: 435-467.

Neches, R., Fikes, R. E., Finin, T. Gruber T. R., Senator, T and Swartout W. R. (1991). *enabling technology for knowledge sharing.* AI magazine 12 (3). pp 36-56

Niles, I and Pease, A. (2001). *Origins of the IEEE Standard Upper Ontology*, Working Notes of the IJCAI-2001 Workshop on the IEEE Standard Upper Ontology, Seattle.

Noy, N. F., Fergerson, R and Musen., M. (2000). *The knowledge model of Protege-2000: Combining interoperability and flexibility*. In R. Dieng and O. Corby, (eds) Proceedings of the 12th EKAW, LNAI, pp. 17-32, Juan-les-Pins, France.

Noy, N. F and McGuinness, D. L. (2001). *Ontology Development 101: A Guide to Creating Your First Ontology*. Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880.

Nunamaker, J. F., Chen, M., and Purdin, T. (1991) *Systems development in Information Systems research.* Journal of Information Management Systems/Winter 1990-1991, Vol 7, No 3, pp.. 89-106.

Olivier,B. (2004) *The Unified Medical Language System (UMLS): integrating biomedical terminology.* Nucleic Acids Research,32,D267-D270.

Rothenfluh, T. R., Gennari, J. H., Eriksson, H., Puerta, A.R., Tu, S.W. and Musen, M.A. (1996). *Reusable ontologies, knowledge-acquisition tools, and performance systems: PROTÉGÉ-II solutions to Sisyphus-2.* International Journal of Human-Computer Studies. Pg: 303-332.

Sowa, J. F. (1999). *Knowledge Representation: Logical, Philosophical, and Computational Foundations.* Brooks Cole Publishing Co, Pacific Grove, California.

Shadbolt, N., Motta, E and Rouge, A. (1993). *Constructing Knowledge Based Systems*. IEEE Software, 10(6), page: 34-38.

Shankar, R., Tu, S and Musen, M. (2002). *Use of Protégé-2000 to Encode Clinical Guidelines*. http://citeseer.ist.psu.edu/shankar02use.html

Staab, S., Schnurr, H. P., Studer, R and Sure, Y. (2001). *Knowledge processes and ontologies*. IEEE Intelligent Systems, 16(1):26-34.

Studer, R., Benjamins, V. R and Fensel, D. (1998). *Knowledge Engineering: Principles and Methods.* IEEE Transactions on Data and Knowledge Engineering 25(1-2), pp: 161-197.

Sure, Y., Erdmann, M., Angele, J., Staab, S., Studer, R and Wenke, D. (2002). *OntoEdit: collaborative ontology engineering for the semantic web.* Proceedings of

the International Semantic Web Conference, Springer-Verlag, pp. 221-235, Sardinia, Italy.

Swartout, B., Ramesh, P., Knight, K and Russ, T. (1997). *Toward Distributed Use of Large Scale Ontologies*. In Farquhar, A., Gruninger, M., Gomez-Perez, A., Uschold, M, van der Vet, P. (eds) AAAI'97 Spring Symposium on Ontological Engineering , Stanford University, California, page: 138-148.

The National Library of Medicine. (2003). *UMLS knowledge Sources: Metathesaurus, Semantic Network and Specialist Lexicon*. Unified Medical Language System. 14th Edition.

Uschold, M. and King, M. (1995). *Towards a Methodology for Building Ontologies.* In Proceedings of the IJCAI'95 Workshop on Basic Ontological Issues in Knowledge Sharing Skuce D (eds). Montreal, Canada, pp. 6.1-6.10.

Uschold, M. and Gruninger, M. (1996). *Ontologies: Principles, Methods and Applications*. Knowledge Engineering Review 11(2).

Uschold, M., King, M., Moralee, S and Zorgios, Y. (1998). *The Enterprise Ontology*. The Knowledge Engineering Review. 13 (1): 31-89.

Uschold, M and Jasper, R. (1999). *A Framework for Understanding and Classifying Ontology Applications.* Proceedings of the IJCAI'99 Workshop on Ontology and Problem Solving Methods: Lessons Learned and Future Trends. Benjamins VR (ed) Stockholm, Sweden.

Vossen, P. (1998). *EuroWordNet: A multilingual database with Lexical Semantic Networks.* Kluwer Academic Publishers, Dordrecht, The Netherlands.

Wang, Y., Gottgtroy, P and Jain, V. (2006). *Ontology visualization: the brain gene ontology.* Proceeding in the SCIS Postgraduate Conference 2006, School of Computer and Information Science, Auckland University of Technology, Auckland, New Zealand. (unpublished)

Yeh, I., Karp, P. D., Noy, N. F and Altman, R. B. (2003). *Knowledge acquisition, consistency checking and concurrency control for Gene Ontology (GO).* Bioinformatics, Vol. 19. pp. 241-248. Oxford University press, UK.