Ontology Development for Drug-Disease Knowledge Management

By

Namira Mohammadi Dinani

A Thesis Submitted to Department of Electrical Engineering

in Partial Fulfillment of the Requirement for the Degree of

Master of Science

at

Lakehead University

Thunder Bay, Ontario, Canada

June 2012

Abstract

Traditionally, physicians rely on medical knowledge learned or acquired through practice to make a correct decision in diagnosis of diseases and affected medications. Progress in knowledge base systems and related Information Technology changed this situation by providing huge amount of medical information that can support physicians in the decision making. Integrating this great amount of information to retrieve the superlative results is however a demanding job. By introducing the concepts of Semantic Web, the sources of medical knowledge have amended to acquire the advantages of this concept and its technologies in gathering and representing the information and recommending superior decision from comprehensive information. Particularly, ontologies are recognized to enhance the efficiency of information management significantly and improve the dependability of communication especially when heterogeneous actors and diverse environments are involved.

In this thesis, various Semantic Web techniques have been employed to support data integration to assist physicians in the process of drug recommendation. The thesis proposes a novel approach for supporting drug recommendation decisions by modeling a Semantic Web-based infrastructure correlating comprehensive medical knowledge which allows making ontology inferences and knowledge discoveries. In this work, we devise a Disease-Drug Ontology (DDO), an ontological model which demonstrates relations between human diseases and their relevant drugs and medications. The DDO, formalized in OWL, allows the integrated representation of various sources of ontologies and data schemas and overcomes the heterogeneity problem among these different sources by applying proper matching techniques. An automated reasoning is performed over the ontology using a Description Logic Reasoner in order to validate the DDO. Our model is also composed of an ontology crawler that provides physicians, by direct queries from DDO, to facilitate the process of making decisions for accurate drug recommendation. More importantly, our system consists of a unique rule-based inferential engine employing drug rules and patient data for the purpose of suitable drug recommendation. In order to prototype the key services of the system and reveal the validity of our

semantically integrated Disease-drug knowledge base, some case studies are provided and the obtained results are very promising.

Acknowledgment

I would like to offer my sincerest gratitude to my supervisor, Dr. Rachid Benlamri, who has always been supportive throughout my thesis with his patience and knowledge. I attribute the level of my Master degree to his encouragement and effort, without him this work would not have been completed. One simply could not wish for a better or friendlier supervisor.

I am grateful to all my professors at Lakehead University and the Northern Light Research Group for their helps and encouragements.

I would also like to thank all my friends and colleagues at Lakehead University especially Osama Mohammed, Shahab Alilou, Bona Ater, Amir GhanbariBavarsad and Luke Docksteader who are always willing to help and give their best suggestions throughout my work.

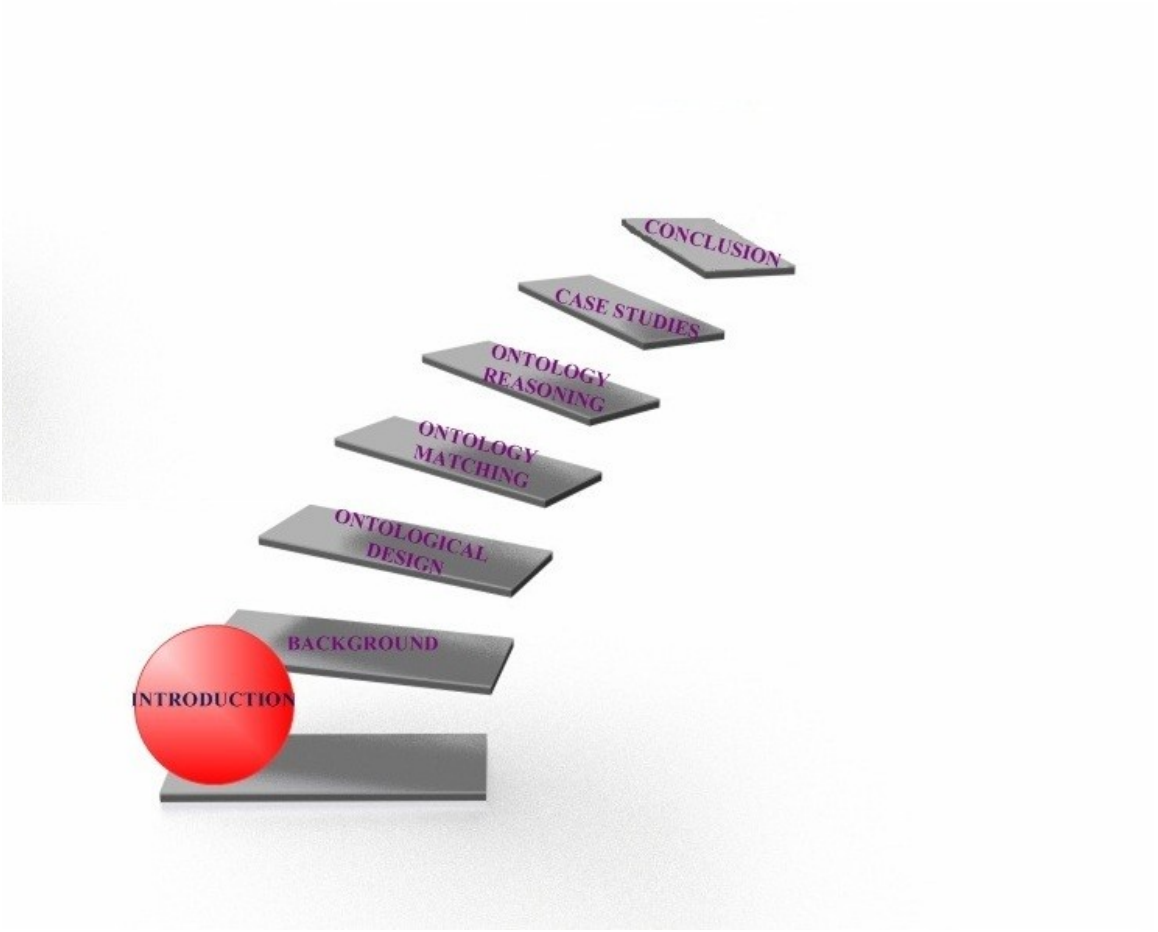
My special gratitude goes to my family for their understanding, affection and support all throughout the years. My parents who have always been supporting and encouraging me through all the challenges of my life for which I am eternally grateful. My sister is the best friend someone may ask for who is always there to cheer me up. My husband who has always stood by me through the difficult times and without his unconditional love and continued encouragement, I wouldn't have accomplished this work and achieved this milestone in my life.

Table of Contents

List of Figures

List of Tables

CHAPTER 1
INTRODUCTION



1.1 Motivation

Prescribing the correct and proper medications for humans' diseases is one of the tough challenges that physicians and healthcare clinicians may face every day. This is not only because of the various and numerous number of diseases and their available drugs, but also due to the lack of a unified documentation in the healthcare system. Most of the information in the health care system is paper-based which causes a limited coordination of data and services. Information in the medical domain lives on islands, which results in raising costs and increasing medical errors. In order to make the right decision to recommend effective treatment, a general physician must carefully investigate a patient's medical history, his diagnosis and the diverse aspects of the relevant medications. Although, most physicians obtain this proficiency after gaining a few years of experience, estimates show that 1.5 million people are harmed every year from preventable mistakes in prescribing and administering medications in the United States alone [1]. Information insights coupled with clinical collaboration can dramatically improve the quality of the process of drug recommendation, patient safety and outcomes, while also being quite cost effective. A smarter drug recommendation system starts with better connections, better data and faster and more detailed analysis. Thus, a forecasting informatics model that can develop data capture, integration and recommendation of suitable medication based on integrated medical knowledge around drugs and diseases is highly desirable. To achieve this, data across knowledge domains of drugs and diseases needs to be linked in an adequately standardized manner to permit efficient inferences. Linking the heterogeneous data from various sources of medical knowledge is a fundamental challenge. Moreover, lacking a well-defined data framework can obstruct the inference capability of the model.

It is thus essential to overcome these limitations during the development of our proposed model in order to reach a medical infrastructure that can handle data complexity, represent both semantic and logic relationships and allow interoperability in a global manner. The development of Semantic Web and knowledge representation technologies offers a promising platform that can broadly integrate heterogeneous data while dealing with semantics and complexity of knowledge interoperability.

Our goal is to employ medical knowledge and semantic concepts to integrate and represent disease and drug relevant entities and the complex relationships among them based on the clinical and therapeutic aspects to assist physicians in identifying novel suitable medications for drug recommendation. In this work, we design and build a semantic web-based infrastructure by associating comprehensive medical information in the domains of drugs and diseases which allows ontology inference and knowledge discovery to aid in selecting the appropriate medications for the diagnosed diseases.

1.2 Thesis Contribution

We implement a proactive drug recommendation system that could analyze and suggest the potential available drugs for diseases based on integrated medical/clinical knowledge around drugs and diseases. The system is also capable to suggest suitable drug doses and potential interactions of drugs with the active medications of the patient. The system is built based on the semantic web approach and its technologies to integrate the disease-drug information for revealing the suitable drug recommendation of diagnosed diseases. The proposed ontology in this work is considered as one of the pioneering ontologies which bring drugs and diseases together based on their therapeutic aspects. Furthermore,

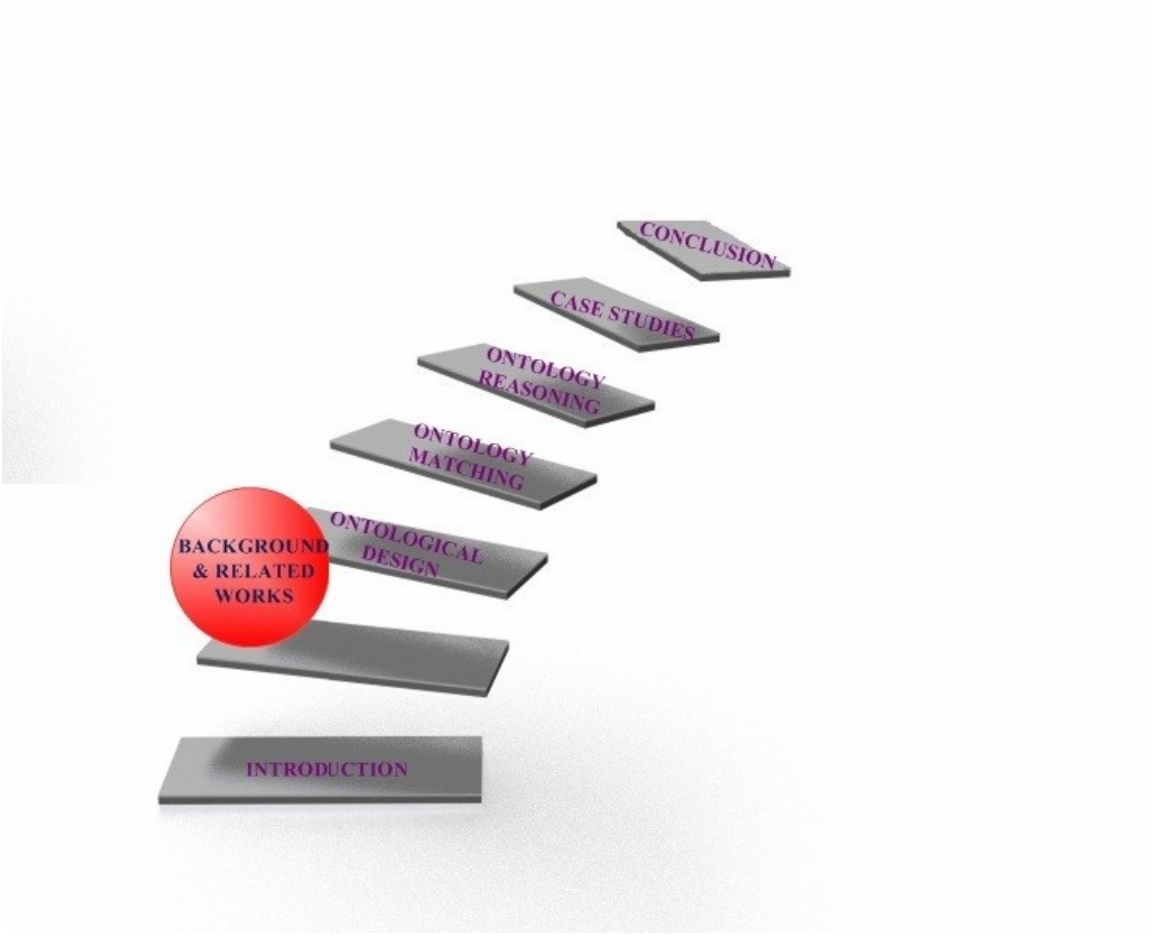
the system shows the great amount of extensibility and adaptability due to applying the reliable and dependable sources of knowledge in developing the proposed ontology. The recommended semantic web framework is designed to allow evolving and querying the information quickly and thoroughly. The system also provides the capability to handle flexible and complex queries and to allow inference across the related features of drugs and diseases.

1.3 Thesis Outline

The rest of the thesis is organized as follows. Chapter 2 outlines some basic background of Semantic Web technologies such as the Extensible Markup Language (XML), Resource Description Framework (RDF), Web Ontology Language (OWL) and Ontology. It also reviews former approaches in drug recommendation and medical knowledge representation field. Chapter 3 presents the influence of ontology on the domain of medical knowledge, and then explains the overall system architecture. Afterward, it focuses on describing our attempt in building the ontology framework including the design, structure and examples of key entities and properties formalized in OWL. Chapter 4 introduces the concept of ontology matching and reviews a variety of classifications of matching techniques. Then it explains how these techniques applied in construction of our ontology framework. Chapter 5 describes the process of validation of ontology framework via a Description Logic Reasoner. Subsequently, it presents the role of query engine and rule-based inference engine in our system and how they are implemented. Chapter 6 portrays some kinds of potential results of query engine and provides some case studies to demonstrate the validity of our semantically integrated

Disease-Drug knowledge base for revealing reliable drug recommendations. Chapter 7, the last Chapter, summarizes the work and provides some perspectives on future development to improve the quality and reliability of drug recommendations and enhance the robustness of the infrastructure.

CHAPTER 2
BACKGROUND & RELATED WORKS



This Chapter discusses the fundamental technical background of ontology and overviews the related research work in this field. In order to understand the ontology more profoundly it is essential to first understand the semantic web on which it relies.

2.1 Semantic Web

The World Wide Web which is basically written in Hyper Text Mark-up Language (HTML) is designed to be understood and interpreted by humans. Gradually by increasing the amount of available data on the current web, the process of finding, organising, accessing and maintaining the information for the users turns to be extremely difficult. Therefore this notion comes up that by shifting the retrieval of data from users to the computers, the web can be optimized and become much more goal based rather than task based. Such a desire leads to the concept of semantic web which is brought up to enhance some of the weaknesses of the current web. The term semantic web is introduced by the inventor of World Wide Web, Tim Berners-Lee. The idea behind that is to extend the capabilities of the current classic web of documents and create a web of data that can be accessed and processed directly or indirectly by machines, devices and computers in addition to users [2-4]. The ultimate goal of semantic web is to provide a common framework that allows data to be shared and reused across applications, enterprises and community boundaries and enables computers and people to work in mutual cooperation [3, 5].

2.2 Semantic Web Architecture and Concepts

The architecture of semantic web is composed of multiple layers (Figure 2.1). At the most basic level, semantic web is dependent on the Unicode and Uniform Resource Identifier (URI) which is simply a web identifier. Unicode is a standard for encoding a set of characters. By using such a standard form, all human languages can be written and read on the web. URI is a sequence of characters which provides a standard form to identify the resources [6-8].

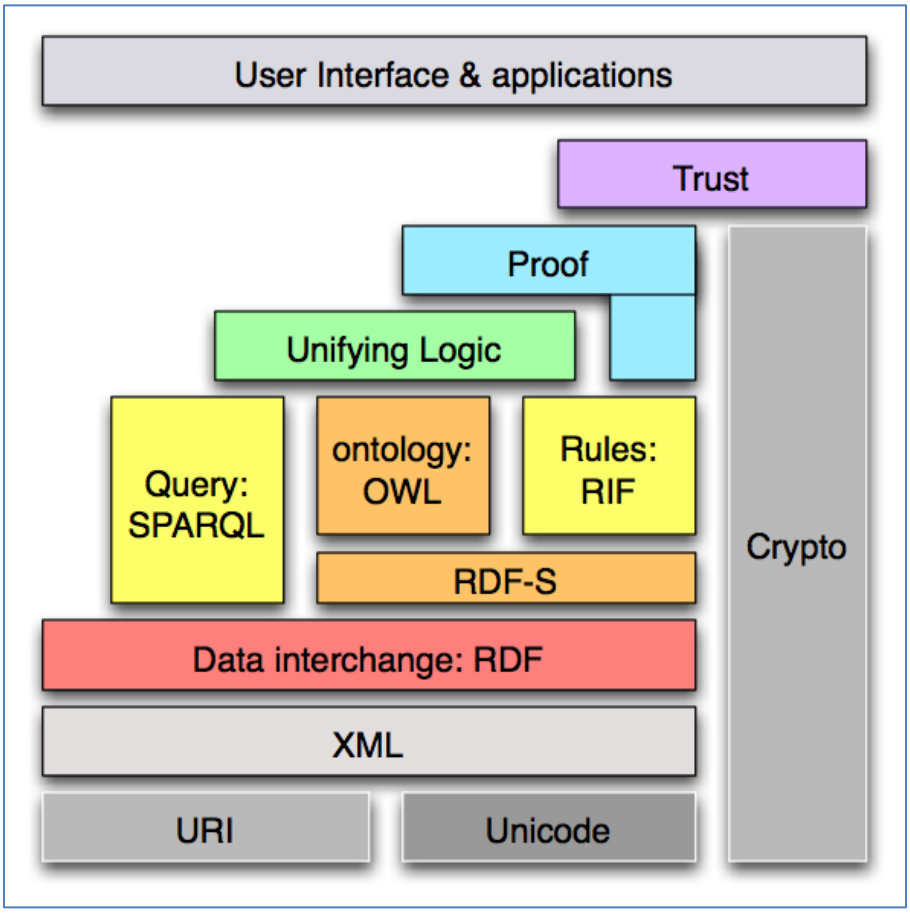


Figure 2.1: Semantic Web Architecture

Extensible Markup Language (XML) constructs the second layer of semantic web architecture. XML is a form of markup language much like HTML that has been

employed on the web since its foundation; however unlike HTML, there is no set of predefined tags in XML. Instead, the beauty of XML lies in the fact that it is extensible and tags can be defined and utilized as required by the specific application. In addition to the difference mentioned, XML is designed to carry data in comparison to HTML which is used to display data. Another advantage of XML is that it is a self-descriptive language which can be used between different platforms and programming languages and still expresses complex messages and functions [3, 9, 10]. XML Schema restricts the structure and content elements of XML documents. The World Wide Web Consortium (W3C) recommends XML namespaces which are applied to offer uniquely named elements and attributes in a XML document. In the semantic web architecture, XML layer with XML namespace and XML schema certify that the common syntax is used in the semantic web [11].

The other important technology for developing the semantic web is the Resource Description Framework (RDF). RDF which was initially designed as a Meta data model is a framework for describing information and resources on the web. Putting information into RDF files, makes it feasible for computer programs to search, discover, pick up, collect, analyze and process information from the web. Since RDF documents are written in XML, they can easily be exchanged among different types of operating systems and application languages. RDF is based on statements which are known as triples. Each triple includes three main parts of subject, predicate and object like a sentence in natural languages. Therefore each triple can be modeled as a graph with two nodes of subject and object that are connected by the edge of predicate [5, 11, 12, 13].

Figure 2.2 illustrates the RDF graph of Coronary Heart Disease and one of the drugs that it may be treated by.

The subject represents the resource itself and must be defined by URI. The predicate which is a relationship is also a URI since each relationship has a standard definition that is expressed through its unique URI. The object which is the resource or state that is being related to is considerable because it can be defined as a URI or a literal value. RDF itself is just a description of graph formed by triples. RDF Schema (RDFS) extends RDF vocabulary to allow describing taxonomies of classes and properties which provides major elements of the description of ontologies [13-14].

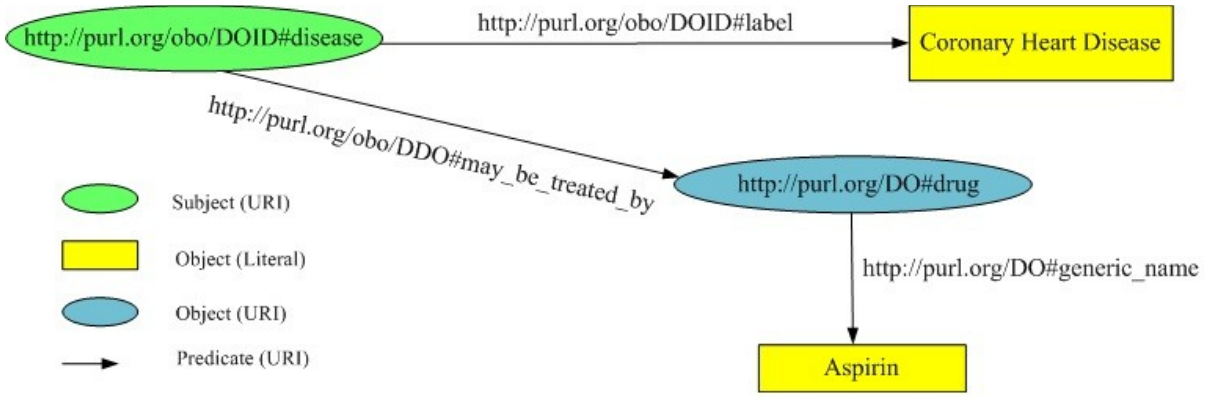


Figure 2.2: RDF Graph for a disease and its possible drug

The next fundamental layer of semantic web architecture is Web Ontology Language (OWL) that can be compared to a grammar system of the Semantic Web. OWL roots are in description logic and its foundational goal is to bring the reasoning power of description logic to the semantic web. OWL which is built on top of the RDF also provides additional standardized vocabulary, superb machine interoperability and a well-

defined syntax which is much stronger than RDF [11, 15]. Mentioned specifications make OWL a suitable language to be employed explicitly for development of an ontology which is a demonstration of concepts and their relationships. Ontology is discussed in more details further in section 2.3.

Based on various range of compatibility and restrictions, OWL comes in three distinct categories of OWL Lite, Web Ontology Language Description Language (OWL DL) and OWL Full. OWL Lite is the most restricted one. It is the simplest OWL sublanguage which can be used for classifying resources in a hierarchical form. OWL Lite is quite useful when an uncomplicated OWL system is required with limited resources or limited speed. OWL DL is a more comprehensive system which comes with greater potential and more flexibility, while still being possible to implement and use consistently. OWL DL is designed for full description logic which offers decidability of reasoning systems. It also allows for more complicated analysis, classification, more complex relationships and properties than OWL Lite. OWL Full is the final level of complication which is theoretically unlimited model, where relationships and classifications can be widely expanded with a great deal of complexity. It mostly targets the users who seek maximum expressiveness and syntactic freedom of RDF with no computational guarantees. OWL Full can fully support OWL DL and OWL Lite systems, although it is impossible to predict and implement correctly. The ontology developers must select OWL specie that best suits their needs. [11, 15, 16, 17]

The role of the Logic layer is to support a powerful logical language for making inferences that make the semantic web expressive enough to aid us expansively in various situations. All the semantics and rules are executed at the layers below Proof and

the result will be used to confirm the reasoning. The top layers of semantic web architecture express both the proof and the trusted data together to demonstrate that the results are trustworthy. In order to make inputs reliable, cryptography means such as digital signatures can be used. A user interface application can be built on the top of all these layers. [11, 18]

2.3 Ontology

The term ontology initially comes from philosophy in which it is a theory about the nature of existence. In the context of knowledge sharing ontology is an explicit specification of a shared conceptualization of a domain of interest. From the view of computer science and artificial intelligence, ontology represents a domain of knowledge or discourse as a set of concepts (classes), their attributes (properties), instances of those concepts (individuals) and the relationships in which classes and individuals can be related to one another [5, 19]. Implementation of ontology is the heart of all semantic web based knowledge representation. OWL is the language that is recommended by W3C Semantic Web standard for encoding the ontologies [17].

