BUILDING A CORE BIOMEDICAL ONTOLOGY ON DISEASES, BODY PARTS, SYMPTOMS AND ENVIRONMENTAL, SOCIAL, NUTRITIONAL AND DIAGNOSTIC FACTORS OF DISEASES

by

Andleeb Shahnaz

2011-NUST-MS PhD-CSE(E)-03

MS-11(CSE)

Submitted to Department of Computer Engineering

in fulfillment of the requirements for the degree of

Masters of Science

in

Computer Software Engineering

Thesis Supervisor

Dr. Usman