While database schema models data at the physical or logical level, ontology is known for modeling the knowledge in the semantic level. Therefore, it performs a vital task in representation of a particular domain which allows for automatic reasoning and interpretation with applicable semantic context. Based on its independence from the lower levels of data models, ontology is capable of integrating and sharing data between heterogeneous information resources and specifying interfaces to independent, knowledge-based services. While offering advantages to facilitate interoperability among

multiple heterogeneous systems, ontology also provides services for answering queries and reusing knowledge resources [3, 19, 20]. In other words, ontology can be used as a way of communication between the human being and a system or system-to-system. Ontology can be used in information retrieval and knowledge management. The more perfect the framework of domain ontology is; the more accurate information can be provided [21].

Since the construction of ontology from scratch is a very time consuming task, ontology developers try to reuse existing ontologies whenever possible. However, handling complex ontologies that are constructed from multiple knowledge domains also brings another issue up that needs to be addressed by ontology engineering. Further details regarding ontology engineering and its relevant concepts are described in Chapter 4.

2.4 Related Works

During last two decades, the development of semantic web and ontology has had a great impact on knowledge representation. This is highly significant especially when the knowledge can be applied to do some reasoning as part of a decision support system [22]. The medical and biomedical domain is also not an exception and developing of semantic web can have dramatic improvements on this area of knowledge. Semantic web framework can help organize, query and evolve the enormous amount of information in these fields quickly and thoroughly. Although much research has been conducted recently on generation of medical ontologies based on the semantic web, most of them have been focused on differential diagnosis of diseases either specific or general. Hence, the systems that are developed from the aspect of drug recommendation in the domain of

medical knowledge are very limited [23]. Some of existing systems in this field are discussed in the following paragraphs.

T.M. Swe et al. in [24] have proposed a case based reasoning methodology for querying a diagnostic knowledge base using ontology on diagnosis of the tuberculosis (TB) disease and recommending the relevant treatments. Their system consists of two main components: domain ontology and a case-based reasoning module. Their domain ontology contains case knowledge, concepts used to describe cases and relationships between the concepts. This domain ontology can be considered as a disease ontology that, in their case, only contains the information about tuberculosis (TB) disease. Case-based reasoning module is based on a technique to retrieve the previous similar cases' information and reuse them to solve the new case's problem. The new solution for the new case can be revised and also retained as a new case in the system.

Nevertheless, one of the main problems of their approach is that it cannot be expanded easily to all diseases; hence, the ontology is only made for the treatment of tuberculosis (TB) disease and it is not considered as a standard ontology. Besides, suggestion of treatment based on the previous cases, alone, cannot satisfy all cases in the area of medical knowledge. Therefore the reliability and applicability of the system is vastly limited.

R.C. Chen et al. in [25] have developed a recommendation system for anti-diabetic drugs. The purpose of their study is to aid the physicians to make a right decision in selecting the anti-diabetic drugs. They have constructed two separate ontologies in their work: patient data ontology and anti-diabetic drugs ontology. Patient data ontology is applied to store personal information, history and test results of the patients and anti-diabetic drugs

ontology has been designed to be used as the source of medicine knowledge. Based on the results of some tests such as liver function test, glucose level of the patient, and the medicine knowledge in the anti-diabetic drugs ontology, they have applied some rules to their rule engine. The final result of the rule engine is considered as the drug recommendation for the diabetic patients. Although their system can be applied as drug recommendation system for diabetic patients, but it has a poor capability to be generalized to other diseases and to be applied in a real system. The assumption of the system is based on one disease and no part has been designed for analyzing different diagnoses. The use of some reliable resources such as the American Association of Clinical Endocrinologists data in the construction of ontology [25] is a positive point, however since their ontology have been built from the scratch, its development into more applicable one is still an extremely time-consuming task.

An ontological system for chronic disease management based on the electronic medical records has been offered by A.M. Iqbal et al. in [26]. Their proposed ontology is comprised of Computer-based Patient Record (CPR) [27] Ontology which is augmented by mapping some concepts from the Electronic Health Record (EHR) Model and Chronic Disease Management (CDM) Model into it. Since their ontology is built in OWL-DL, the decision support system can be implemented through reasoning based on the description logic. The system is mostly patient centered and focuses on the diagnosis of chronic diseases. According to their results, they were not able to capture the medication into their ontology. Although, they do not offer any treatment for the chronic diseases in their ontological system, however, the idea of mapping and alignment between concepts of different source of knowledge can be extracted from their work.

M. Hadzic et al. in [28] have designed an ontology-based approach to gain support for research into genetic human diseases. Their proposed ontology represents the information in four categories including types of diseases, symptoms, genetic and environmental causes and treatments that might be available such as drug therapy, physiotherapy, surgery, etc. The components of their ontology are mainly based upon the biomedical aspects and the relationship of genome and genes to diagnostics of diseases. Therefore, it cannot be referred to when the clinical aspects of diseases and drugs are willing to be considered more. Besides, the treatment category of their ontology is very general and cannot lead to a drug recommendation for a patient. No reasoning has been done on the ontology in this work to demonstrate the support of the rule engines for the proposed ontology.

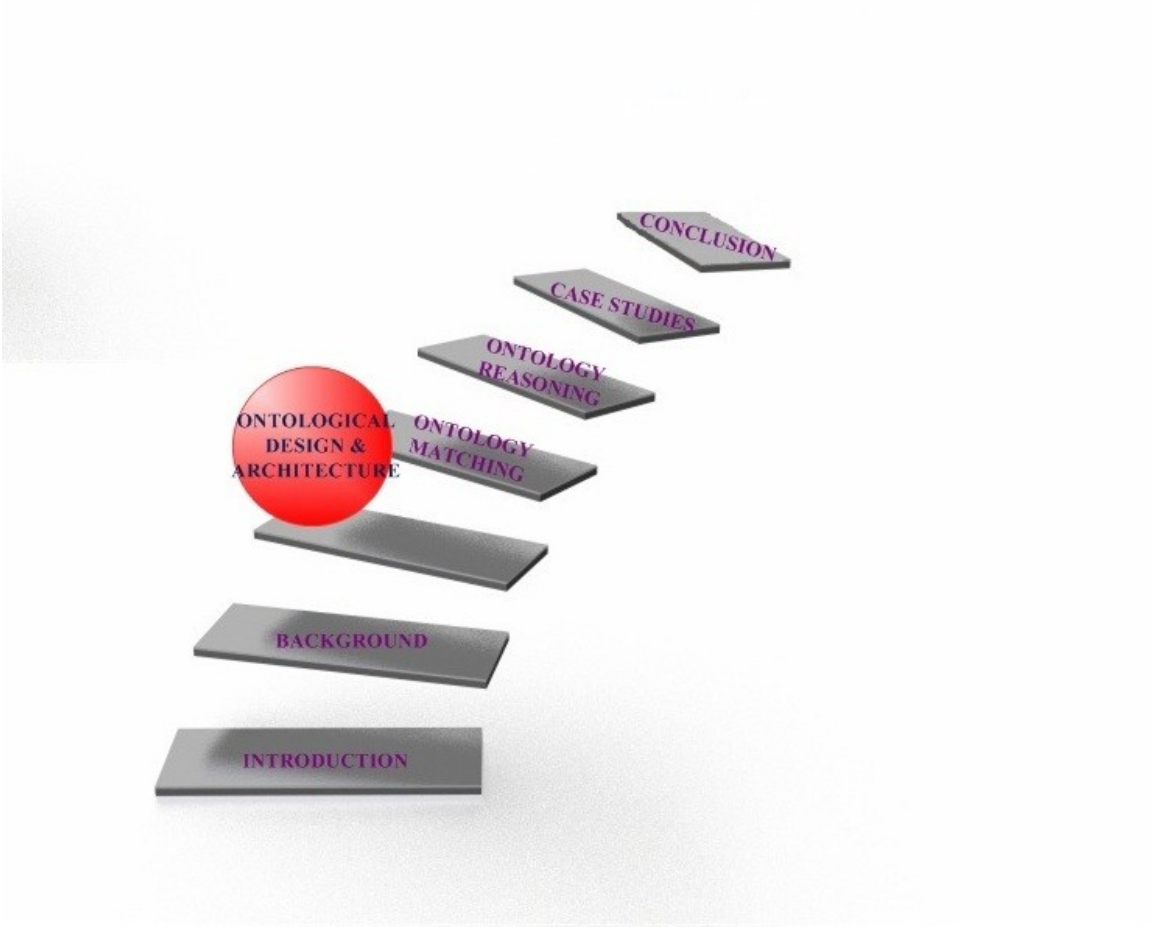
A.X. Qu et al in [29] have developed a semantic structure that helps in discovering treatments for a disease from drug entities that are already approved for another disease. They have striven to make relationships between the pharmacological aspects of drugs and knowledge of biological systems and disease processes in their proposed semantic infrastructure. This work is done based on the design of Disease-Drug Correlation Ontology (DDCO). The developed DDCO which is formalized in OWL integrates multiple ontologies, vocabularies and datasets that extracted from pharmacological and biological domains. Their knowledge framework which is capable of interconnecting drug actions and disease mechanisms across biological contexts demonstrates a great flexibility for data mining and reasoning across the range of human diseases. Although, their system recommends a great structure of relationships between drugs and diseases, however, it is mainly focused on the chemical and pharmacological aspects of drugs and

it can be specifically applied in the process of discovery of drug development. The proposed system in this thesis is mainly concentrated on the therapeutic and clinical aspects of drugs which can be applied as a drug recommendation system.

The system that has been modeled by A. Rodriguez et al. in [23] is the most similar system to the one that is proposed in this thesis. Their system is initiated to assist the clinicians with drug prescription. In their system, physicians are required to enter the diagnosis and allergies of the patient manually, and then system returns the possible drugs that match with that diagnosis. The architecture of the system includes ontology that is constructed by the developers of the system which consists of three classes of diseases, medicines and allergies and the relationship between these concepts. In next step, the offered ontology is queried through Jena which results in the outcome of their system. Jena is a query engine that is introduced in Chapter 5. Their system has some defects that are corrected in the proposed approach of this thesis. Since their ontology has been designed from scratch, the development of such a system into a real system that includes all the diseases and drugs in the medical domain is an enormously time consuming task. The proposed approach in this thesis has fixed this problem by using the standard valid ontologies and authentic data bases in the medical domain. Their ontology is just limited to three components, which affects the reliability of the output of the system. During the process of drug recommendation, there are many other aspects that need to be considered such as age of the patient, some test results and the interactions of drugs. The proposed ontology of this thesis is concerned about these various features in order to recommend the suitable drug. Therefore our approach can provide more accurate real time results. In our approach all the information about the patient is retrieved from

the patient database automatically, which has not been offered in [23]. We also extend the outcome of our work by employing a rule engine in addition to the Jena query engine. The rule engine allows for more complicated queries based on the various rules that are defined across the ontology which leads the system into more realistic results.

CHAPTER 3
ONTOLOGICAL DESIGN & ARCHITECTURE



This Chapter presents the overall architecture of the proposed system and provides a detailed description of its main components. It first describes the impact of ontology on the domain of medical knowledge. Then presents the whole architecture of the proposed system and finally focuses on describing our effort in developing the ontology framework which can be considered as the heart of our system. The novelty of the approach is that it not only integrates authentic knowledge sources for the disease and drug entities, but it also deals with the semantic interoperability among these sources. A premise to achieve this is a knowledge framework representing medical entities and relations among them which enables inference extractions.

3.1 Ontology in Medicine

Traditionally the base of the medical knowledge has been located in the heads of experienced doctors, the ones who have dedicated years of training and practice in order to make correct decisions in diagnosis of diseases and their effective treatments [24, 28]. This practice worked well in the past when production of the new data needed huge amount of effort and the flow of the new data was not as great as to overwhelm the experts. New modern experimental techniques changed this situation quickly by providing huge amount of information. During the last few decades this information has been collected and evaluated in different databases and by now a great amount of medical knowledge can be accessed via Internet [28]. Assimilating this great amount of information to retrieve the best results is impossible for humans; therefore by introducing the concepts of semantic web and ontology in the World Wide Web, the sources of medical knowledge have also altered to get the benefits of these concepts in storing and

representing their information. Currently, it is extensively accepted that ontologies can make a major contribution to the design and implementation of information systems in the medical field [30]. Ontology can be indeed useful in medicine where it can enhance the efficiency of information management dramatically and improve the reliability and consistency of communication, especially when heterogeneous actors and different environments are involved.

The use of ontology in medicine is mostly concentrated on the representation of medical terminologies. Physicians developed their own particular languages and lexicons to assist them in gathering and communicating general medical knowledge and patient-related information proficiently. This language was appropriate for keeping, processing and spreading the records of knowledge on paper or similar media. However, the paper-based terminology systems is not able to fulfill the new expectations of healthcare information systems anymore, such as the demand for reuse and sharing of patient data, or communication of complicated and comprehensive medical concepts that was possibly expressed in different languages. Accomplishment of such a task demands a deep analysis of the structure and concepts of medical terminology. Such analysis can be achieved by assuming an ontological approach for representing medical terminology systems and integrating them in a medical ontology [30]. Although terminologies are an excellent starting point for the ontology construction, they are quite different from ontologies. Terminologies are static structures used for knowledge reference, while ontologies are created to be applied in knowledge inference and reasoning [22]. Unified Medical Language System (UMLS) can be considered as an example of terminology which contains many clinical terms and integrates about 100 different vocabularies [31].

It is currently considered as a major reference for medical terms. The following ontologies can be considered as known ontologies in the medical domain:

- NCI Thesaurus (National Cancer Institute Thesaurus) [32]: an ontology vocabulary that includes broad coverage of the cancer domain, including cancer related disease, anatomy, genes and drugs.
- ICD-10 (the tenth version of International Classification of Diseases): An international standard used to classify diseases and other health problems adopted by World Health Organization (WHO) [33].
- Human disease Ontology (DOID): an open source ontology for the integration of biomedical data that is associated with human diseases [34, 35].

Creation of medical ontologies also brings great advantages to the health care system. Ontologies can be utilized to build more powerful and more interoperable information systems in healthcare. In addition they can support the need of the healthcare process to transmit, re-use and share patient data. Besides all above, possibly the most significant benefit that ontologies may bring to healthcare systems is their ability to support the indispensable integration of knowledge and data [30].

In spite of all the advances in the construction of medical ontologies, most of standard medical ontologies are built as a single domain-specific ontology. Unfortunately, there is no credible medical ontology that makes relations between various domains of drugs and diseases based on their clinical and therapeutic aspects. Furthermore, some ontologies such as NCI Thesaurus that includes both domains, only focuses on a specific category of

diseases and its related drugs which clearly cannot serve the purpose of drug recommendation for diverse category of diseases.

3.2 Overall System Architecture

This section explains the overview of the system architecture and the main software components for setting up our system environment. Figure 3.1 describes our proposed model for the drug recommendation system which is composed of three subsystems: the ontology crawler unit, the ontology rules manager and the ontology reasoning unit.

The ontology crawler unit consists of the ontology and a query engine that connects the ontology to the Java platform which allows querying the ontologies directly. Our model contains two ontology crawlers, one for the Disease-Drug Ontology (DDO) and another one for the Patient Ontology (PO). Java is the main programming language in our system. An IDE (Eclipse SDK 3.3.1) is used as software development platform. System uses Protégé 3.4 Beta for ontology editing and knowledge acquisition purposes. Jena API, which is a Java-based framework for building semantic web applications, is used as a query engine of our ontology crawler. It is used to read the ontology framework and to create prerequisite individuals.

The ontology rules manager, which sets the rules based on ontology knowledge and the facts from other sources, works with rule engine to provide knowledge rules. Finally the ontology reasoning unit contains a rule engine that infers some facts by applying rules on the existing facts. The OpenRules is used as rule engine in our system that is described further in Chapter 5. In our model patient database feeds the patient data into the rule engine and rules manager also feeds the Drug Rules into it. The system uses Apache-

Tomcat 6.0.14 to show the final results of rule engine which can be manually started and stopped.

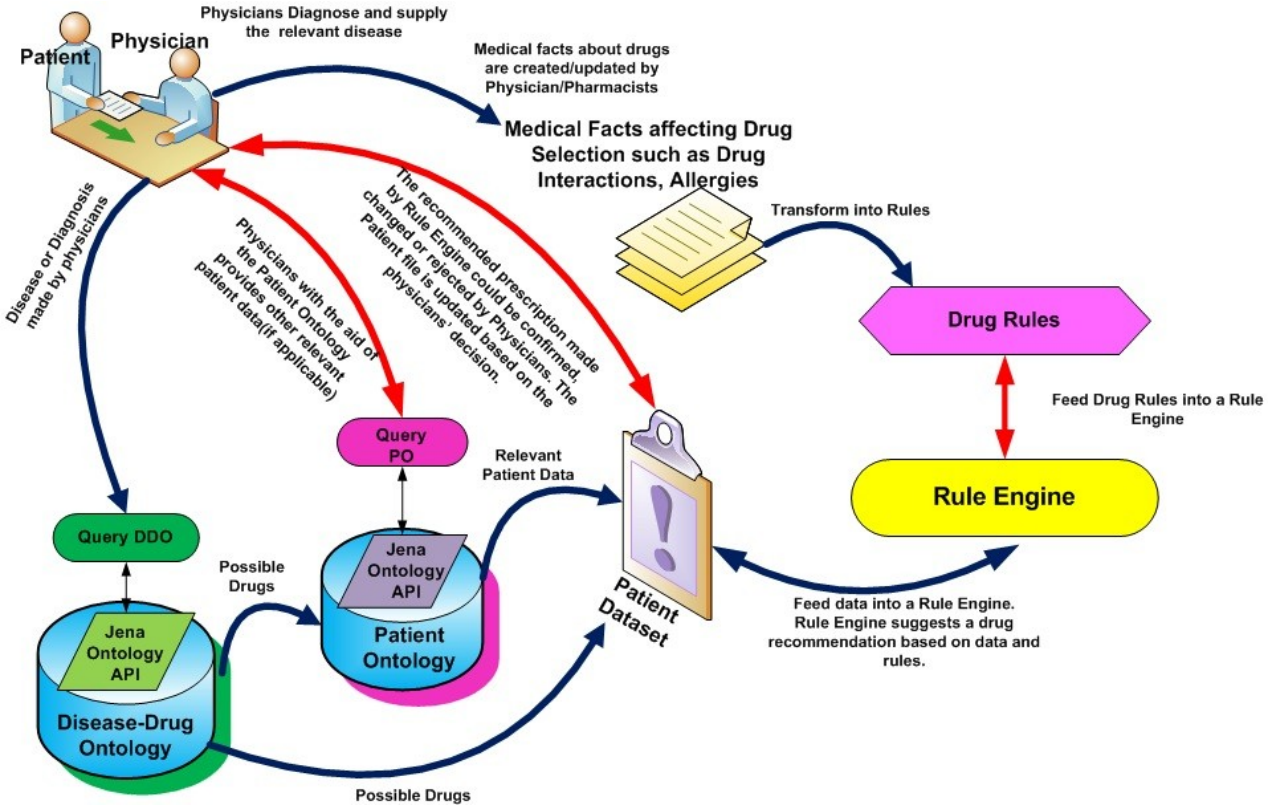


Figure 3.1: System Architecture

As shown in Figure 3.1, in our proposed model, first, physicians diagnose the relevant diseases; then DDO can be queried based on the physicians' diagnosis to show the general medical knowledge about the possible drugs for the relevant diseases. Meanwhile the physicians are able to provide other relevant patient data with the aid of the PO. The results of these queries can be transferred to the patient file, but the final result of the system as a drug recommendation system is shown after applying the rules via our rule engine. Besides the effect of medical knowledge in the DDO on drug recommendation, there are always some medical facts that may be created or updated by the physicians or

pharmacists which may also affect drug selection. To apply these medical facts, we need to transform them into rules which are called Drug Rules in our system. These rules and data from patient database feed the rule engine, and then the rule engine offers a drug recommendation based on data and rules. Ultimately the recommended prescription made by rule engine can be confirmed, changed or rejected by physicians and the patient file is updated based on the physician's decision.

3.3 Design of Disease-Drug Ontology

The first step in modeling our system is to develop a base for building our ontology. Ontologies can be classified into three major categories: upper-level ontology (general model to represent great range of tasks, domain, and application areas), domain-specific ontology (representing conceptualization of a specific domain), and application and task ontology (suitable for specific applications) [36], as shown in Figure 3.2. Accurate results for drug recommendation can only be made when a good knowledge base is available. Therefore, an ontology framework is required which contains the broad and interdisciplinary range of concepts of drugs, diseases and the relationship between them in clinical domain. Although, there are various well-accepted ontologies available in medical areas, most of them are built as a single domain-specific ontology which conceivably cannot serve our purpose. Even the existing multi-domain ontologies or controlled vocabularies, such as UMLS (The Unified Medical Language System) [31] which offer a decent framework for joining some biomedical databases, lacks sufficient coverage for the therapeutic aspect of drugs area [37]. To the best of our knowledge there is no standard medical ontology that contains the large amount of knowledge for both

domains of diseases and drugs, and provides the therapeutic relationships between the two domains. Furthermore, there is not even a single domain-specific medical ontology that provides sufficient coverage information for the therapeutic classification of drugs. Therefore, we decided to build a suitable framework, called Disease-Drug Ontology (DDO), which can support all above features that are required for implementation of our system. It requires being relatively unrestrictive, having appropriate conceptualization level, and allowing for scalable reasoning. Consequently, our proposed ontology is the first ontology which puts drugs and diseases in one umbrella and makes the relationship between them based on clinical aspects.

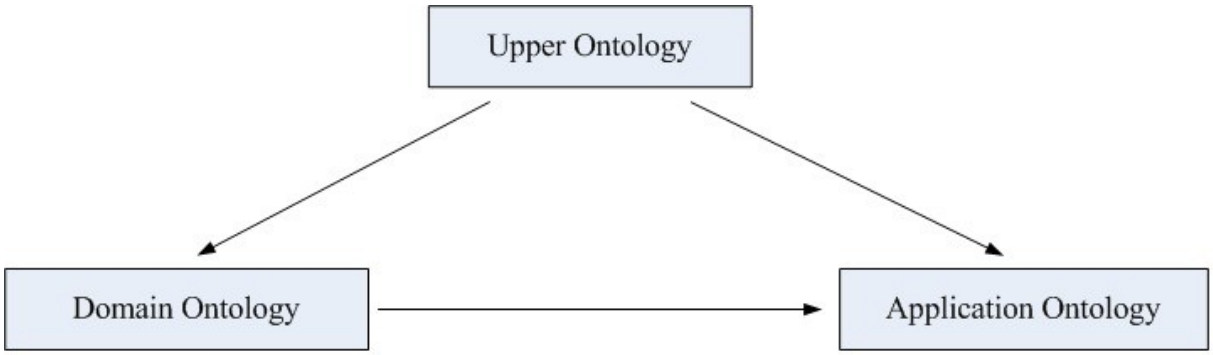


Figure 3.2: Modularization of Ontology Depending on the Scope

We have considered that the key point in ontology development is to reuse knowledge components whenever possible. Thus, previously existing ontology sources were carefully studied to choose pertinent reusable knowledge resources to allow efficient knowledge mapping and sharing among independent data sources. Details for these sources are described in sections 3.3.2 and 3.3.3. The knowledge components from these sources were tested and mapped to our knowledge framework. Therefore, data from these sources is obviously compatible to the DDO for integration to our knowledge base. A

sustained effort is made to maintain high-order hierarchy of DDO to ensure flawless data integration.

3.3.1 Ontology Development Tools

There are over 50 ontology editing environments available to aid ontology development, such as Protégé, Chimaera, OilEd and DAG-Edit [38-40]. Upon surveying and testing, Protégé was selected as the primary tool for developing the OWL framework due to the following reasons: 1) Protégé is an open source, free ontology editor which maintains two key types of modeling ontologies via the Protégé-Frames and Protégé-OWL editors; 2) It provides a wide set of customizable user interface elements which allows easy access, hierarchical tree structure for class browsing, form interface for filling in slot values; 3) It supports several formats including RDF(S), OWL, and XML Schema; 4) Protégé which is based on Java has a great extensibility and scalability with its open modular design, which allows convenient functionality extension by adding or creating plug-ins; 5) Such a plug-and-play environment makes Protégé a flexible base for rapid prototyping and application development; 6) Protégé has been developed and tested for many years with a big group of users in bioinformatics area worldwide and with continuous support commitment [38-41].

To increase the editing capability of Protégé in supporting for ontology merging, mapping, comparison and improving the visualization flexibility, some additional plug-ins are adopted besides what has been packed in Protégé, including:

- Prompt [42]: This plug-in allows to manage several ontologies in Protégé, including comparing versions of same ontology, moving frames between

ontologies, and merging all or extracted portion of ontology into one. More details about Prompt and how is used in building of our system is described in Chapter 4.

- Pellet [43]: It is a reasoner that can be called to check consistency, classify the taxonomy and automatically compute the class hierarchy of ontology. Pellet reasoner is studied further in Chapter 5.

3.3.2 Human Disease Ontology

The first step of our ontological development is to find a good source of clinical knowledge to construct the upper level ontology based on them. After evaluating some known resources such as MeSH (Medical Subject Headings) [44], SNOMED CT(Systematized Nomenclature of Medicine-Clinical Terms) [45], OMIM (Online Mendelian Inheritance in Man) [46], DOID, ICD-10, NCI Thesaurus and UMLS, Human Disease Ontology (DOID) is opted as the source of knowledge of diseases for our ontological design due to the following reasons. Human Disease Ontology is open source ontology for the integration of biomedical data that is related to human diseases. DOID has a formally correct, semantically computable structure. Terms in DOID are well defined, employing standard references. These terms are connected to well-established, well-adopted terminologies and ontologies that contain disease and disease-related concepts such as SNOMED CT, ICD-10 and UMLS. Such a combination of a semantically computable structure and the external references to these sources facilitates useful inference between dissimilar datasets by applying one or more of these standard terminologies to code diseases. This property makes DOID a community-accepted ontology of diseases for clinical research and medicine in the wide range of environmental, genetic and infectious diseases [34, 35, 47]. By containing 8608 classes,

DOID is much larger than MeSH and OMIM and should therefore provide greater disease coverage [34].

As shown in Figure 3.3 diseases in DOID are divided into eight main categories including disease of mental health, disease of metabolism, genetic disease, medical disorder, disease by infectious agent, disease of cellular proliferation, syndrome and disease of anatomical entity. Each of these classes has their own relevant subclasses and this flow continues to make all the disease classes of ontology.

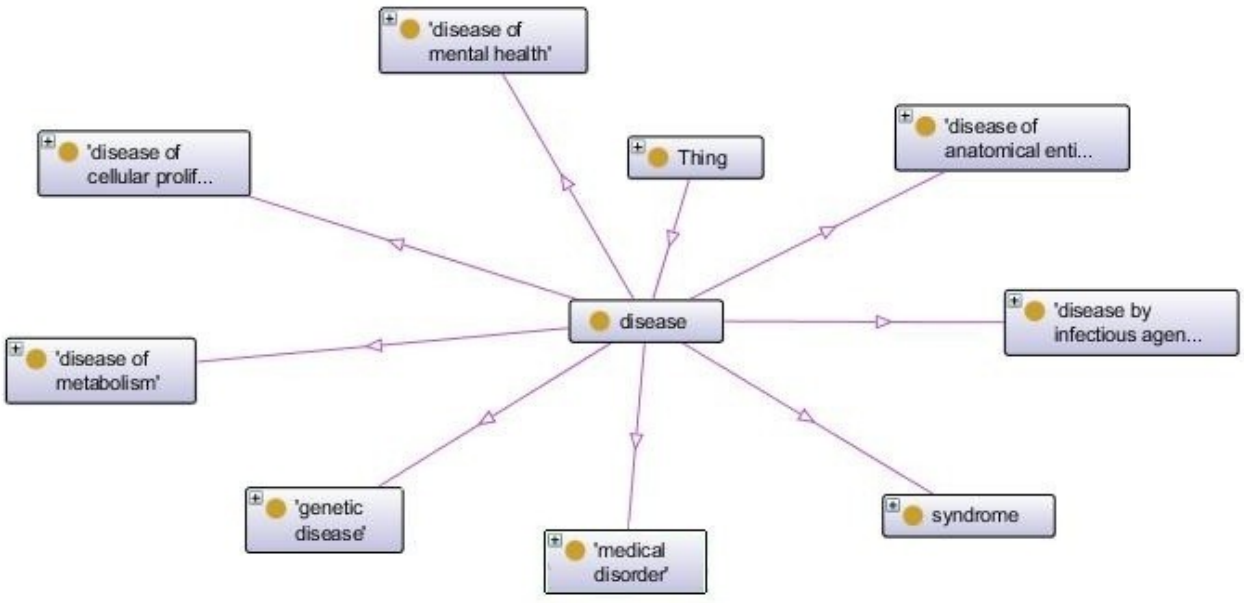


Figure 3.3: Human Disease Ontology (DOID)

Obviously, due to the large size of ontology, we cannot picture all classes of the ontology in a diagram. In Figure 3.4, DOID ontology is shown with one level of its subclasses in which as we can see the class of disease by infectious agent has four subclasses of fungal infectious disease, bacterial infectious disease, parasitic infectious disease and viral infectious disease or as another example the class of disease of mental health has

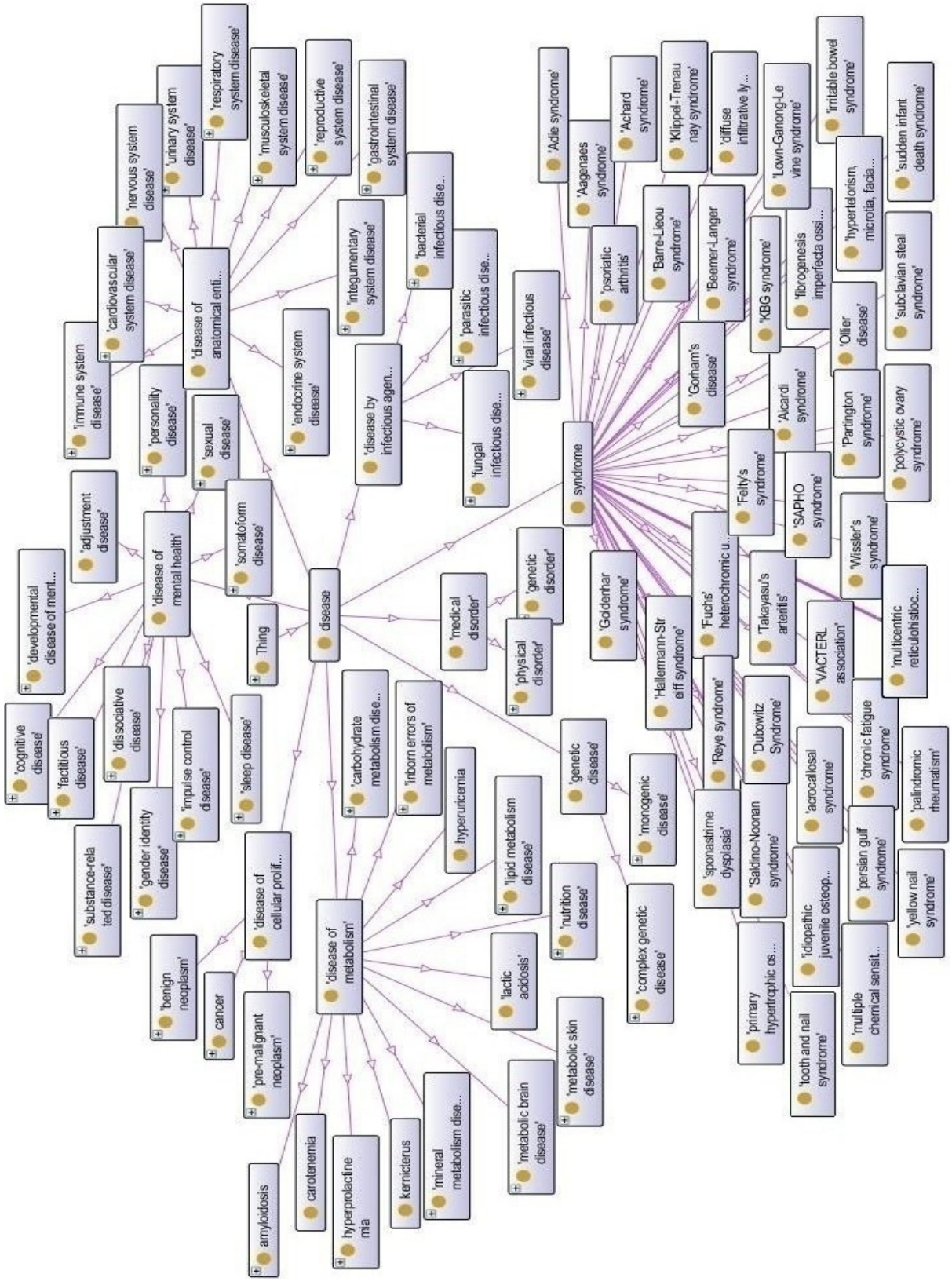


Figure 3.4: Human Disease Ontology with one level of subclasses

twelve subclasses such as personality disease, adjustment disease, developmental disease of mental health, dissociative disease, sleep disease, impulse control disease, gender identity disease, cognitive disease and so on.

In addition to the classes, Human Disease Ontology contains several object properties that are used to define the relationships between classes. A list of these properties is shown in Figure 3.5. Some of these properties such as *is-a* are already in use in the ontology, and some others such as *has-symptom* are defined to be used further in case of mapping with other ontologies. As an example, all of the eight subclasses of disease class are in a relationship with disease class based on is-a relationship.

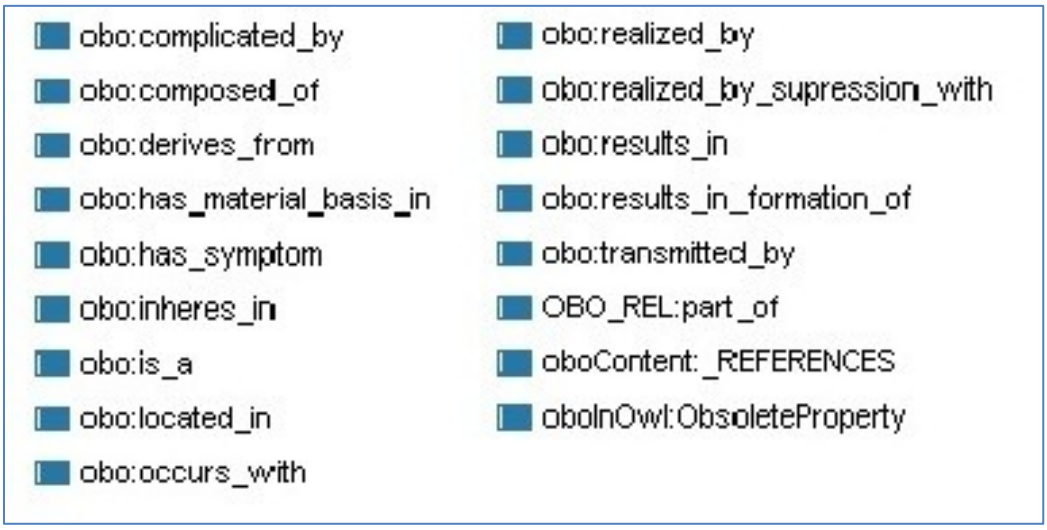


Figure 3.5: List of object properties in Human Disease Ontology

3.3.3 DrugBank Database and Drug Ontology

Building a drug Ontology is the other important part of construction of DDO as an upper level ontology. Since there is no standard ontology in the domain of drugs [29, 48, 49], it was decided to choose a suitable database from which a drug ontology can be built. There

are essentially two kinds of open source online drug databases: 1) Clinically Oriented drug resources such as PharmGKB (Pharmacogenomics Knowledge Base) [50], RxList [51] and 2) Chemically oriented drug databases include KEGG (Kyoto Encyclopedia of Genes and Genomes) [52] and PubChem [53]. DrugBank [54] is also another open source database which is developed and supported by the Departments of Computer Science and Biological Sciences of University of Alberta. It is extended to make a bridge between the clinically and chemically oriented databases. As a clinically oriented drug encyclopedia, DrugBank is able to provide detailed, up-to-date, quantitative, and analytic information regarding drugs, drug targets and the biological and physiological consequences of drug actions. It also contains links to most of bioinformatics and biomedical databases such as PubChem , KEGG and also drug and pharmaceutical databases such as PharmGKB and RxList. The database contains 6711 drug entries including 1447 FDA-approved (US Food and Drug Administration) small molecule drugs, 131 FDA-approved biotech drugs, 85 nutraceuticals and 5080 experimental drugs [55]. Besides, DrugBank also includes some specific data fields such as: drug synonyms, drug brand names, drug-drug interactions, food-drug interactions, prices in USD and dosages, which make the database a perfect choice in building up the desired drug ontology. In order to employ DrugBank database in construction of DDO, modification of the database into appropriate ontology is required. For this purpose, the XML format of DrugBank database is used to convert it into ontology. Figure 3.6 shows part of Drug Bank database XML file for Insulin Glargine Drug.

In order to transform DrugBank database into ontology, the XML-tab that is a plug-in in Protégé is applied. XML-tab allows importing XML file into Protégé and save it as

ontology in OWL format [56]. As a result after importing the DrugBank database XML file in Protégé, the drug ontology becomes ready for use in the construction of an upper level ontology.

Figure 3.7 displays the drug ontology that contains sixteen classes including: drug, manufacturers, packagers, ahfs-codes, dosages, categories, groups, synonyms, food-interactions, mixtures, external-identifiers, external-links, brands, drug-interactions, patents and prices. All these classes are subclasses of Thing class which is the super class in all OWL ontologies.

```
<drug type="biotech" updated="2010-10-16 15:58:24 -0600" created="2005-06-13 07:24:05 -0600" version="3.0">
  <drugbank-id>DB00047</drugbank-id>
  <name>Insulin Glargine</name>
  <description>Insulin glargine is produced by recombinant DNA technology using a non-pathogenic laboratory strain
  of Escherichia coli (K12) as the production organism. Small amounts of insulin glargine are slowly released from microprecipitates giving
  the drug a long duration of action and no pronounced peak concentration. </description>
 <indication>For the treatment of Type 1 or 2 diabetes mellitus in patients over 17 years old who require a long-acting (basal) insulin
 for the control of hyperglycemia. May be used in pediatric patients with Type 1 diabetes mellitus who require a long-acting (basal) insulin
 for glycemic control.  </indication>
  <toxicity>Inappropriately high dosages relative to food intake and/or energy expenditure may result in severe and sometimes prolonged and
  life-threatening hypoglycemia. Neurogenic (autonomic) signs and symptoms of hypoglycemia include trembling, palpitations, sweating, anxiety,
  hunger, nausea and tingling. Neuroglycopenic signs and symptoms of hypoglycemia include difficulty concentrating, lethargy/weakness, confusion,
  drowsiness, vision changes, difficulty speaking, headache, and dizziness. Mild hypoglycemia is characterized by the presence of autonomic symptoms.
  Moderate hypoglycemia is characterized by the presence of autonomic and neuroglycopenic symptoms. Individuals may become unconscious in severe
  cases of hypoglycemia. Other adverse events that may occur include allergic reaction, injection site reaction, lipodystrophy, pruritis, and rash.</toxicity>
  <brands>
   <brand>Lantus</brand>
   <brand>Lantus SoloStar</brand>
  </brands>
  <packagers>
   <packager>
    <name>Gruppo Lepetit SPA</name>
    <url></url>
   </packager>
   <packager>
    <name>Physicians Total Care Inc.</name>
    <url>http://www.physicianstotalcare.com</url>
   </packager>
   <packager>
    <name>Sanofi-Aventis Inc.</name>
    <url>http://www.sanofi-aventis.com</url>
   </packager>
  </packagers>
  <manufacturers>
   <manufacturer generic="false">Sanofi aventis us llc</manufacturer>
  </manufacturers>
```

Figure 3.6: Portion of Drug Bank Database XML File

The drug class is the center class of the ontology that all other fifteen classes are in a relationship with through relevant object properties. In Drug ontology some of the information such as DrugBank-id or name of each drug is demonstrated as a Data type property of that drug. Therefore Drug ontology contains some data type properties, as well. The list of all data type and object properties of Drug ontology is shown in Figures 3.8 and 3.9.

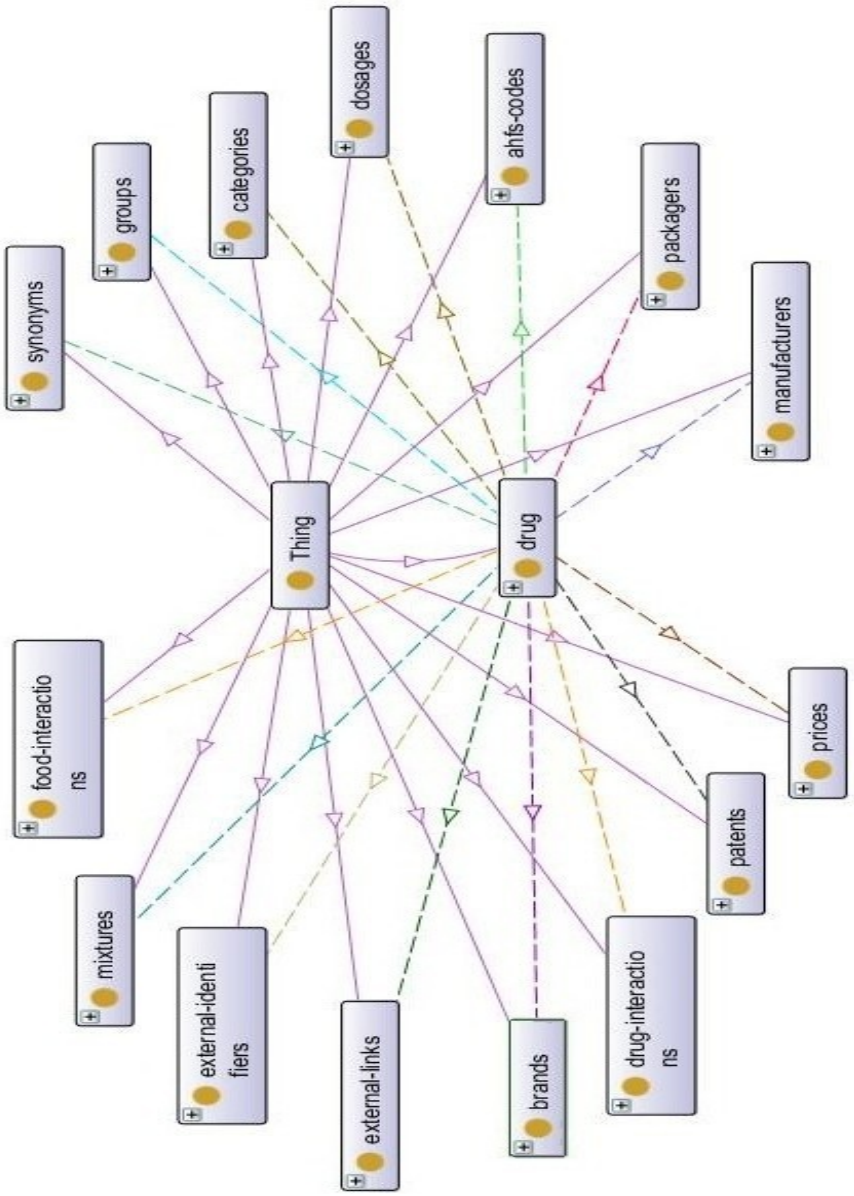


Figure 3.7: Drug Ontology

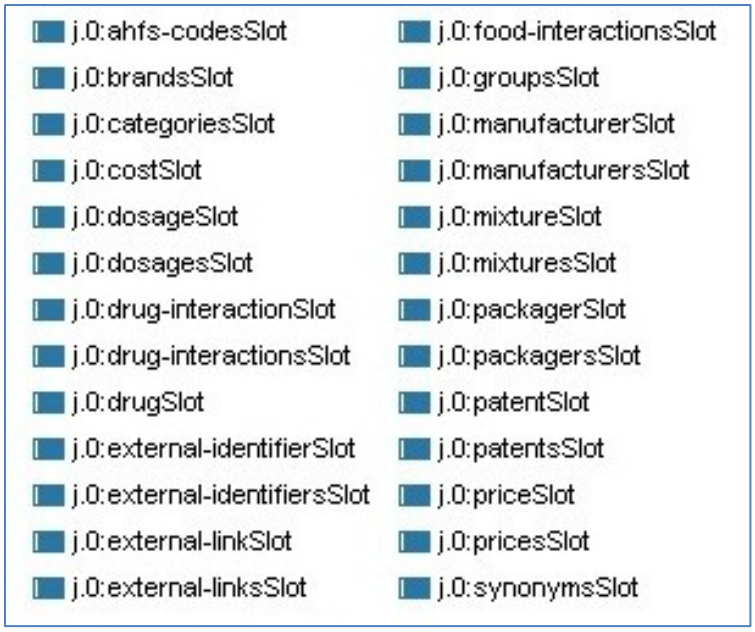


Figure 3.8: List of Object Properties in Drug Ontology

Figure 3.9: List of Data Type Properties in Drug Ontology

3.3.4 Disease-Drug Ontology

Since none of the Drug or Human Disease Ontologies, individually, can meet our desired requirements, there is a need to employ a method to link these two main ontologies

together and construct an upper level Disease-Drug Ontology. Ontology matching is the method that is opted to make the relation between the two ontologies. Ontology matching and its relevant concepts are fully discussed in the Chapter 4 of this thesis.

After extracting the Drug Ontology into the Human Disease Ontology, there is a need that the class of Drug is defined disjointed from the class of Disease. Making two classes disjointed from each other clarifies that an object cannot be an instance of more than one of these classes. The Disjoint Class button in Protégé editor is applied to specify the disjoint classes. Next step in construction of Disease-Drug Ontology is to define relationships between the diseases and their relevant drugs. In order to form a relationship among Drug and Disease classes, appropriate object properties are required to be selected. There is no such object property available among the properties of Drug and Human Disease Ontologies, hence two object properties of *may treat* and *may be treated by* are created to establish the relationship between Disease and Drug classes. These two object properties are defined as inverse of each other which means the domain of one is the range of the other one. Consequently, the domain of *may treat* property is restricted to the Drug class and its range is limited to the Disease class of DDO, while the domain and range of *may be treated by* property is defined vice versa. Figure 3.10 displays a diagram of Disease-Drug Ontology and the added object properties.

Using the two object properties of *may treat* and *may be treated by*; the relevant drugs of each specific disease can be defined. As a prototype in this thesis, the relationship between eleven diseases and their relevant drugs which are seventeen drugs has been established. For instance, six following drugs are considered for Coronary Heart Disease in DDO including: Acetylsalicylic acid (ASA), Clopidogrel, Heparin, Metoprolol,

Perindopril and Lovastatin. The list of all these eleven diseases and their medications are summarized in Table 3.1.

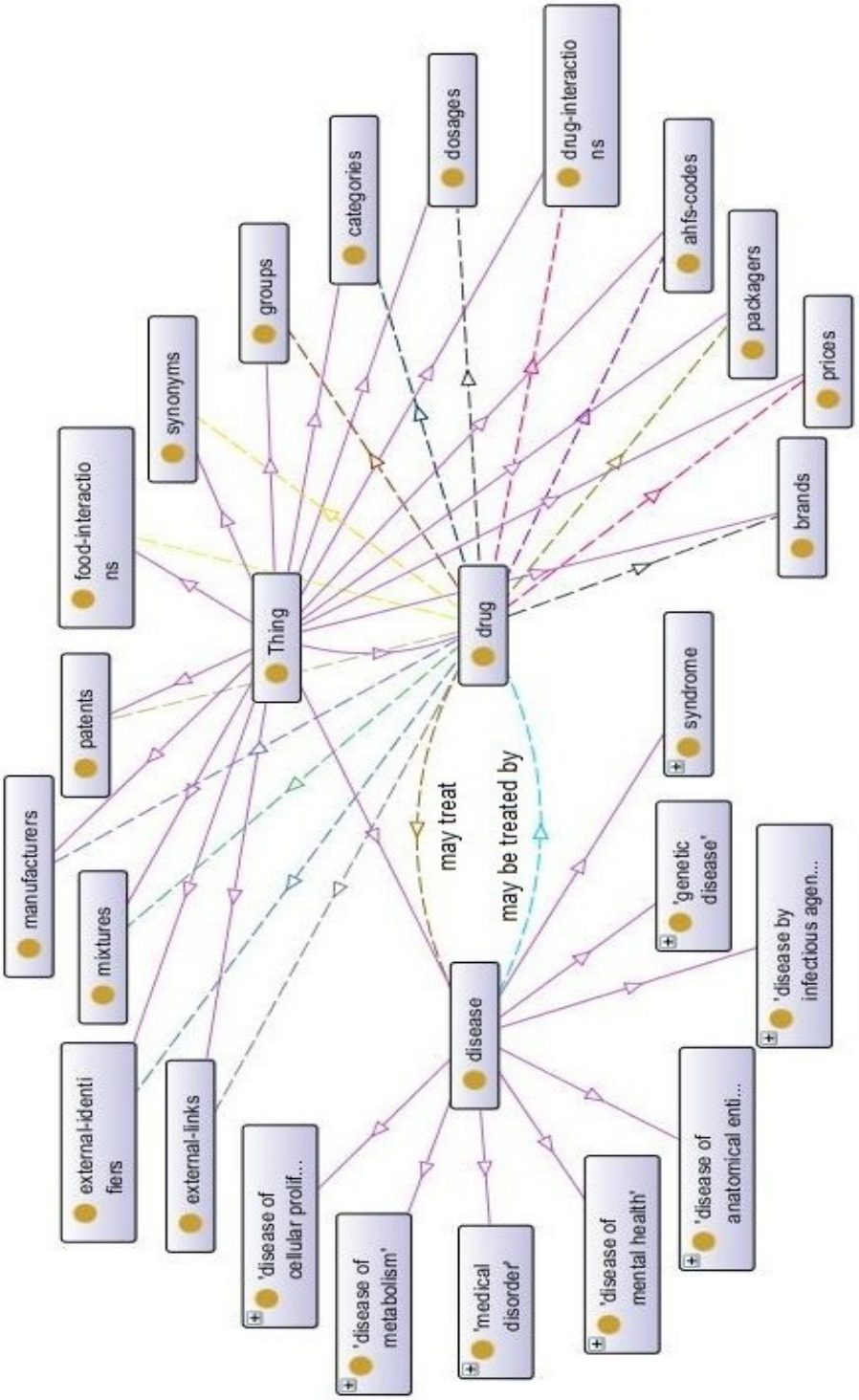


Figure 3.10: Disease-Drug Ontology

Table 3.1: List of Diseases and their Relevant Drugs in DDO

Disease	Disease ID	Drug	DrugBank ID	Drug ID
Acromegaly	DOID_2449	Somatropin recombinant	DB00052	Drug_317
Otitis media	DOID_10754	Azithromycin	DB00207	Drug_348
Esophagitis	DOID_11963	Pantoprazole	DB00213	Drug_678
Coronary heart disease	DOID_3393	Acetylsalicylic acid	DB00945	Drug_60
		Clopidogrel	DB00758	Drug_285
		Heparin	DB01109	Drug_701
		Metoprolol	DB00264	Drug_546
		Perindopril	DB00790	Drug_585
		Lovastatin	DB00227	Drug_648
Pulmonary hypertension	DOID_6432	Treprostinil	DB00374	Drug_560
Nephrotic syndrome	DOID_1184	Triamterene	DB00384	Drug_354
Gastroesophageal reflux disease	DOID_8534	Cimetidine	DB00501	Drug_444
Diabetes mellitus type 1	DOID_9744	Insulin Glargine	DB00047	Drug_458
Diabetes mellitus type 2	DOID_9352	Insulin Glargine	DB00047	Drug_458
		Glibenclamide	DB01016	Drug_718
		Metformin	DB00331	Drug_640
Lemierre's syndrome	DOID_11337	Drotrecogin alfa	DB00055	Drug_88
Human immunodeficiency virus infectious disease	DOID_526	Amprenavir	DB00701	Drug_117

3.3.5 Patient Ontology

The Patient Ontology that is offered in this thesis is built based on some materials from Meditch (Medical Information Technology, Inc.) documents [57, 58]. Meditech's software and information systems are installed in health care organizations around the world; therefore their resources are reasonably trustworthy. The proposed Patient Ontology can be applied to compare physician's decisions in similar patient cases. As it is shown in Figure 3.11 Patient Ontology contains four super classes including: Personal Information, Patient History, Patient Allergies, and Disorder class. The Personal

Information class consists of seventeen subclasses that are all related to the personal information of patients such as first name, last name, address, birth date, blood type and so on. Any diagnosis or injuries, family or hospitalization history is gathered in the six subclasses of Patient History class. The Patient Allergies class contains four subclasses that specify the allergic reaction, severity, status and type of allergy of the patient. The condition, date of diagnosis, disease's type or name, symptoms, treatment and any other similar information is included in the eight subclasses of Disorder class.

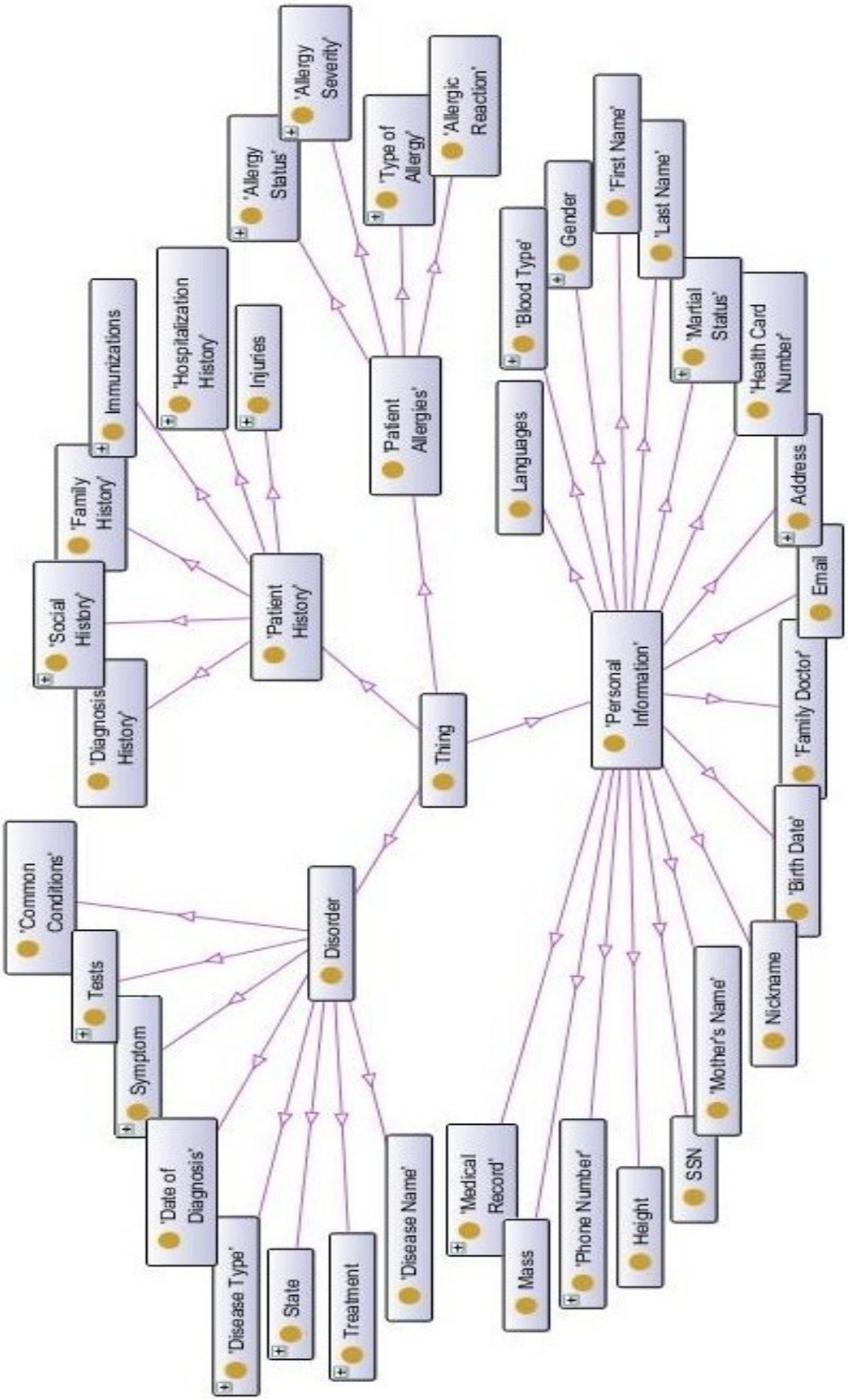
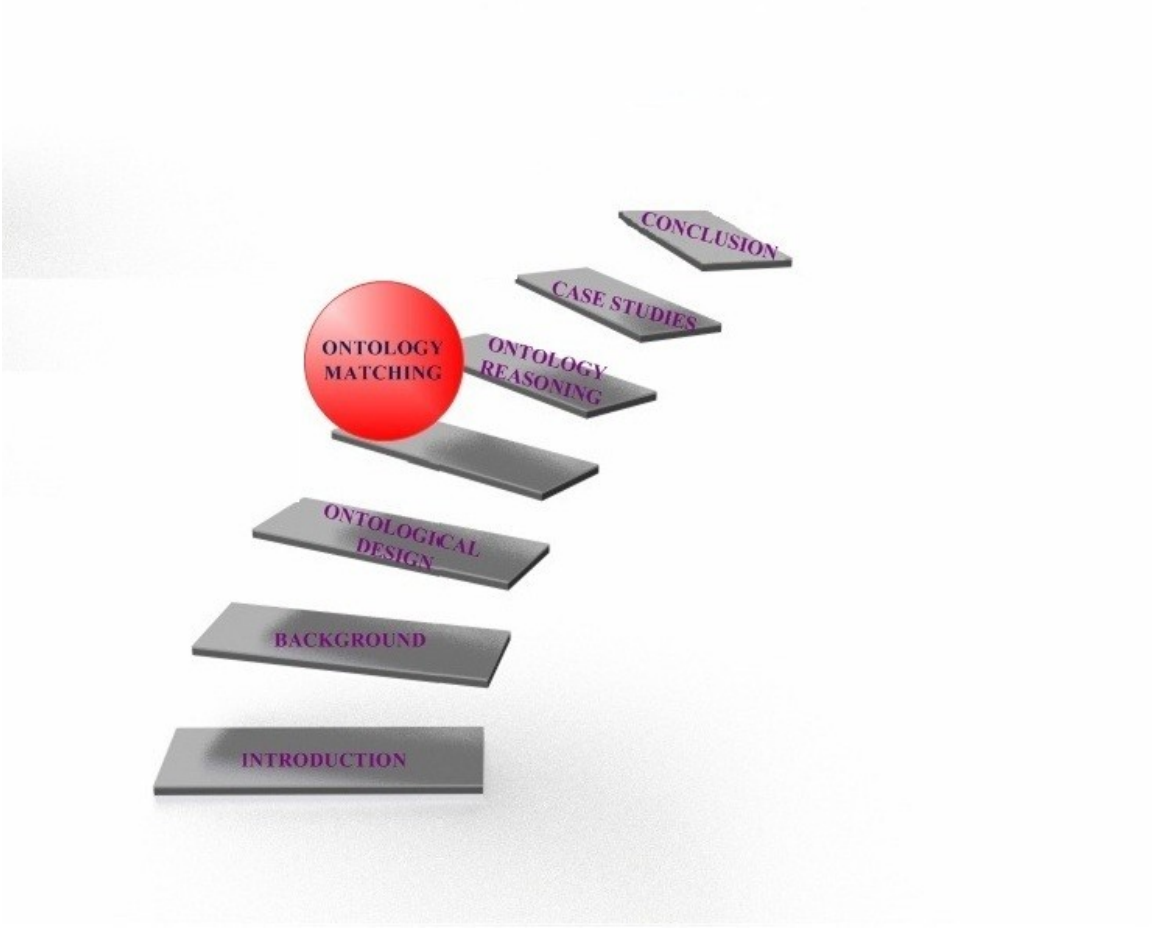


Figure 3.11: Patient Ontology

CHAPTER 4
ONTOLOGY MATCHING



This Chapter describes the concept of ontology matching and its related notions as a solution to the semantic heterogeneity problem faced by ontology models. It first presents the motivation and definition of the ontology matching problem, and then overviews various classifications of matching techniques. Finally, it introduces one of the common frameworks that is available for matching and translating between different knowledge representations and explains how matching techniques are applied in construction and development of the proposed Disease-Drug Ontology (DDO).

4.1 Ontology Matching

Since ontologies typically provide a vocabulary describing one domain of interest, one ontology is not enough to support most of the tasks in reality, and there is a need for applications to use variety of ontologies together [59]. However, linking the ontologies is not as simple as it may look. As each ontology defines its own set of concepts and relations, interoperability issues arise when exchanging information among heterogeneous ontologies [60-61]. In order to overcome this heterogeneity among ontologies and make them understand each other, a context of ontology engineering has been proposed. Since ontology engineering has to deal with multiple and distributed ontologies, it needs support of ontology matching.

During study of ontology matching, we may confront with some other related concepts such as correspondence, alignment, mapping, and merging. Since in this area of ontology matching, different authors use different words to refer to similar concepts, and, vice versa, sometimes different concepts are referred to by the same name, it may seem quite

useful to get familiar with the general meaning and definition of each concept before going into the details.

Correspondence is the relation holding, or supposed to hold according to a particular matching algorithm or individual, between entities of different ontologies. These entities can be as different as classes, individuals, properties or formulas. Some authors use the term mapping instead [62]. Based on the terminology defined by J. Euzenat, correspondence definition can be indicated as the following [63]. *A correspondence between an entity e belonging to ontology o and an entity e' belonging to ontology o' is a 5-tuple <id, e, e', r, conf> where:*

- *id is a unique identifier of the correspondence,*
- *e and e' are the entities (e.g., properties, classes, and individuals) of o and o', respectively,*
- *r is a relation such as "equivalence," "more general," "disjointness," "overlapping," holding between the entities e and e' , and*
- *conf is a confidence measure (typically in the [0,1] range) holding for the correspondence between the entities e and e'.*

The next key concept in the studying of ontology matching, called alignment, which is a set of correspondences between two or more (in case of multiple matching) ontologies. Basically, the alignment is the output of the matching process [64]. In other words, *an alignment of ontologies o and o' is a set of correspondences between entities of o and o'* [63]. Alignments can be used for various tasks such as ontology merging, query answering, data translation or for browsing the semantic web.

Mapping is the other notion in the area of ontology matching which is mainly oriented or directed version of an alignment. It maps the entities of one ontology to at most one entity of another ontology [60].

Ultimately, ontology merging is the creation of a new ontology from two, possibly overlapping, source ontologies. In this process the initial ontologies remain unaltered and the merged ontology is supposed to contain the knowledge of the initial ontology [65].

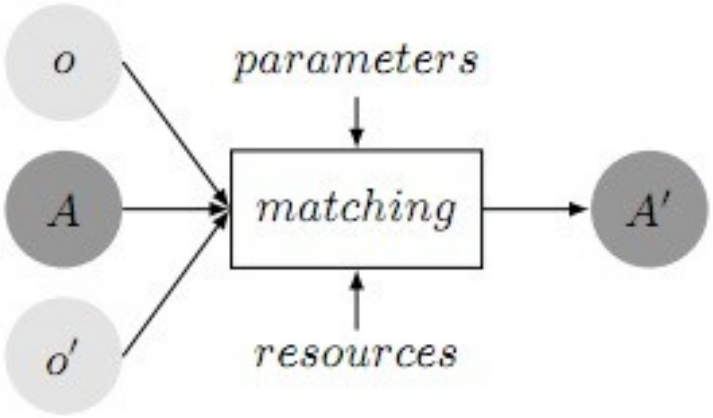


Figure 4.1: The Matching Process

Ontology matching is the process of finding relationships or correspondences between semantically related entities of different ontologies [63]. These correspondences may stand for equivalence as well as other relations such as consequence, subsumption or disjointness between ontology entities [64]. Ontology entities usually denote the named entities of ontologies such as classes, properties or individuals. However, these entities can also be more complex expressions, such as formulas, concept definitions, queries or term building expressions [62, 66]. The matching process can be seen as *a function f*

which takes two ontologies o and o' , an input alignment A, a set of parameters p, and a set of oracles and resources r, and returns an alignment A' between o and o' [63]:

$$A' = f(o, o', A, p, r)$$

The matching process is schematically represented in Figure 4.1.

In this thesis ontology matching is applied to find the correspondences between entities of Drug Ontology and Human Disease Ontology. This process is fully described in section 4.3.

4.2 Matching Techniques Classification

There are various techniques for computing the matching process and there is still much work going on in finding better methods. Many different matching solutions have been proposed so far from various viewpoints such as databases, information systems and artificial intelligence. They take advantage of various properties of ontologies such as structures, data instances, semantics, or labels, and use techniques from different fields such as statistics and data analysis, machine learning, automated reasoning, and linguistics. These solutions share some techniques and tackle similar problems, but differ in the way they combine and exploit their results [62, 63, 64, 66]. Basically, matching techniques are categorized in two main groups of Element-level and Structure-level. In the following some of the most used techniques are classified.

4.2.1 Element-level

Element-level matching techniques calculate correspondences by analyzing entities or instances of those entities separately and do not pay attention to their relations with other entities or instances. Particularly, they are concerned about the entities or their instances

in isolation from their relations with other entities or their instances [61, 64]. Some of the known techniques that are based on Element-level are: String-based techniques, Language-based techniques, and Alignment reuse that are briefly described in the following.

String-based techniques can be applied to compute string similarity between the ontology entities. The string may represent the names, name descriptions, the label or the comments of entities and treated as sequence of letters.

String-based methods are normally based on the following intuition: the more similar the strings, the more likely they are to indicate the same concepts. String methods may use some functions to show the distance between a pair of strings, this distance is usually shown as a real number. The smaller value of the real number shows a greater similarity between the strings. String-based methods can easily find similar classes such as book and textbook but typically they will not be able to find book and volume as the similar classes. There are several software packages for computing string distances such as Simetrics1, Prompt, the Alignment API3 and SimPack4 [62-63].

Language-based techniques consider names as words in some natural language such as English. They take advantage of natural language processing techniques to help extract the meaningful terms from a text and based on that find the similarity between two strings as meaningful pieces of text rather than sequences of characters. Some of these methods may use the external resources, such as dictionaries, lexicons, thesauri to compute the similarities between terms. Perl package WordNet and the Java package SimPack apply these techniques to find the existing similarities [62-64].

Alignment reuse techniques apply an alternative way of using external resources, which record alignments of previously matched ontologies. For instance, when we need to match ontology o and o", we use the given alignments between o and o' as an external resource to be available between o and o". This method is based on this idea that most of ontologies to be matched are similar to already matched ontologies, especially if they are in the same application domain. COMA++ applies alignment reuse to find the similarities between entities [63].

4.2.2 Structure-level

Compared with element-level techniques, structure-level techniques are concerned about the relations between the entities and their instances with other entities and their instances in the process of computing the correspondences. In fact, structure-level matching techniques compute correspondences by considering how entities or their instances appear together in a structure [61, 64]. Graph-based techniques, Taxonomy-based techniques, and Model-based techniques are some of known methods that are based on Structure-level techniques. They are explained in brief in the following paragraphs.

Graph-based techniques are based on relational structure which allows all the relations between entities to be taken account. Ontology can be considered to be a graph whose edges are labeled by relation names. Finding the correspondences between elements of such graphs corresponds to solving a form of the graph homomorphism problem. Usually, the similarity comparison between a pair of nodes from the two ontologies is based on the analysis of their positions in the graphs. The idea behind this is that, if two nodes from two ontologies are similar, their neighbors must also be somehow similar.

Kang&Naughton can be named as an example of a system that applies this technique [63].

Taxonomy-based techniques are also based on graph algorithms; however consider only the specialization relation. Taxonomic techniques idea is that *is-a* links connect concepts that are already similar (being taken as a subset or superset of each other), therefore their neighbors may be also somehow similar. Prompt and OLA are two systems that use this technique [62-63].

Model-based algorithms manage the input based on its semantic interpretation. The idea is that if two entities are the same, then they share the same interpretations. They are deductive methods; hence they do not perform very well alone for an essentially inductive task like ontology matching. An important challenge of these techniques is their combination with inductive techniques. S-Match and CtxMatch use the semantic methods to find similar correspondences [62-64].

Prompt Suite is applied in this work to perform the matching process employs string-based techniques to find the correspondences between the entities of Drug and Human Disease Ontology. This is explained in detail in section 4.3.

4.3 Prompt Suite

In this section Prompt Suite is introduced. This Suite is applied in this thesis to perform ontology matching. Among other available tools and environments dealing with ontology matching, Prompt is selected due to a good quality of its suggestions [42, 65, 67]. Based on the research that was done [42, 65, 67], human experts accepted 90% of Prompt's suggestions and 75% of the conflict-resolution strategies that Prompt proposed.

The Prompt Suite is an interactive framework for comparing, matching, merging, maintaining versions, and translating between different knowledge representations. It is introduced as a plug-in for Protégé. Prompt allows managing multiple ontologies in Protégé. Prompt takes two ontologies as input and leads the user in the construction of combined ontology as output. In the first step, Prompt computes an initial set of matches based on lexical similarity between class names which is based on string-based technique. The rest of the process go on the following cycle: First, user picks an operation to perform, either by selecting one from the Prompt suggestion list or specifies the desired operation directly via ontology-editing environment. Then, Prompt performs the operation chosen by user. It automatically identifies inconsistencies, such as name conflicts, redundancy in the class hierarchy, and suggests possible strategies to resolve them. Finally, it generates a list of suggestions for users [42, 63, 65, 67]. Figure 4.2 shows the flow of Prompt Algorithm, in which the pink box indicates the action that is performed by the user and the blue boxes refer to actions that are executed by Prompt.

In this thesis the extract option of Prompt Tab is used to create the upper level Disease-Drug Ontology from the two main Drug and Human Disease Ontologies. Extract option which is based on ontology matching, allows us to move desired part of drug ontology to our disease ontology which can lead to reduce the size of ontology and increase the efficiency of the final result; however the information that are not transferred can be added to the system any time that is required. By matching each class of Drug Ontology into the Human Disease Ontology, all the properties and slots that are related to that class are also transferred to the ontology.

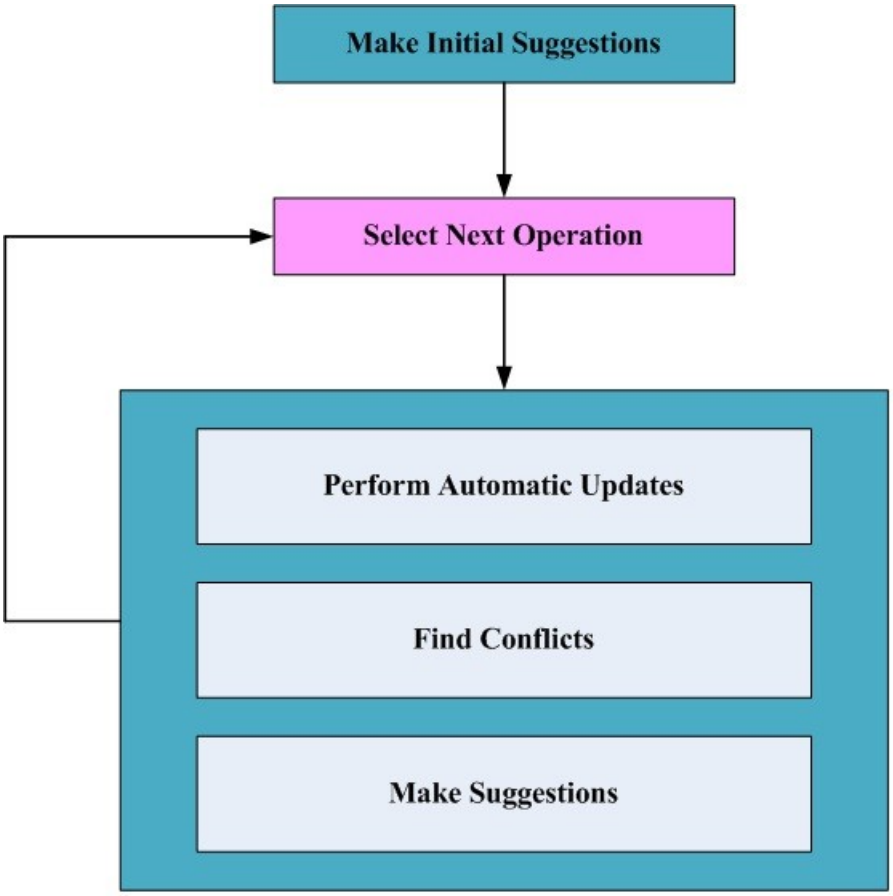
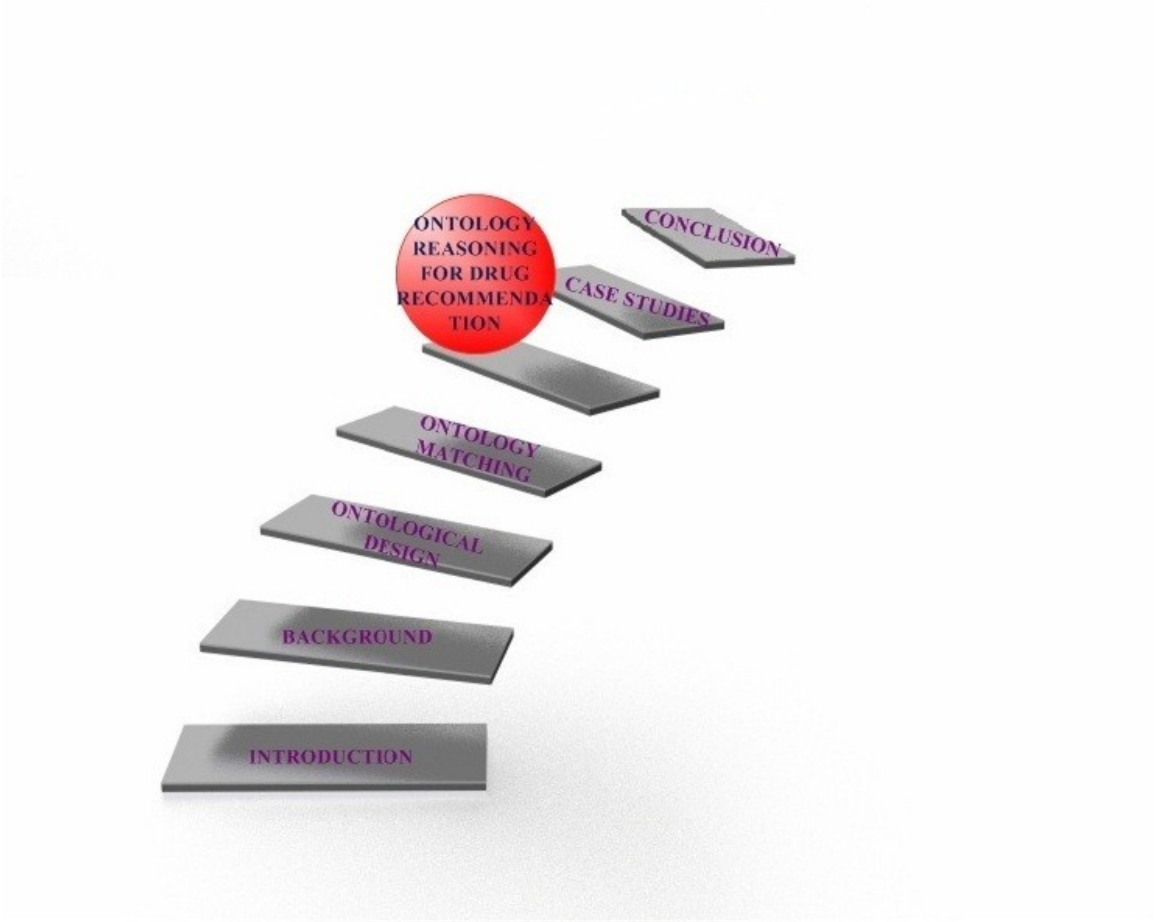


Figure 4.2: The flow of Prompt Algorithm

CHAPTER 5
ONTOLOGY REASONING FOR DRUG RECOMMENDATION



This Chapter describes OWL-DL reasoning as well as rules and ontology reasoning. It first presents the process of reconciliation and validation of DDO via a Description Logic Reasoner. Afterward, it describes the notion of ontology crawler and how it provides physicians by direct queries from DDO to facilitate the decision making for correct drug recommendation. This Chapter comes to an end with the discussion about the rule-based inference engine and the relevant rules that are required to infer the correct results for the drug recommendation system.

5.1 Pellet Reasoning

Like any Semantic Web framework, the DDO involves knowledge components from multiple sources which may introduce inconsistencies that demands reconciliation and validation [68]. The DDO is formalized in OWL-DL which has foundation for reasoning based on Description Logics. Thus, it is feasible to perform automated reasoning over the ontology using a Description Logic reasoner [29]. A Description Logic reasoner presents a variety of inference services. One of the main services offered by a Description Logic reasoner is to determine whether or not a class is consistent. A class is called inconsistent if it cannot possibly have any instances. Ability to compute inferred instances is another standard service that is offered by a reasoner. Furthermore, a Description Logic reasoner is capable, for instance, to check whether or not a class is a subclass of another. By performing such a test on the classes of ontology, a reasoner would be able to compute the inferred ontology class hierarchy. Being able to use a reasoner to automatically compute the class hierarchy is one of the major benefits of building an ontology using the OWL-DL sub-language. Indeed, in construction of very large ontologies with several

thousand classes, the use of a reasoner to compute subclass-super class relationships between classes becomes vital. Without a reasoner it is very difficult to keep large ontologies in a maintainable and logically correct state. Computing and maintaining multiple inheritances are also done by the reasoner. This technique helps to keep the ontology in a maintainable and modular state. This does not only promote the reuse of the ontology by other ontologies and applications, it also minimizes human errors that are inherent in maintaining a multiple inheritance hierarchy [29, 69, 70].

In this thesis, Pellet 1.5.2 is utilized to validate the ontology framework that is generated. Pellet reasoner is an open-source Java based OWL-DL reasoner that can be used in conjunction with both Jena and OWL API Libraries. For systems applying OWL to represent information, Pellet is the leading choice where sound and complete OWL-DL reasoning is essential [29, 43, 71]. In this study Pellet is used as a direct reasoner which can be set up as an additional plug-in for Protégé 3.4 and be invoked from Protégé by calling directly the reasoner API. The Pellet uses the methods of *Check Consistency, Classify Taxonomy, and Compute Inferred Types* to evaluate the ontology. The result of employing Pellet reasoner indicates that DDO is consistent and computes its inferred types. However, four classes are detected to have more than one super class in DDO hierarchy. We have examined and resolved each of them by eliminating redundancies or redefining restrictions. Therefore, after reclassifying and applying the inferred hierarchy suggestions, the general processing time for Pellet reasoner to process the DDO takes less than 163 seconds which is shown in Figure 5.1. Green dots denote the items that are checked and validated.

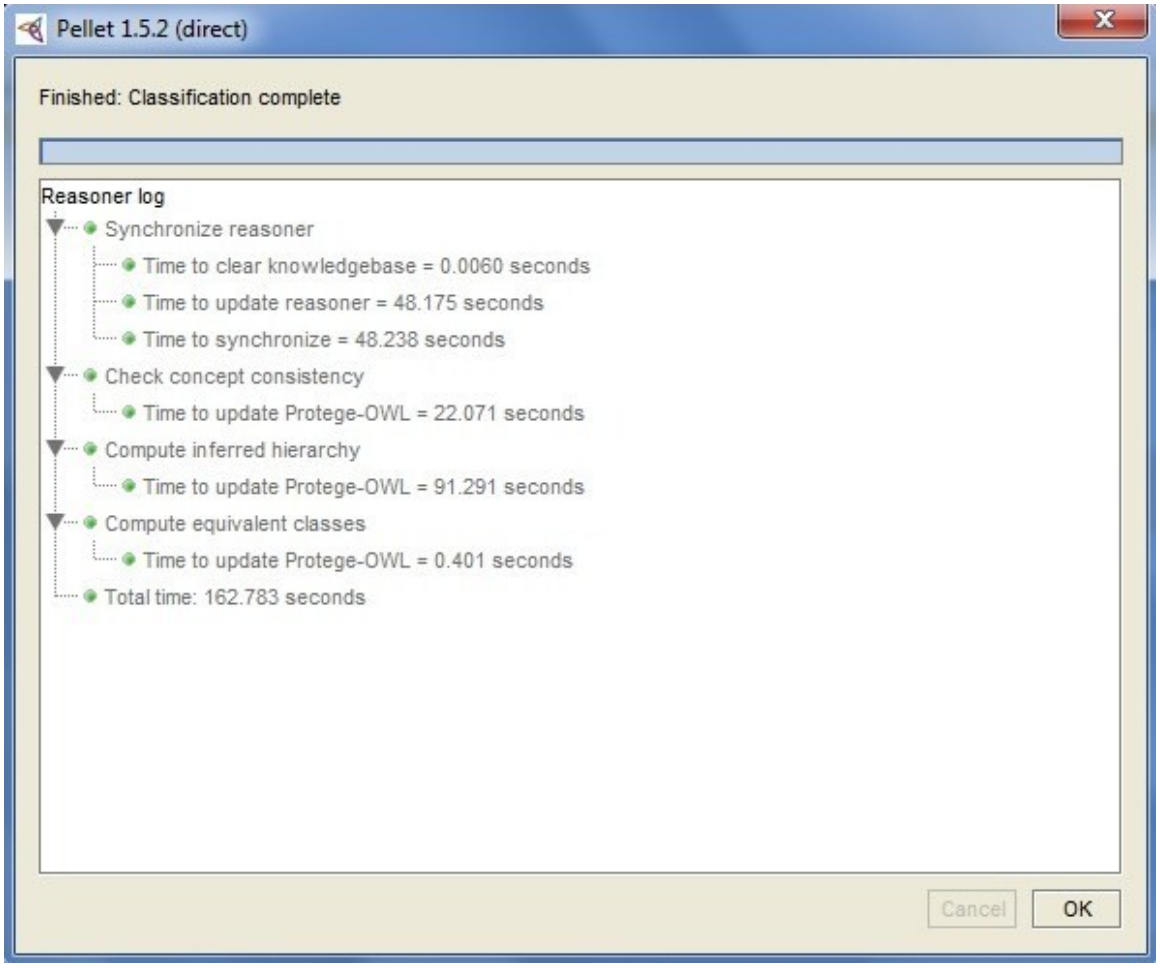


Figure 5.1: OWL-DL Reasoning Log of Finalized DDO by Pellet Reasoner

5.2 Disease-Drug Ontology Crawler

Disease-Drug Ontology Crawler provides physicians with essential queries to choose pertinent drugs and make a suitable decision for drug recommendation. DDO Crawler is comprised of DDO itself, and Jena API which is a Java framework for developing semantic web application. Jena Framework is a widespread open-source project and stable API that has been broadly applied in an extensive range of semantic web applications. It includes an Ontology API for handling OWL and RDFS ontologies and

also a query engine in order to support direct queries from ontologies. Furthermore, it provides a wide variety of Java Libraries to aid developers with writing Java codes that handle ontologies [72-74]. In this thesis Jena is employed as a query engine to read the Disease-Drug Ontology and to act as Ontology API between DDO and Java framework. Eclipse SDK 3.3.1 is used as the Java development platform.

Initially, in Jena all the information are encoded as RDF triples and stored in the RDF Model. Thus, in order to work with ontologies that are defined in OWL, Jena as an Ontology API offers the concept of ontology model which is an extended version of Jena RDF model. Ontology model provides additional capabilities for handling ontologies such as adding extra supports for some sort of concepts expected to be in an ontology including classes, properties and individuals. Ontology models are built through Jena *ModelFactory* class. The default setting of creating ontology model is defined for ontologies in OWL-Full. Therefore, in order to create an ontology model for ontologies that are constructed in other languages, *OntModelSpec* object is also required to be applied. This specification allows complete control over the configuration choices for the ontology model [72-74]. The following code creates an ontology model with specifications of OWL-DL language and in-memory storage that is used in this thesis.

```
OntModel m = ModelFactory.createOntologyModel(OntModelSpec.OWL_DL_MEM);
```

Each ontology model has a correlated *document manager* which supports for the processing and handling of ontology documents. Besides, the *read* method is applied to load an ontology document into an ontology model. Several variants can be defined on

read to handle different sources of document such as URL, local location or an input stream. The DDO is loaded into the ontology model through following codes which indicates that is located locally on disk.

```
m.getDocumentManager().addAltEntry(null,"c:/Namira/Thesis/DDO/DDO.owl");
m.read("file:/C:/Namira/Thesis/DDO/DDO.owl") ;
```

After loading DDO into the ontology model, understanding of classes and its related concepts in the Ontology API is required to aid programming of desired queries. All of the classes in the Ontology API that stand for ontology values share *OntResource* as a common super class. Therefore, all shared functionality for such classes can be placed in *OntResource.* Since the Java interface *OntResource* extends Jena's RDF Resource interface, any general method that accepts a resource will accept an *ontResource,* and consequently, any other ontology value as well. *Label, sameAs* and *differentFrom* can be named as some of common attributes of ontology resources. These attributes can be expressed through methods on *OntResource*. Some of standard pattern of these methods are including: *add*, *set, list, get, has* and *remove*. For example *getLabel("")* returns that human-readable label of the respective class or the values of a named property can be listed with *listPropertyValues( ).* Besides, each simple class is represented in Jena by an *OntClass* object. In other words, Jena converts Ontology classes into objects of type *OntClass.* Thus *OntClass* allows accessing all the information in an ontology class. Once the ontology class object is defined, it can be processed through the methods defined on *OntClass*. The properties of a class are handled through similar methods that are mentioned on *OntResource.* Some properties of classes that are expressed in this way are:

subClass, superClass, equivalentClass, disjointWith and instances. As an example we can print a list of instances of Drug Class of DDO as follows (cls is an Ontclass object that points to Drug Class):

```
protected void getDrugInstance(OntClass cls){
ExtendedIterator<? extends OntResource> instances = cls.listInstances();
       while(instances.hasNext()) {
            System.out.println("Instances of Drug Class = " +
             instances.next().getLocalName());
                        }
}
```

Our system considers 9 following inquiries to query DDO :

```
1 - Get Drugs of Diseases Query
2 - Get Brands of Drugs Query
3 - Get Drug interactions of Drugs Query
4 - Get Food interactions of Drugs Query
5 - Get Synonyms of Drugs Query
6 - Get Side effects and Toxicity of Drugs Query
7 - Get Indication of Drugs Query
8 - Get Prices of Drugs Query
9 - Get Manufacturers of Drugs Query
```

These queries are programmed in Java framework and the information to respond to these queries are retrieved from DDO through Jena API. These direct queries can act as a significant aid for physicians to make more precise and appropriate decision to recommend drugs for diseases. Besides, the physicians can be informed about the current

prices, brands and manufacturers of Drugs by performing the relevant queries. The results of the queries are indicated in Chapter 6 of this thesis.

5.3 Rule Engine Reasoning

The Semantic Recommender System is the fundamental system component of our proposed model. This module is composed of a unique rule-based inference engine employing Drug Rules in conjunction with Disease-Drug Ontology and relevant medical facts. The rule engine produces the inference process to offer the clinicians proper consultations. In our proposed model, the knowledge in the DDO and other pertinent medical facts that may be created or updated by physicians are transformed into Drug Rules to feed the rule engine. Besides, the rule engine receives patient data from patient database; afterwards, it proposes a drug recommendation based on available data and rules. The drug recommendation includes generic name of drugs, proper doses based on the patient's conditions and warning about any interaction between the recommended medication and the ones that patient is using.

In this thesis OpenRules rule engine is used to carry out the inference process and lead to the drug recommendation. OpenRules provides an open source rule engine for rules-based web application development. It is mainly based on widespread used tools including MS Excel and Eclipse IDE and is created to maintain complex decision support systems which proficiently execute various set of rules and methods. In contrast with other rule engines, OpenRules proposes to apply Excel directly as the Rules Repository and Management Tool. Such a specification allows non technical users, who may not have any knowledge of coding, such as physicians, to be able to modify or update the

rules directly in Excel after the initial implementation of the system. Furthermore, OpenRules approach applies Eclipse as a powerful IDE for rule integration within a java-based development environment. Practically, Eclipse offers the control mechanism for Excel-based rules and is applied for code editing, debugging and testing of rule projects. OpenRules also supports implementation of a web-based graphical user interface (GUI) via Apache Tomcat Java Servlet. Moreover, its libraries allow defining layouts of web pages and relationships among them directly in Excel. Such a powerful combination of Excel, Eclipse and OpenRules libraries and tools forms a practical framework for Rules Management and Web Application Development [75]. Figure 5.2 shows the OpenRules' rules repository and its supporting tools.

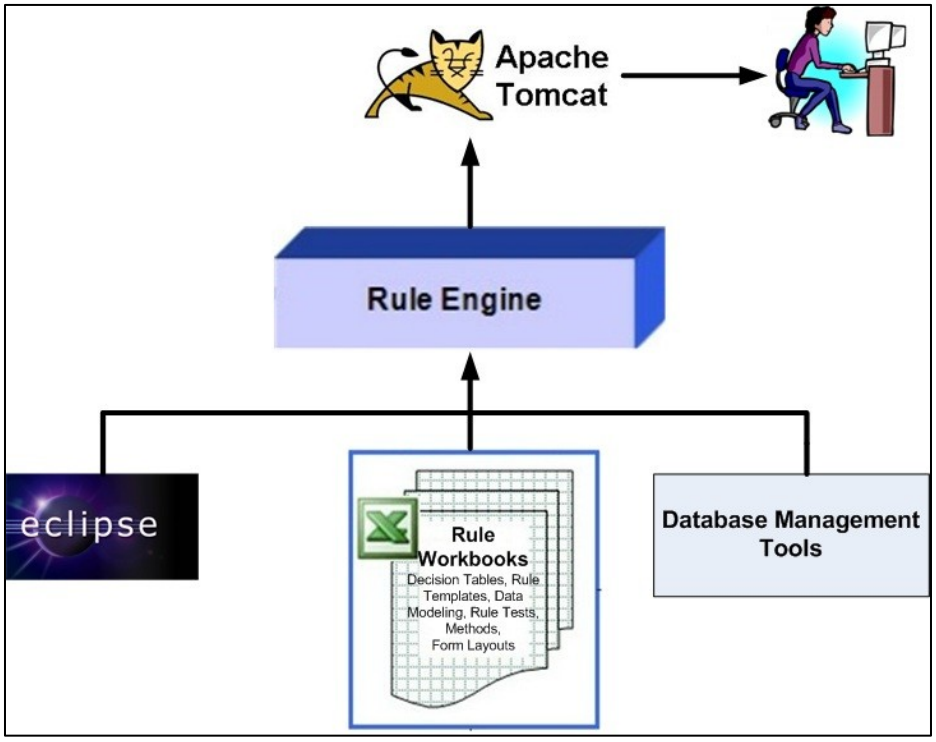


Figure 5.2: OpenRules' Rules Repository and Supporting Tools

As mentioned above, OpenRules utilizes Excel's workbooks to represent and maintain rules and web forms. Each workbook is comprised of one or more worksheets in order to

separate information by categories and each worksheet includes one or more tables. Some of the typical types of tables that are supported by OpenRules are including: Decision, Form Layouts, Data and Datatypes, Methods and Environment tables.

The most common way to indicate a set of rules is Decision tables. They are employed to express and evaluate various decision situations, where the state of a number of conditions resolves the execution of a set of actions. The execution logic of one rule which points to one row in the decision table is as follows:

IF all conditions are satisfied THEN execute actions

Therefore, actions are executed if all conditions in the same row are assessed to be true. An empty condition cell means that condition is always true. Three decision tables that are constructed for this thesis are *recommendTherapy, recommendDose* and *drugInteraction* that are displayed in Figure 5.3, 5.4 and 5.5, respectively. These tables are placed in three different worksheets that are located in *HealthCareRules.xls* workbook. The rules in the above tables are constructed based on the knowledge in DDO and the medical facts that are extracted from an accepted drug guide in medicine [76].

HealthCareRules Workbook also contains another worksheet that includes a method table to implement the formula for *creatinineClearance*, for instance. Creatinine Clearance level is a useful measure for indicating the state of kidney functionality which is a significant factor in medicine for recommendation of some specific drugs [76, 77]. Creatinine Clearance is calculated through following formula [76]:

Creatinine Clearance = [[140 – age (yr)] weight (kg)] / [72* Creatinine Level]*

The method table that computes *creatinineClearance* is shown in Figure 5.6.

Rules void **recommendTherapy**(Visit visit)					
C1	C2	C3	C4	A1
visit.encounterDiagnosis.equals(Dx)	visit.patient.age > minAge	visit.patient.age <= maxAge	contains(visit.patient.allergies,allergy)	visit.medication = medication
String Dx	int minAge	int maxAge	String allergy	String medication
If Encounter diagnosis is	**If patient is older than**	**If patient is younger than**	**If patient is allergic to**	**Define Medication**
Acute Sinusitis		18		Cefuroxime
Acute Sinusitis	18			Amoxicillin
Acute Sinusitis			Penicillin	Levofloxacin
Diabetes mellitus type2	6			Insuline Glargine, Glibenclamide, Metformin
Diabetes mellitus type1	6			Insuline Glargine
Coronary Heart Disease				Asa,Metoprolol, Heparin, Lovastatin, Perindopril, Clopidogrel
Acromegaly				Somatropin Recombinant
Otitis Media				Azithromycin
Esophagitis				Pantoprazole
Pulmonary hypertension				Treprostinil
Nephrotic Syndrome				Triamterene
Lemierre's Syndrome				Drotrecogin Alfa
Gastroesophageal Reflux Disease				Cimetidine
Human Immunodeficiency Virus Infectious Disease				Amprenavir

Figure 5.3: Recommend Therapy Decision Table

Rules void **recommendDose(Visit visit)**						
	C1	C2	C3	C4	C5	A1
	contains(medications, visit.medication)	visit.patient.age > minAge	visit.patient.age <= maxAge	visit.patient.creatinineLevel > clMin	visit.patient.creatinine Clearance < ccrMax	visit.dose = dose
	String[] medications	int minAge	int maxAge	double clMin	double ccrMax	String dose
#	**If recommended medication is one of the following**	**If patient is older than**	**If patient is younger than**	**If patient's Creatinine Level is more than**	**If patient's Creatinine Clearance is less than**	**Recommended Dose**
1	Amoxicillin Cefuroxime Levofloxacin		15			250mg every 24 hours for 7 days
2	Amoxicillin Cefuroxime Levofloxacin	60				250mg every 24 hours for 10 days
3	Amoxicillin Cefuroxime Levofloxacin	15	60			500mg every 24 hours for 14 days
4	Amoxicillin Cefuroxime Levofloxacin	15	60	1.4	50	250mg every 24 hours for 14 days
5	Insuline Glargine	18				1 unit/kg every day
6	Insuline Glargine		18			0.5 unit/kg every day
7	Insuline Glargine, Glibenclamide, Metformin	18				10 units once or twice a day, 2.5-5 mg daily with breakfast, 500mg once daily
8	Asa,Metoprolol, Heparin, Lovastatin, Perindopril, Clopidogrel	17				325 mg every day,100 mg a day in two divided doses, 150 unit/kg every 4 hour, 30 mg a day with meals, 4 mg daily, 75 mg a day
9	Asa,Metoprolol, Heparin, Lovastatin, Perindopril, Clopidogrel		17			15mg every 4 hour, 25 mg a day in two divided doses, 50 units/kg every 4 hour, 10 mg a day with meals, 1 mg daily, 30 mg a day
10	Asa,Metoprolol, Heparin, Lovastatin, Perindopril, Clopidogrel	17			30	325 mg every day,100 mg a day in two divided doses, 150 unit/kg every 4 hour, 20 mg a day with meals, 2 mg daily, 75 mg a day
11	Somatropin Recombinant	15				0.1 mg/kg every day
12	Somatropin Recombinant		15			0.3 mg/kg every week
13	Azithromycin	12				250 mg daily for 7 days
14	Azithromycin		12			30 mg daily for 5 days
15	Pantoprazole	18				40 mg for 8 weeks
16	Treprostinil	18				1.25 nanograms/kg/min/wk for first 4 weeks, then 2.5 nanograms/kg/min/wk
17	Triamterene	18				100 mg every other day after meals
18	Drotrecogin Alfa	18				24mcg/kg/hr for total of 96 hour
19	Cimetidine	18				200mg as the symptom occur, up to 800mg in 24 hour
20	Cimetidine	12	18			100mg as the symptom occur, up to 400mg in 24 hour
21	Amprenavir	18				1200 mg twice a day

Figure 5.4: Recommend Dose Decision Table

Rules void **drugInteraction(Visit visit)**		
C1	C2	A1
visit.medication.contains(medication1)	visit.patient.activeMedication.equals (medication2)	visit.warning = warning;
String medication1	String medication2	String warning
Recommended Medication	**Active Medication**	**Produce Warning**
		None
Levofloxacin	Coumadin	Coumadin and Levofloxacin can result in reduced effectiveness of Coumadin.
Amprenavir	Lovastatin	The beta-blocker, metoprolol, may decrease symptoms of hypoglycemia.
Insuline Glargine	Metoprolol	The beta-blocker, metoprolol, may decrease symptoms of hypoglycemia during concomitant use of Insuline Glargine.
Insuline Glargine	Somatropin Recombinant	Somatropin may antagonize the hypoglycemic effect of insulin glargine. Monitor for changes in fasting and postprandial blood sugars.
Drotrecogin Alfa	Treprostinil	The prostacyclin analogue, Treprostinil, increases the risk of bleeding when combined with the anticoagulant, Drotrecogin alfa. Monitor for increased bleeding during concomitant thearpy.
Drotrecogin Alfa	Clopidogrel	Antiplatelet agents such as clopidogrel may enhance the adverse/toxic effect of Drotrecogin Alfa. Bleeding may occur. Increase monitoring for signs/symptoms of bleeding during concomitant therapy. If possible, avoid use of drotrecogin within 7 days of use of any IIb/IIIa antagonists, higher dose aspirin (more than 650 mg/day), or use of other antiplatelet agents.
Drotrecogin Alfa	Heparin	The potential benefits of drotrecogin alfa should be weighed against an increased risk of bleeding in patients receiving therapeutic doses of heparin. Monitor for bleeding during concomitant therapy, and immediately stop infusion of drotrecogin if clinically important bleeding occurs. In patients receiving prophylactic heparin doses, consider continuing this during drotrecogin.
Somatropin Recombinant	Insuline Glargine	Somatropin may antagonize the hypoglycemic effect of insulin glargine. Monitor for changes in fasting and postprandial blood sugars.
Somatropin Recombinant	Metformin	Somatropin may antagonize the hypoglycemic effect of metformin. Monitor for changes in fasting and postprandial blood sugars.
Somatropin Recombinant	Glibenclamide	Somatropin may antagonize the hypoglycemic effect of glibenclamide. Monitor for changes in fasting and postprandial blood sugars.
Azithromycin	Lovastatin	The macrolide antibiotic, azithromycin, may increase the serum concentration of lovastatin by decreasing its metabolism. Monitor for changes in the therapeutic and adverse effects of lovastatin if azithromycin is initiated, discontinued or dose changed.
Triamterene	Perindopril	Increased risk of hyperkalemia.
Triamterene	Treprostinil	Additive hypotensive effect. Monitor antihypertensive therapy during concomitant use.
Cimetidine	Metformin	Cimetidine may increase the therapeutic and adverse effects of metformin by increasing its serum concentration. Consider alternate therapy.
Cimetidine	Metoprolol	Cimetidine may increase the serum concentration of metoprolol by decreasing its metabolism.
Treprostinil	Metoprolol	Additive hypotensive effect. Monitor antihypertensive therapy during concomitant use.
Treprostinil	Perindopril	Additive hypotensive effect. Monitor antihypertensive therapy during concomitant use.
Treprostinil	Triamterene	Additive hypotensive effect. Monitor antihypertensive therapy during concomitant use.
Treprostinil	Asa	The prostacyclin analogue, Treprostinil, increases the risk of bleeding when combined with the antiplatelet agent, Acetylsalicylic acid. Monitor for increased bleeding during concomitant thearpy.
Treprostinil	Heparin	The prostacyclin analogue, Treprostinil, increases the risk of bleeding when combined with the anticoagulant, Heparin. Monitor for increased bleeding during concomitant thearpy.
Treprostinil	Drotrecogin Alfa	The prostacyclin analogue, Treprostinil, increases the risk of bleeding when combined with the anticoagulant, Drotrecogin alfa. Monitor for increased bleeding during concomitant thearpy.
Treprostinil	Clopidogrel	The prostacyclin analogue, Treprostinil, increases the risk of bleeding when combined with the antiplatelet agent, Clopidogrel. Monitor for increased bleeding during concomitant thearpy.
Pantoprazole	Clopidogrel	Pantoprazole may decrease serum concentrations of the active metabolite(s) of clopidogrel. Due to the possible risk for impaired clopidogrel effectiveness with this combination, clinicians should carefully consider the need for concurrent pantoprazole therapy in patients receiving clopidogrel. Monitor response to clopidogrel closely when using clopidogrel with pantoprazole. Whether there are differences among individual proton pump inhibitors is unclear. Other acid-lowering therapies (e.g., H2-receptor antagonists, antacids, etc.) do not appear to share this interaction with clopidogrel.

Clopidogrel	Pantoprazole	Pantoprazole may decrease serum concentrations of the active metabolite(s) of clopidogrel. Due to the possible risk for impaired clopidogrel effectiveness with this combination, clinicians should carefully consider the need for concurrent pantoprazole therapy in patients receiving clopidogrel. Monitor response to clopidogrel closely when using clopidogrel with pantoprazole. Whether there are differences among individual proton pump inhibitors is unclear. Other acid-lowering therapies (e.g., H2-receptor antagonists, antacids, etc.) do not appear to share this interaction with clopidogrel.
Clopidogrel	Drotrecogin Alfa	Antiplatelet agents such as clopidogrel may enhance the adverse/toxic effect of Drotrecogin Alfa. Bleeding may occur. Increase monitoring for signs/symptoms of bleeding during concomitant therapy. If possible, avoid use of drotrecogin within 7 days of use of any IIb/IIIa antagonists, higher dose aspirin (more than 650 mg/day), or use of other antiplatelet agents.
Clopidogrel	Treprostinil	The prostacyclin analogue, Treprostinil, increases the risk of bleeding when combined with the antiplatelet agent, Clopidogrel. Monitor for increased bleeding during concomitant thearpy.
Metoprolol	Insuline Glargine	The beta-blocker, metoprolol, may decrease symptoms of hypoglycemia.
Metoprolol	Glibenclamide	The beta-blocker, metoprolol, may decrease symptoms of hypoglycemia.
Metoprolol	Treprostinil	Additive hypotensive effect. Monitor antihypertensive therapy during concomitant use of Metoprolol and Treprostinil.
Metoprolol	Cimetidine	Cimetidine may increase the serum concentration of metoprolol by decreasing its metabolism.
Perindopril	Triamterene	Concomitant use of Triamterene and Perindopril may increase the risk of hyperkalemia .
Perindopril	Treprostinil	Additive hypotensive effect. Monitor antihypertensive therapy during concomitant use of Treprostinil and Perindopril.
Asa	Glibenclamide	Acetylsalicylic acid increases the effect of the sulfonylurea, glibenclamide.
Asa	Heparin	Concomitant use of Asa and Heparin may increase the risk of bleeding.
Asa	Treprostinil	The prostacyclin analogue, Treprostinil, increases the risk of bleeding when combined with the antiplatelet agent, Acetylsalicylic acid. Monitor for increased bleeding during concomitant thearpy.
Lovastatin	Amprenavir	Amprenavir may increase the serum concentration of the lovastatin. Concomitant therapy is contraindicated.
Lovastatin	Azithromycin	The macrolide antibiotic, azithromycin, may increase the serum concentration of lovastatin by decreasing its metabolism. Monitor for changes in the therapeutic and adverse effects of lovastatin if azithromycin is initiated, discontinued or dose changed.
Heparin	Drotrecogin Alfa	The potential benefits of drotrecogin alfa should be weighed against an increased risk of bleeding in patients receiving therapeutic doses of heparin. Monitor for bleeding during concomitant therapy, and immediately stop infusion of drotrecogin if clinically important bleeding occurs. In patients receiving prophylactic heparin doses, consider continuing this during drotrecogin.
Heparin	Treprostinil	The prostacyclin analogue, Treprostinil, increases the risk of bleeding when combined with the anticoagulant, Heparin. Monitor for increased bleeding during concomitant thearpy.
Heparin	Asa	Acetylsalicylic acid can increase the risk of bleeding for patients on heparin therapy or vice versa.
Glibenclamide	Metoprolol	The beta-blocker, metoprolol, may decrease symptoms of hypoglycemia during concomitant use of Glibenclamide.
Glibenclamide	Somatropin Recombinant	Somatropin may antagonize the hypoglycemic effect of glibenclamide. Monitor for changes in fasting and postprandial blood sugars.
Glibenclamide	Asa	Acetylsalicylic acid increases the effect of the sulfonylurea, glibenclamide.
Metformin	Somatropin Recombinant	Somatropin may antagonize the hypoglycemic effect of metformin. Monitor for changes in fasting and postprandial blood sugars.
Metformin	Cimetidine	Cimetidine may increase the therapeutic and adverse effects of metformin by increasing its serum concentration. Consider alternate therapy.

Figure 5.5: Drug Interaction Decision Table

```
Method double creatinineClearance(Patient patient)

patient.creatinineClearance =
    (140 - patient.age) * patient.weight / (patient.creatinineLevel * 72);
return patient.creatinineClearance;
```

Figure 5.6: Creatinine Clearance Method Table

In order to run the OpenRules engine, it is required to create a main method that is defined particularly for the engine run. The main method of our work is denoted in the *HealthCareMain.xls* workbook as shown in the Figure 5.7. In the same work sheet another method is created with the purpose of validation of Creatinine level that is used in the computation of Creatinine Clearance Formula which is displayed in Figure 5.8.

```
Method TableLayout main(Dialog d)

Visit visit = (Visit)d.get("visit");
out(visit.toString());
// validate current step input
nextLayout(visit);
if (dialog().errors == 0)  {   // define next step
    processingFlowRules(visit);
    creatinineClearance(visit.patient);
    recommendTherapy(visit);
    recommendDose(visit);
    drugInteraction(visit);
    nextLayout(visit);
}
return mainLayout();
```

Figure 5.7: Main Method Table

```
Method String validateCreatinineLevel(Patient patient)
String msg ="";
if (patient.creatinineLevel < 0.01)
      msg = "ERROR: creatinine level can not be less than 0.01";
if (patient.creatinineLevel > 15)
      msg = "ERROR: creatinine level can not be more than 15";
if (patient.creatinineLevel > 3 && patient.age < 7)
      msg = "ERROR: a child can not have creatinine level higher
than 3";
if (patient.creatinineLevel <0.2 && patient.age > 15)
      msg = "ERROR: an adult can not have creatinine level lesser
than 0.2";
if (!msg.equals(""))
   dialog().addError();
return msg;
```

Figure 5.8: Validate Creatinine Level Method Table

When OpenRules engine runs, it downloads the main xls file and any other files and libraries that are defined in the main.xls file. Thus, to make references to our additional Excel files and libraries, making an Environment table in the main workbook is essential. This table, shown in Figure 5.9, allows the engine to download all other files using the *include* properties and recognize the other application files that the system is connected to such as relevant java files or Tomcat libraries.

Environment	
include.path	http://localhost:8080/openrules.forms.lib/
include	../include/HealthCareRules.xls
	../include/HealthCareForms.xls
	<Dialog.xls>
import.static	com.openrules.tools.Methods
import.java	healthcare.*

Figure 5.9: Environment Table

In this thesis, a web-based GUI via Apache Tomcat 6.0 is implemented to communicate with the user and present the results of our rules. OpenRules provides a library called *Forms* that allows defining layouts of web pages and relationships among them directly in Excel as Layout tables. Figure 5.10 indicates the Layout tables that are designed to define 3 web forms of *VisitInformation*, *PatientInformation* and *Recommendations*. To facilitate the navigation from one form to another, two buttons of *Next* and *Prev* are added at the top of each form. Moreover a *Refresh* button is implemented in the *PatientInformation* page to let recalculation of the Creatinine Clearance value whenever is needed. Figure 5.11 displays the *processingFlowRules* and *nextLayout* tables that are created to establish the relationship among the 3 web pages. All of the above tables are built in the *HealthCareForms.xls* workbook.

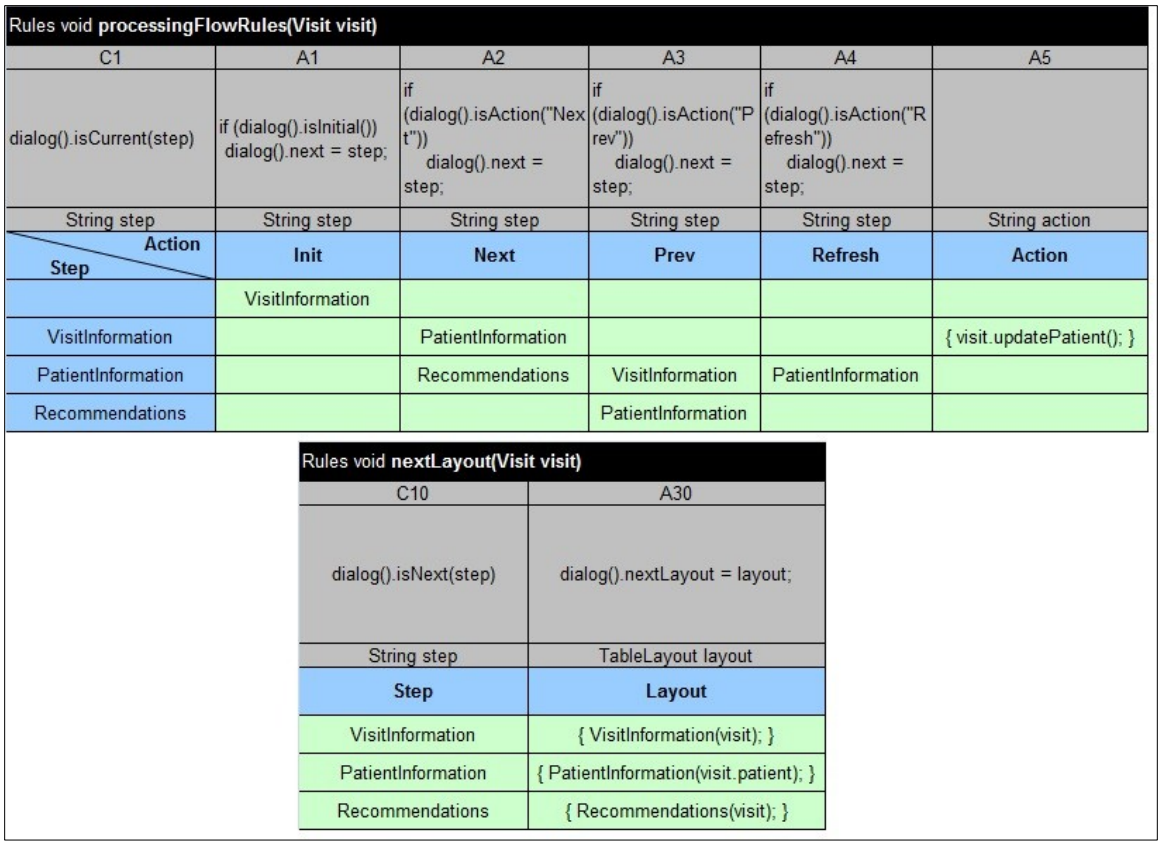
Layout TableLayout mainLayout()		
properties	width	100%
	cellspacing	4
	cellpadding	2
	style	background-color:steelblue
	border	1
actionButton("Prev")	Patient Therapy Recommendations	actionButton("Next")
CurrentLayout()		

Layout TableLayout VisitInformation(Visit visit)		
properties	width	100%
	cellspacing	4
	cellpadding	2
	style	background-color:lightblue
	border	0
header("VISIT INFORMATION")		
"Visit Date:"	[visit.date]	
"Patient:"	[visit.patient.name][visit.getNames()]	
"Encounter Diagnosis:"	[visit.encounterDiagnosis]	

Layout TableLayout PatientInformation(Patient patient)			
properties	width	100%	
	cellspacing	4	
	cellpadding	2	
	style	background-color:lightblue	
	border	0	
header("PATIENT INFORMATION")			
"Name:"	patient.name		
"Age:"	patient.age		
"Weight"	[patient.weight]		
"Creatinine Level:"	[patient.creatinineLevel]	validateCreatinineLevel(patient);	
"Creatinine Clearance:"	format(patient.creatinineClearance)	actionButton("Refresh")	
"Active Medication:"	[patient.activeMedications[0]] [medications()]	[patient.activeMedications[1]] [medications()]	[patient.activeMedications[2]] [medications()]
"Allergies:"	[patient.allergies[0]][medications()]	[patient.allergies[1]][medications()]	[patient.allergies[2]][medications()]

Layout TableLayout Recommendations(Visit visit)		
properties	width	100%
	cellspacing	4
	cellpadding	2
	style	background-color:lightblue
	border	0
header("RECOMMENDATIONS")		
"Recommended Medication:"	visit.medication	
"Recommended Dose:"	visit.dose	
"Warning:"	visit.warning	

Figure 5.10: Layout Tables of Web Pages



Rules void **processingFlowRules(Visit visit)**

C1	A1	A2	A3	A4	A5
dialog().isCurrent(step)	if (dialog().isInitial()) dialog().next = step;	if (dialog().isAction("Next")) dialog().next = step;	if (dialog().isAction("Prev")) dialog().next = step;	if (dialog().isAction("Refresh")) dialog().next = step;	
String step	String step	String step	String step	String step	String action
Step \ Action	**Init**	**Next**	**Prev**	**Refresh**	**Action**
	VisitInformation				
VisitInformation		PatientInformation			{ visit.updatePatient(); }
PatientInformation		Recommendations	VisitInformation	PatientInformation	
Recommendations			PatientInformation		

Rules void **nextLayout(Visit visit)**

C10	A30
dialog().isNext(step)	dialog().nextLayout = layout;
String step	TableLayout layout
Step	**Layout**
VisitInformation	{ VisitInformation(visit); }
PatientInformation	{ PatientInformation(visit.patient); }
Recommendations	{ Recommendations(visit); }

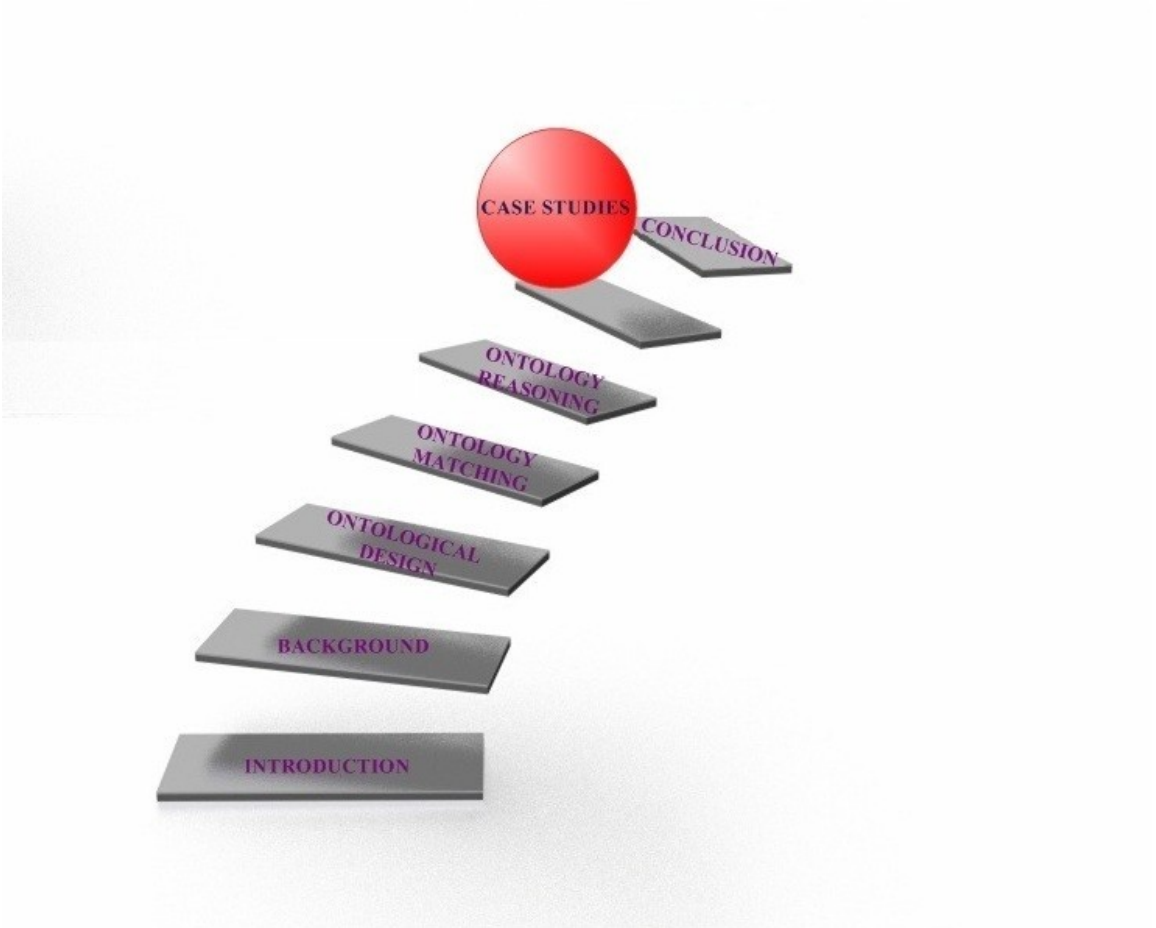
Figure 5.11: Processing Flow Rules and Next Layout Tables

The OpenRules engine receives patient information from a patient database. To prototype our work, a sample of patient database is created in order to test the system which is illustrated in Figure 5.12. This database is built in Access 2007 and is connected to the rule engine through some coding in java framework. The results of reasoning via rule engine are demonstrated in Chapter 6 of this thesis.

File As	ID	Last Name	First Name	Date of Birth	Gender	Marital Status	Mobile Phone	Allergies	Medications	Weight	Creatinine Level
Smith, John	1	Smith	John	7/12/1990	Male	Single	9875635	Penicillin	Coumadin	78.00	2.00
Smith, Mary	2	Smith	Mary	2/12/1993	Female	Married	1327233			48.00	2.35
Peng, Allan	3	Peng	Allan	3/12/1994	Male	Single	6312587		Metoprolol	64.00	2.40
Gray, John	4	Gray	John	5/7/1996	Male	Single	2101655	Penicillin, Streptomycin	Asa	70.00	3.00
Jones, Lisa	5	Jones	Lisa	6/9/1984	Female	Single	8590668	Streptomycin	Treprostinil	57.00	2.57
Howell, Ann	6	Howell	Ann	8/11/1980	Female	Married	5534988		Pantoprazole	60.00	2.64
Sakahara, Linda	7	Sakahara	Linda	9/4/2001	Female	Single	1184529			37.00	2.00
Graham, Sarah	8	Graham	Sarah	8/5/1970	Female	Married	7864449		Metformin	75.00	2.75
Sanchez, Anthony	9	Sanchez	Anthony	12/2/1999	Male	Single	5981549			45.00	3.25
Dalton, Shawn	10	Dalton	Shawn	4/3/1988	Male	Single	9359139		Insuline Glargine	90.00	2.59
Davids, Donald	11	Davids	Donald	1/1/1989	Male	Single	4760055	Streptomycin	Glibenclamide	81.00	2.45
Clark, Kevin	12	Clark	Kevin	6/8/1975	Male	Married	4218615		Heparin	85.00	3.35
Moore, Isabela	13	Moore	Isabela	10/11/1979	Female	Married	0446846	Penicillin, Streptomycin	Triamterene, Metformin	69.00	2.85
Vernon, David	14	Vernon	David	9/12/1960	Male	Married	7278660		Asa	83.00	3.15
Stephens, James	15	Stephens	James	8/8/1965	Male	Married	6578990	Streptomycin		88.00	3.50
Green, Janet	16	Green	Janet	7/10/1971	Female	Married	8556846		Drotrecogin Alfa	66.00	2.77
Lee, Maria	17	Lee	Maria	5/4/1985	Female	Single	8227270		Cimetidine	54.00	2.89
Hamilton, Mandy	18	Hamilton	Mandy	9/10/1977	Female	Married	9254811	Penicillin	Clopidogrel	62.00	2.95
Albin, Brad	19	Albin	Brad	2/1/1981	Male	Single	5546476	Penicillin	Lovastatin	79.00	3.35
Clauda, Ryan	20	Clauda	Ryan	5/7/1973	Male	Married	3372118	Penicillin, Streptomycin	Coumadin, Treprostinil	84.00	3.60
Pierce, Abigail	21	Pierce	Abigail	8/1/1967	Female	Married	2972744	Penicillin	Amprenavir	72.00	2.69

Figure 5.12: A Sample of Patient Database

CHAPTER 6
CASE STUDIES



This Chapter portrays the operational environment of the developed system. The results of methodologies associated with semantic characteristics and techniques for drug recommendation, including ontology query and semantic inference are described. For each method, some scenarios are provided to prototype the main system services and demonstrate the validity of our semantically integrated Disease-Drug knowledge base.

6.1 Query Engine Results

As mentioned in Chapter 5, our system provides physicians with some essential direct queries that can be made from DDO through Java framework and Jena API. Moreover, these queries are applied to illustrate the validity of our semantic disease-drug knowledge base as well as experiment with semantic web techniques. In this practice, we have conducted a series of questions and issues that are usually faced with therapeutic cases. The particular questions to be answered are what are the typical drugs for the diseases and their relevant features including various prices, possible side effects, accessible brands, applicable synonyms, potential indications, multiple manufacturers and drugs or food that can interact with. Therefore, our system is capable of responding to nine significant queries regarding disease-drug knowledge base.

The first step when the system runs is to select the desired query among nine queries that are shown previously in Chapter 5. Figure 6.1 and 6.2 display the results of the first query which retrieves the relevant drugs of a disease for *Coronary Heart Disease* and *Diabetes mellitus type 2*. To retrieve the typical drugs of these diseases, initially the system looks for the class of desired disease in DDO. Then it acquires and returns the labels of the drug classes that are in a relationship with this disease class through *may treated by* property.

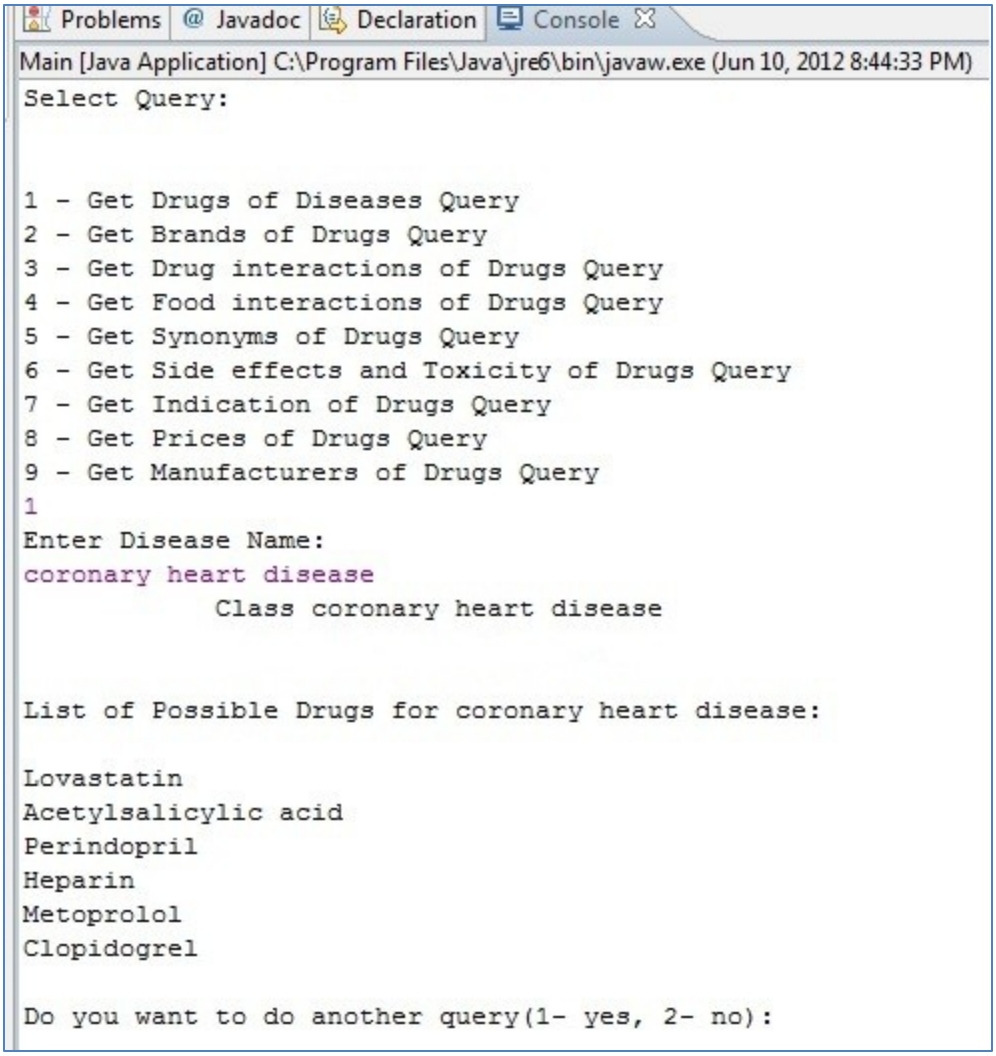


Figure 6.1: Retrieved drugs from DDO for Coronary Heart Disease

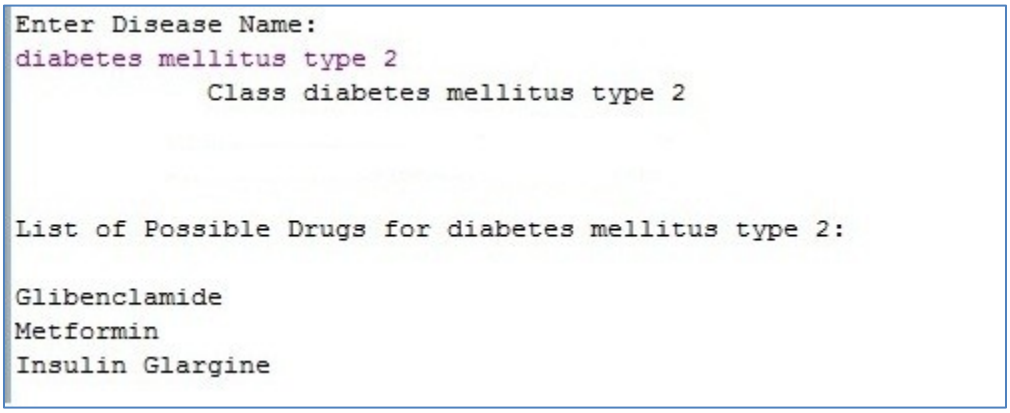


Figure 6.2: Retrieved drugs from DDO for Diabetes mellitus type 2

In order to obtain the brands of one specific drug, first the system seeks for the class of listed drug in DDO. After that, it finds the *brandsSlot* property of the determined drug class and obtains the object of data type property, then returns it as a String. As an example the accessible brands of *Metoprolol* and *Metformin* are shown in Figure 6.3.

```
2
Enter Drug Name:
Metoprolol
    Class Metoprolol

List of Possible Brands for Metoprolol:

Toprol-XL
Selopral
Selo-Zok
Lopressor
Toprol
Lopresor
Beloc
Seloken
Lopresoretic
Lopressor HCT
Metroprolol
Toprol XL
Betaloc

2
Enter Drug Name:
Metformin
    Class Metformin

List of Possible Brands for Metformin:

Novo-Metformin
Glucophage
Fortamet
Apo-Metformin
Glucophage XR
Glycon
Gen-Metformin
Nu-Metformin
Riomet
```

Figure 6.3: Obtained brands of Metoprolol and Metformin from DDO

The rest of the queries follow similar process as the previous one that by obtaining the desired drug class, the relevant *Slot* property is found and the object of data type property of that slot is returned as a String. In the following, separated examples are provided for each of the queries. Metoprolol and Metformin are two drugs that are applied in these examples to run the queries. The possible drug interactions and food interactions of *Metoprolol* and *Metformin* are displayed in Figure 6.4 and Figure 6.5 respectively.

```
3
Enter Drug Name:
Metoprolol
    Class Metoprolol

List of Possible Drug Interactions for Metoprolol:

Cimetidine
Cimetidine may increase the serum concentration of metoprolol
by decreasing its metabolism.
Treprostinil
Additive hypotensive effect. Monitor antihypertensive therapy
during concomitant use.
Glibenclamide
The beta-blocker, metoprolol, may decrease symptoms of hypoglycemia.
Insulin Glargine
The beta-blocker, metoprolol, may decrease symptoms of hypoglycemia.
3
Enter Drug Name:
Metformin
    Class Metformin

List of Possible Drug Interactions for Metformin:

Cimetidine
Cimetidine may increase the therapeutic and adverse effects of metformin
increasing its serum concentration. Consider alternate therapy.
Somatropin recombinant
Somatropin may antagonize the hypoglycemic effect of metformin. Monitor
for changes in fasting and postprandial blood sugars.
```

Figure 6.4: Possible Drug Interactions for Metoprolol and Metformin

```
4
Enter Drug Name:
Metoprolol
    Class Metoprolol

List of Possible Food Interactions for Metoprolol:

Take with food.
Avoid natural licorice.
Avoid alcohol.
4
Enter Drug Name:
Metformin
    Class Metformin

List of Possible Food Interactions for Metformin:

Take with food to reduce irritation.
Avoid alcohol.
```

Figure 6.5: Possible Food Interactions for Metoprolol and Metformin

Figure 6.6 shows the applicable synonyms of Metoprolol and Metformin.

```
5
Enter Drug Name:
Metoprolol
    Class Metoprolol

List of Possible Synonyms for Metoprolol:

Metoprolol succinate
Metoprolol Tartrate
5
Enter Drug Name:
Metformin
    Class Metformin

List of Possible Synonyms for Metformin:

metformin hydrochloride
Metformin HCL
```

Figure 6.6: Appropriate Synonyms for Metoprolol and Metformin

Correspondingly, the possible side effects and potential indications of Metoprolol and Metformin are demonstrated in Figure 6.7 and Figure 6.8.

```
6
Enter Drug Name:
Metoprolol
    Class Metoprolol

List of Possible Side Effects and Toxicities for Metoprolol:

LD<sub>50</sub>=5500 mg/kg (orally in rats). toxic effects include
bradycardia, hypotension, bronchospasm,and cardiac failure.
LD<sub>50</sub>=2090 mg/kg (orally in mice)
6
Enter Drug Name:
Metformin
    Class Metformin

List of Possible Side Effects and Toxicities for Metformin:

Acute oral toxicity (LD<sub>50</sub>): 350 mg/kg [Rabbit]. It would
be expected that adverse reactions of a more intense character
including epigastric discomfort, nausea, and vomiting followed by
diarrhea, drowsiness, weakness,dizziness, malaise and headache might
be seen.
```

Figure 6.7: Probable Side Effects and Toxicities for Metoprolol and Metformin

```
7
Enter Drug Name:
Metoprolol
    Class Metoprolol

List of Possible Indications for Metoprolol:

For the treatment of hypertension and angina pectoris.
7
Enter Drug Name:
Metformin
    Class Metformin

List of Possible Indications for Metformin:

For use as an adjunct to diet and exercise to improve glycemic
control in adult patients (18 years and older) with type 2 diabetes.
```

Figure 6.8: Feasible Indications for Metoprolol and Metformin

A variety of prices for Metoprolol and Metformin are retrieved through query 8 which is displayed in Figure 6.9.

```
8
Enter Drug Name:
Metoprolol
    Class Metoprolol

List of Possible Prices for Metoprolol:

tablet
Metoprolol Succinate 25 mg 24 Hour tablet   1.17 USD
tablet
Toprol xl 50 mg tablet  1.24 USD
tablet
Lopressor 100 mg tablet  2.75 USD
tablet
Toprol xl 100 mg tablet  0.8  USD
ml
Lopressor 1 mg/ml   1.29  USD

8
Enter Drug Name:
Metformin
    Class Metformin

List of Possible Prices for Metformin:

ml
Riomet 500 mg/5 ml solution  0.27 USD
tablet
Nu-Metformin 850 mg tablet  0.21  USD
tablet
Novo-Metformin 500 mg tablet  0.13  USD
tablet
Glucophage 850 mg tablet  1.94  USD
tablet
Metformin hcl 850 mg tablet   1.22  USD
```

Figure 6.9: Available Prices for Metoprolol and Metformin

Several manufacturers for Metoprolol and Metformin are acquired through query 9 which is exemplified in Figure 6.10.

```
9
Enter Drug Name:
Metoprolol
    Class Metoprolol

List of Possible Manufacturers for Metoprolol:

Teva pharmaceuticals usa inc
Watson laboratories inc florida
Wockhardt ltd
Teva pharmaceuticals usa
Bedford laboratories

9
Enter Drug Name:
Metformin
    Class Metformin

List of Possible Manufacturers for Metformin:

Sandoz inc
Aurobindo pharma ltd
Bristol myers squibb co
Mutual pharmacal co
Alphapharm party ltd
```

Figure 6.10: Different Manufacturers for Metoprolol and Metformin

6.2 Drug Recommendation System Results

In this Section the final results of drug recommendation system is exemplified through diverse case studies. These results are achieved by applying *RecommendTherapy Rules, RecommendDose Rules and DrugInteaction Rules* which are described in Chapter 5. OpenRules engine employs above rules to infer the appropriate drug recommendation results based on the available facts and presented data. Obtainable data from patient database is employed to feed the rule engine. The system communicates with physicians through three pages of *Visit Information*, *Patient Information* and *Recommendations* which are displayed via GUI that is associated with OpenRules engine. In the *Visit*

Information form, name of the desired patient and the disease that is diagnosed for the patient is entered by the physician. Visit date is programmed to display the current date of visit. Based on the name of the patient, the medical history of patient is retrieved from patient database and presented in the next form which is *Patient Information*. However, the system also allows the physicians to make changes to some of this information where it is required. As an example the physician is able to change the amount of current patient's weight if it differs from the one that is stored in the database. In such a case, the amount of Creatinine Clearance needs to be recalculated by pushing the refresh button, as well. Finally, the results of drug recommendation are indicated in the third form that is called *Recommendations*. Following case studies present the usability and capability of the drug recommendation system in making distinct decisions based on the various circumstances and conditions.

In the first case study, we assume Shawn Dalton and John Smith as the patients who are diagnosed with Acute Sinusitis. And then we investigate the results of drug recommendation system based on their medical information and distinct rules that should be applied to their cases. Let's start our experiment with Shawn's information. The Visit Information for Shawn Dalton is displayed in Figure 6.11(a). Shawn Dalton's medical information that is retrieved from patient database is shown in the Patient Information form which is illustrated in Figure 6.11(b). Since Shawn is older than 18 and does not have any allergy to Penicillin, the system recommends Amoxicillin as a suitable drug for him based on the *RecommendTherapy Rules.* Based on the *RecommendDose Rules,* 500mg every 24 hours for 14 days is suggested as the appropriate dose because his age is between 15 and 60 and the amount of his Creatinine Clearance is more than 50. The

recommended drug, Amoxicillin, does not have any interaction with his active medication, Insulin Glargine based on the *DrugInteraction Rules*. Hence, the system does not show any warning. The recommendations for Shawn are illustrated in Figure 6.11(c).

The Visit Information of John Smith who is another patient with the Acute Sinusitis diagnosis is shown in Figure 6.12(a). From his medical information that is displayed in Figure 6.12(b), it is realized that he has allergies to Penicillin and Streptomycin and his active medication is Coumadin. Although his age is more than 18 like Shawn, he is not able to use Amoxicillin due to his allergy to Penicillin. Therefore, based on the *RecommendTherapy Rules*, our system proposes Levofloxacin as the alternative drug. Based on John's age and the amount of his Creatinine Clearance the same dosing rule as Shawn's case is also applied to his. Thus, 500mg every 24 hours for 14 days is recommended for John, as well. Ultimately, the proposed medication, Levofloxacin, shows some sort of interaction with Coumadin, a medication that John is currently on, based on the *DrugInteraction Rules.* Therefore, system shows warning for the interaction that may happen due to concurrent consumption of these two medications. The appropriate drug recommendation for John is displayed in Figure 6.12(c).

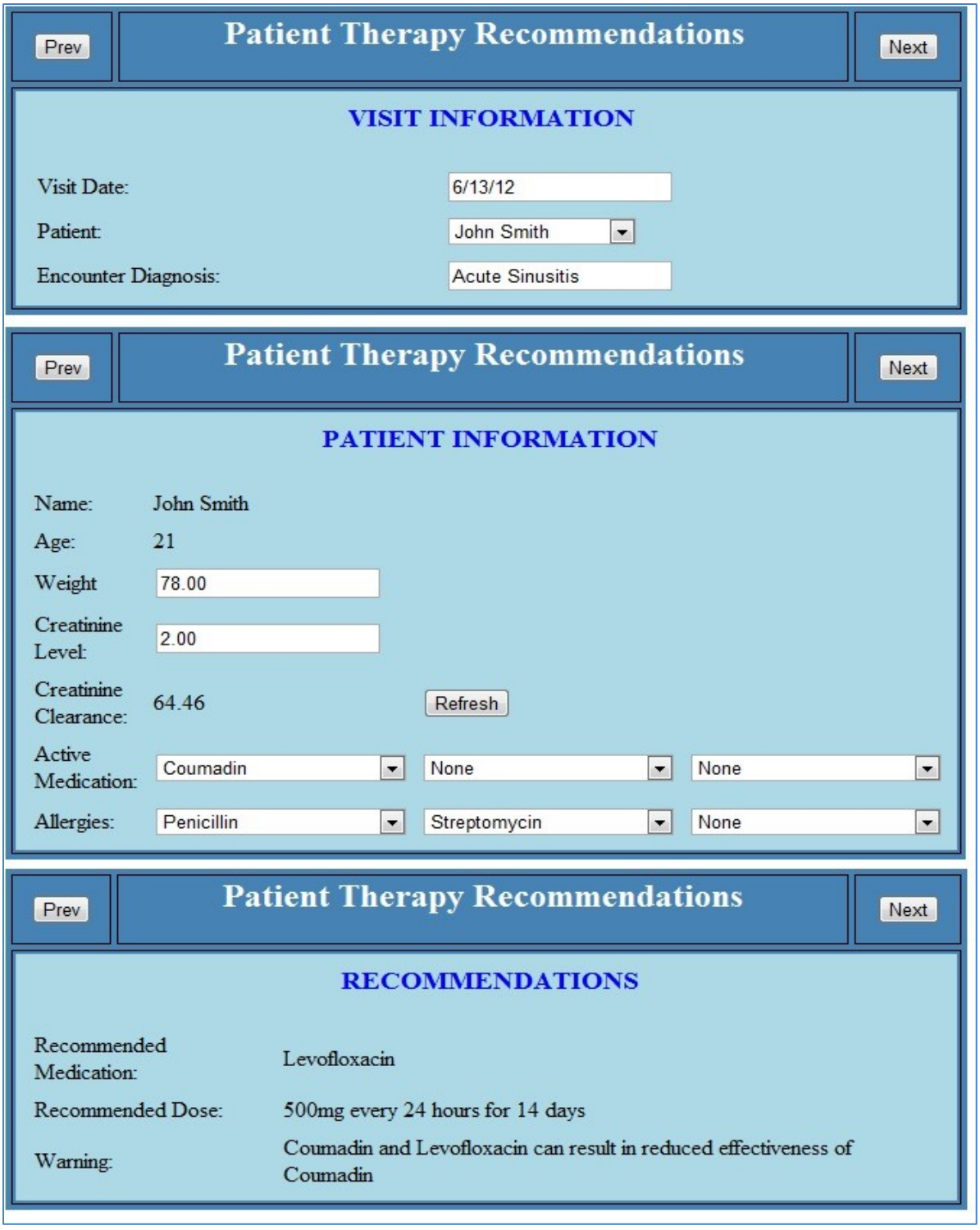
The outcome of this case study demonstrates how our proposed system is capable to apply proper rules and suggest different recommendations that are suitably based on the patients' allergies and active medications.

a

b

c

Figure 6.11: (a) Visit Information form of Shawn Dalton. (b) Patient Information form of Shawn Dalton. (c) Drug Recommendations for Shawn Dalton

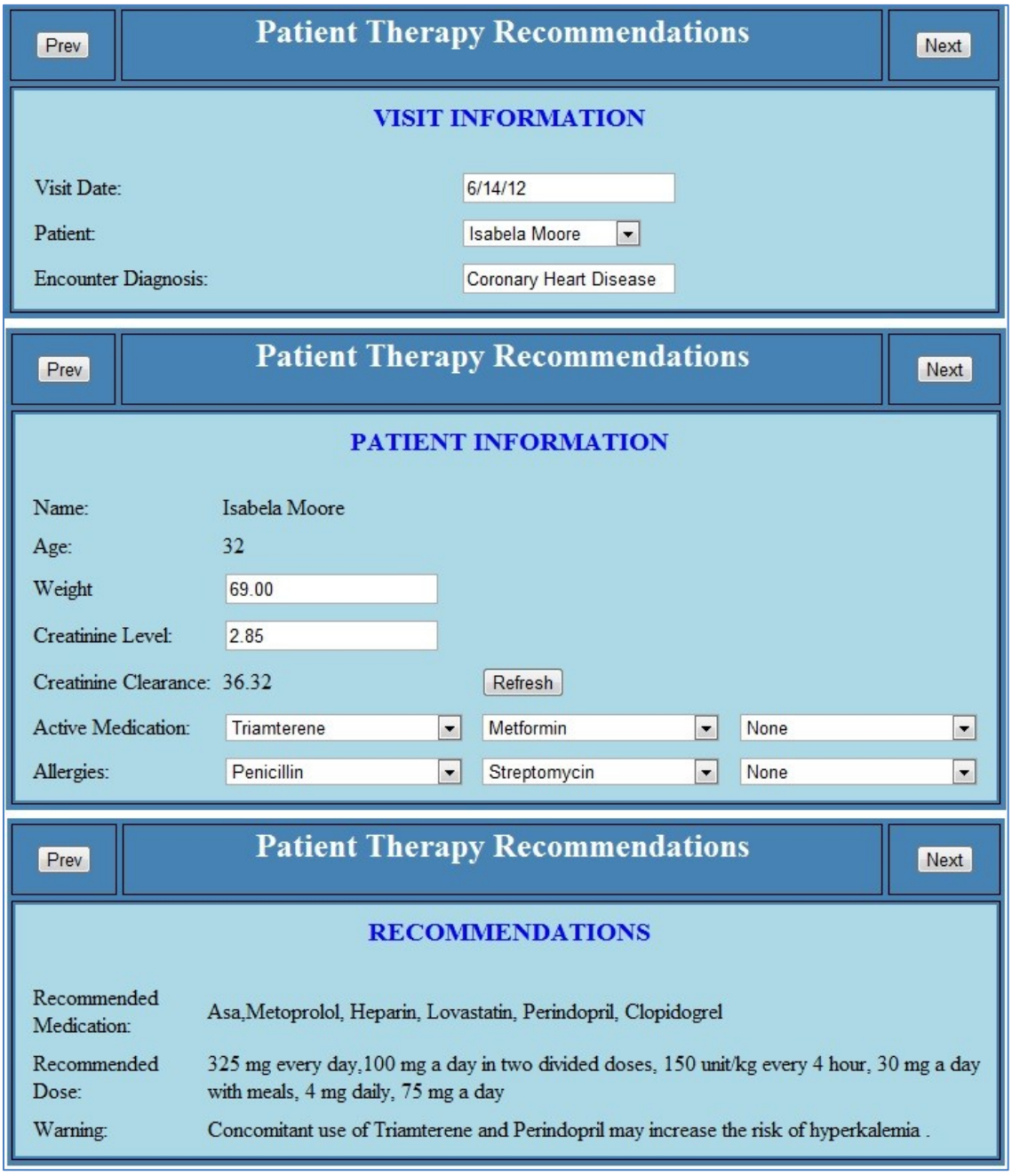


a
b
c

Figure 6.12: (a) Visit Information form of John Smith. (b) Patient Information form of John Smith. (c) Drug Recommendations for John Smith

In the second case study, three patients who are diagnosed with Coronary Heart Disease are used as our experimental cases. This case study shows the ability of our system to recommend appropriate drug doses depends on the amount of Creatinine Clearance and age of patient. Three patients that are retrieved from patient database are: Isabella More, Anthony Sanchez and Maria Lee. The Visit Information for each of them is displayed in Figure 6.13(a), 6.14(a) and 6.15(a), respectively. Based on the *RecommendTherapy Rules,* the same rule is applied in order to suggest proper medications for all three used patients. As a result, a set of drugs including: Asa, Metoprolol, Heparin, Lovastatin, Perindopril and Clopidogrel are recommended as the suitable medications for all of them that are diagnosed with Coronary Heart Disease. After that, system looks for Dosing Rules in *RecommendDose Rules* table to advocate the correct drug doses for each patient. From Isabella's medical Information, it is realized that she is 32 years old and the amount of her Creatinine Clearance is more than 30. Hence, the dosing rule that is relevant to patients older than 17 is considered for her. Figure 6.13(b) points to Isabella's information. Anthony who is just 12 years old is included in another category of dosing rules which is suitable for patients younger than 17. Anthony's relevant Patient Information form is illustrated in Figure 6.14(b). Although Maria is older than 17 like Isabella, the dosing rule that is applied to her case differs from Isabella's. The system allocates another dosing rule to Maria's case due to the amount of Maria's Creatinine Clearance which is less than 30. Figure 6.15(b) illustrates Maria's medical information. Based on the *DrugInteraction Rules,* the system reveals an interaction between one of Isabella's active medications, Triamterene and Perindopril which is one of the

recommended drugs for her. This is illustrated in the list of recommendations for her in Figure 6.13(c).



a
b
c

Figure 6.13: (a) Visit Information form of Isabella Moore. (b) Patient Information form of Isabella Moore. (c) Drug Recommendations for Isabella Moore

In Anthony's recommendation form which is shown in Figure 6.14(c), no warning is displayed since he is not on any other drugs.

a
b
c

Figure 6.14: (a) Visit Information form of Anthony Sanchez. (b) Patient Information form of Anthony Sanchez. (c) Drug Recommendations for Anthony Sanchez

a

b

c

Figure 6.15: (a) Visit Information form of Maria Lee. (b) Patient Information form of Maria Lee. (c) Drug Recommendations for Maria Lee

Ultimately, as Maria's recommendation form is displayed in Figure 6.15(c), the system warns an interaction between her active medication, Cimetidine and Metoprolol which is one of the recommended drugs for her.

Consequently, these case studies indicate the novelty of our approach to facilitate precise and reliable drug recommendations by considering various medical variables and diversity of patients' situations by applying suitable rules to the rule engine.

CHAPTER 7
CONCLUSION & FUTURE WORKS

In this thesis, a drug recommendation system based on the Semantic Web standards and technologies has been proposed to aid physicians and pharmacists in making the right decision in selecting suitable medications for the diagnosed diseases. Our approach in this work offers a flexible and powerful framework to achieve this by providing solutions for knowledge representation in the domains of drugs and diseases based on their clinical and therapeutic aspects. The concept of ontology has been utilized in constructing a comprehensive knowledge framework across drugs and diseases. We perceive our proposed ontology as one of the pioneering ontologies which gathers both domains of drugs and diseases in one umbrella based on the therapeutic aspects of drugs and diseases. The DDO constructed in this thesis provides a reasonably unrestrictive framework with suitable conceptualization level that allows for a unified reasoning. Using the dependable and authentic ontologies and data bases in developing DDO supports the extensibility and adaptability of the system in future developments. The novelty of our proposed system is that it not only integrates reliable knowledge sources for the disease and drug entities, but it also overcomes the heterogeneity among these sources by applying appropriate matching techniques. As DDO has been implemented in OWL-DL, the process of reconciliation and validation of that has been performed through a Description Logic Reasoner. In particular, a query engine which has been assembled in the ontology crawler provides physicians with essential direct inquiries from DDO. Moreover, the outcome of our work has been specifically extended by employing a unique rule-based inference engine which carries out the inference process and leads to the acceptable drug recommendation. The case studies provide valuable

insights of the operational environment of the developed system and show the uniqueness of our method in facilitating the detection of reliable drug recommendations.

This research can be expanded in several ways. In this thesis, the relationship between eleven diseases and their applicable drugs which are seventeen has been built. In order to transform this system to a more realistic one, a detailed relationship among the diseases and their relevant drugs needs to be created in the future. Furthermore, to raise the usability and reliability of our drug recommendation system, more specific drug rules such as any interaction of recommended drugs with pregnancy, life style and herbs can be combined to the system. Another possible extension is the use of data mining techniques to offer prognostic drug recommendations in addition to the rule-based inference engine system that is created in this thesis. Data mining techniques evaluates preceding available cases to anticipate the drug recommendations of new cases. Comparing the drug recommendation results from data mining techniques with the ones coming from rule engine could bring up more controversial discussion or more accurate, dependable results.

References

[1] Stencel, C., Dobbins, C.: "Preventing Medication Errors." National Academies Press, News Release (2006)

[2] "Semantic Web." In: World Wide Web Consortium, Available: http://www.w3.org/standards/semanticweb/

[3] Berners-Lee, T., Hendler, J., Lassila, O.: "The Semantic Web." Scientific American Magazine. (2001)

[4] Davies, J., Lytras, M., Sheth, A.: "Semantic Web Based Knowledge Management." J. IEEE Internet Computing. 11(5), 14–16 (2007)

[5] "W3C Semantic Web Activity." In: World Wide Web Consortium, Available: http://www.w3.org/2001/sw/

[6] "What is Unicode?" In: Unicode Consortium General Information, Available: http://www.unicode.org/standard/WhatIsUnicode.html

[7] URI Planning Interest Group.: "URIs, URLs, and URNs: Clarifications and Recommendations." In: World Wide Web Consortium, Available: http://www.w3.org/TR/uri-clarification/

[8] Berners-Lee, T., Fielding, R.T., Masinter, L.: "Uniform Resource Identifier (URI): Generic Syntax." Internet RFC 3986, W3C/MIT (2005)

[9] Nandigam, J., Gudivada, V.N., Kalavala, M.: "Semantic Web Services." J. Computing Sciences in Colleges, 21(1), 50–63 (2005)

[10] Booth, D., Haas, H., McCabe, F., Newcomer, E., Champion, M., Ferris, C., Orchard, D.: "Web services Architecture." In: World Wide Web Consortium, Available: http://www.w3.org/TR/ws-arch/#XML-infoset

[11] Obtiko, M.: "Semantic Web Architecture." In: Ontologies and Semantic Web, available: http://www.obitko.com/tutorials/ontologies-semantic-web/semantic-webarchitecture.html

[12] Lassila, O., Swick, R.R.: "Resource Description Framework Model and Syntax Specification." In: World Wide Web Consortium, Available: http://www.w3.org/TR/PR-rdf-syntax/

[13] Manola, F., Miller, E., McBride, B.: "RDF Primer." In: World Wide Web Consortium, Available: http://www.w3.org/TR/rdf-primer/

[14] Shadbolt, N., Hall, W. Berners-Lee, T.: "The Semantic Web Revisited." J. IEEE Intelligent Systems. 21(3), 96–101 (2006)

[15] Antoniou, G., Harmelen, F.V.: "Web Ontology Language: OWL." Available: http://www.cs.vu.nl/~frankh/postscript/OntoHandbook03OWL.pdf

[16] Smith, M.K., Welty, C., McGuinness, D.L.: "OWL Web Ontology Language Guide." In: World Wide Web Consortium, Available: http://www.w3.org/TR/owl-guide/#owl_Class

[17] McGuinness, D.L., Harmelen, F.V.: "OWL Web Ontology Language Overview." In: World Wide Web Consortium, Available: http://www.w3.org/TR/owl-features/

[18] Palmer, S.B.: "The Semantic Web: An Introduction." Available: http://infomesh.net/2001/swintro/

[19] Gruber, T., Liu, L., Ozsu, M.T.: "Ontology." The Encyclopedia of Database Systems, Springer-Verlag (2009)

[20] Gruber, T.R.: "A Translation Approach to Portable Ontology Specifications." J. Knowledge Acquisition. 5, 199–220 (1993)

[21] Chen, R.C., Bau, C.T., Huang, Y.H.: "Development of Anti-diabetic Drugs Ontology for Guideline-based Clinical Drugs Recommend System Using OWL and SWRL." In: 2010 IEEE International Conference on Fuzzy Systems, pp. 1–6. Barcelona (2010)

[22] Jovic, A., Prcela, M., Gamberger, D.: "Ontologies in Medical Knowledge Representation." In: 29th International Conference on Information Technology Interfaces, pp. 535–540. Cavtat (2007)

[23] Rodriguez, A., Jimenez, E., Fernandez, J., Eccius, M., Gomez, J.M., Alor-Hernandez, G., Posada-Gomez, R., Laufer, C.: "SemMed: Applying Semantic Web to Medical Recommendation Systems." In: First International Conference on Intensive Applications and Services, pp. 47–52. Valencia (2009)

[24] Mya Swe, T.M., Moon Kham, N.S.: "Case-based Medical Diagnostic Knowledge Structure Using Ontology." In: 2nd International Conference on Computer and Automation Engineering (ICCAE), pp. 729–733. Singapore (2010)

[25] Chen, R.C., Chiu, J.Y., Bau, C.T.: "The Recommendation of Medicines based on Multiple Criteria Decision Making and Domain Ontology- an Example of Anti-Diabetic Medicines." In: 2011 International Conference on Machine Learning and Cybernetics (ICMLC), pp. 27–32. Guilin (2011)

[26] Iqbal, A.M., Shepherd, M., Abidi, S.S.R.: "An Ontology-Based Electronic Medical Record for Chronic Disease Management." In: 44th Hawaii International Conference on System Sciences (HICSS), pp. 1–10. Kauai (2011)

[27] "Computer-Based Patient Record Ontology." In: World Wide Web Consortium, Available:
http://code.google.com/p/cpr-ontology/

[28] Hadzic, M., Chang, E.: "Ontology-Based Support for Human Disease Study." In: 38th Annual Hawaii International Conference on System Sciences, pp. 143–149. (2005)

[29] Qu, A.X., Guidivada, R.C., Neumann, E.K., Aronow, B.J.: "Inferring novel disease indications for known drugs by semantically linking drug action and disease mechanism relationships." J. BMC Bioinformatics. 10 (5), 1–16 (2009)

[30] Pisanelli, D.M.: "Ontologies in Medicine." IOS Press (2004)

[31] Bodenreider, O.: "The Unified Medical Language System (UMLS): Integrating Biomedical Terminology." J. Nucleic Acids Res. 32(Database issue), D267–D270 (2004)

[32] De Coronado, S., Haber, M.W., Sioutos, N., Tuttle, M.S., Wright, L.W.: "NCI Thesaurus: Using Science-Based Terminology to Integrate Cancer Research Results." J. Stud. Health Technol. Inform. 107(Pt 1), 33–37 (2004)

[33] World Health Organization.: "ICD-10, International Statistical Classification of Diseases and Health Related Problems." 10th Revision. 2nd ed. Vol.3 (1998-2003)

[34] Osborne, J.D., Flatow, J., Holko, M., Lin, S.M., Kibbe, W.A., Zhu. L., Danila, M.I., Feng, G., Chisholm, R.L.: "Annotating the Human Genome with Disease Ontology." J. BMC Genomics, Vol. 10, Suppl. 1 (2009)

[35] "Human Disease Ontology." In: The Open Biological and Biomedical Ontologies, Available:
http://www.obofoundry.org/cgi-bin/detail.cgi?id=disease_ontology

[36] Davies, J., Struder, R., Warren, P.: "Semantic Web Technologies." John Wiley & Sons Ltd (2006)

[37] Cimino, J., Sideli, R.: "Using the UMLS to Bring the Library to the Bedside." J. Med. Decis. Making. l1 (4), 116–120 (1991)

[38] Lambrix, P., Habbouche, M., Perez, M.: "Evaluation of Ontology Development Tools for Bioinformatics." J. Bioinformatics. 19(12), 1564–1571 (2003)

[39] Denny, M.: "Ontology Building: A Survey of Editing Tools." Available:
http://www.xml.com/pub/a/2002/11/06/ontologies.html

[40] Angele, J., Sure, Y.: "Evaluation of ontology-based Tools." In: 13th International Conference on Knowledge Engineering and Knowledge Management, Spain (2002)

[41] "Welcome to Protégé." In: Protégé, Available: http://protege.stanford.edu/

[42] Noy, N.F., Musen, M.A.: "The PROMPT Suite: Interactive Tools For Ontology Merging And Mapping." J. International Journal of Human-Computer Studies. 59(6), 983–1024 (2003)

[43] "Pellet." In: Clark&Parsia, Available: http://clarkparsia.com/pellet/

[44] "Medical Subject Headings." In: US National Library of Medicine, Available: http://www.nlm.nih.gov/mesh/

[45] Stearns, M.Q., Price, C., Spackman, K.A., Wang, A.Y.: "SNOMED Clinical Terms: Overview of the Development Process and Project Status." J. Proc AMIA Symp. 662–666 (2001)

[46] Hamosh, A., Scott, A., Amberger, J., Bocchini, C., McKusick, V.: "Online Mendelian Inheritance in Man (OMIM), a Knowledge of Human Genes and Genetic Disorders." J. Nucleic Acids Res. 33, 514–517 (2005)

[47] Du, P., Feng, G., Flatow, J., Song, J., Holko, M., Kibbe, W.A., Lin, S.M.: "From Disease Ontology to Disease-Ontology Lite: Statistical Methods to Adapt a General-Purpose Ontology for the Test of Gene-Ontology Associations." J. Bioinformatics. 25(12), 63–68 (2009)

[48] Cure, O., Giroud, J.P.: "Ontology-based Data Quality Enhancement for Drug Databases." Available: http://www2007.org/workshops/paper_137.pdf

[49] Merrill. G.H., Ryan, P.B., Painter, J.L.: "Construction and Annotation of a UMLS/SNOMED-Based Drug Ontology for Observational Pharmacovigilance." In: Intelligent Data Analysis for Biomedicine and Pharmacology, Washington (2008)

[50] Hodge, A.E., Altman, R.B., and Klein, T.E.: "The PharmGKB: Integration, Aggregation, and Annotation of Pharmacogenomic Data and Knowledge." J. Clinical Pharmacology & Therapeutics. 81, 21–24 (2007)

[51] Hatfield, C.L., May, S.K., Markoff, J.S.: "Quality of Consumer Drug Information Provided by Four Web Sites." J. American Journal of Health System Pharmacy. 56(22), 2308–2311 (1999)

[52] Kanehisa, M., Goto, S., Hattori, M., Aoki-Kinoshita, K.F., Itoh, M., Kawashima, S., Katayama, T., Araki, M. Hirakawa, M.: "From Genomics to Chemical Genomics: New Developments in KEGG." J. Nucleic Acids Res. 34 (Database issue), D354–D357 (2006)

[53] Wheeler, D.L., Barrett, T., Benson, D.A., Bryant, S.H., Canese, K., Chetvernin, V., Church, D.M., DiCuccio, M., Edgar, R., Federhen, S., Geer, L.Y., Kapustin, Y., Khovayko, O., Landsman, D., Lipman, D.J., Madden, T.L., Maglott, D.R., Ostell, J., Miller, V., Pruitt, K.D., Schuler, G.D., Sequeira, E., Sherry, S.T., Sirotkin, K., Souvorov, A., Starchenko, G., Tatusov, R.L., Tatusova, T.A., Wagner, L., Yaschenko, E.: "Database Resources of the National Center for Biotechnology Information." J. Nucleic Acids Res. 35(Database issue), D5–D12 (2007)

[54] Wishart, D.S., Knox, C., Guo, A.C., Cheng, D., Shrivastava, S., Tzur, D., Gautam, B., Hassanali, M.: "DrugBank: a Knowledgebase for Drugs, Drug Actions and Drug Targets." J. Nucleic Acids Res. 36(Database issue), D901–6 (2008)

[55] Wishart, D.S., Knox, C., Guo, A.C., Shrivastava, S., Hassanali, M., Stothard, P., Chang, Z., Woolsey, J.: "DrugBank: a Comprehensive Resource for in Silico Drug Discovery and Exploration." J. Nucleic Acids Res. 34(Database issue), D668–72 (2006)

[56] "XML Tab." In: Protégé, Available:
http://protegewiki.stanford.edu/wiki/XML_Tab

[57] "Meditech Client Server user Manual for Physicians." In: Meditech Fraser Health, Available:
http://physicians.fraserhealth.ca/media/Reference%20Manual%20for%20Physicians%20FHA%20(April%202011).pdf

[58] "Patient Care System." In: Centre Health Sciences, Available:
http://www.ontarioshores.ca/Meditrain/PCS/index.html

[59] Kalfoglou, Y., Schorlemmer, M.: "Ontology Mapping: The State of Art." J. The Knowledge Engineering Review. 18(1), 1–31 (2003)

[60] Zhang, J., Yan, J., Fang, L., Wang, P.: "Ontology Mapping Approach based on Concept Dimensions." In: Sixth International Conference on Fuzzy Systems and Knowledge Discovery, pp. 436–440. Tianjin (2009)

[61] Yang, K., Steele, R.: "Ontology Mapping Based on Concept Classification." In: 3rd IEEE International Conference on Digital Ecosystems and Technologies, pp. 656–661. Istanbul (2009)

[62] Mascardi, V., Locoro, A., Rosoo, P.: "Automatic Ontology Matching via Upper Ontologies: A Systematic Evaluation." J. IEEE Transactions on Knowledge and Data Engineering. 22(5), 609–623 (2010)

[63] Euzenat, J., Shvaiko, P.: “Ontology Matching.” Springer-Verlag, Berlin Heidelberg (2007)

[64] Yang, F., Liu, L.: “A Flexible Approach for Ontology Matching.” In: International Conference on Computational Intelligence and Software Engineering, pp. 1–4. Wuhan (2009)

[65] Malik, S.K., Prakash, N., Rizvi, S.M.: “Ontology Merging Using Prompt Plug-in of Protégé in Semantic Web.” In: International Conference on Computational Intelligence and Communication Networks, pp. 476–481. Bhopal (2010)

[66] Lanzenberger, M., Sampson, J.: “Making Ontologies Talk: Knowledge Interoperability in the Semantic Web.” J. IEEE Intelligent Systems. 23(6), 72–85 (2008)

[67] Noy, N.F., Musen, M.A.: “PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment.” In: Proceedings of the National Conference on Artificial Intelligence (AAAI), pp. 450–455. Austin (2000)

[68] Qu, A.X., Gudivada. R.C, Jegga, A.G., Neumann, E.K., Aronow, B.J.: “A Semantic Web-based Data Representation and Reasoning Applied to Disease Mechanism and Pharmacology.” In: IEEE International Conference on Bioinformatics and Biomedicine, pp. 131–143. Fremont (2007)

[69] “Reasoner Api.” In: Protégé, Available:
http://protege.stanford.edu/plugins/owl/api/ReasonerAPIExamples.html

[70] Horridge, M., Brandt, S.: “A Practical Guide To Building OWL Ontologies Using Protégé 4 and CO-ODE Tools.” Edition 1.3, University of Manchester (2011)

[71] “Pellet.” In: Mindswap, Available:
http://www.mindswap.org/2003/pellet/

[72] “Apache Jena.” In: Jena, Available:
http://jena.apache.org/index.html

[73] “Jena Ontology API.” In: Sourceforge, Available:
http://jena.sourceforge.net/ontology/

[74] “Introduction to Jena.” In: IBM, Available:
http://www.ibm.com/developerworks/xml/library/j-jena/

[75] “Rule Engine.” In: OpenRules, Available:
http://openrules.com/

[76] Karch, A.M.: “Lippincott’s Nursing Drug Guide.” Lippincott Williams & Wilkins (2010)

[77] “Creatinine Clearance.” In: MedlinePlus of the US National Library of Medicine, Available: http://www.nlm.nih.gov/medlineplus/ency/article/003611.htm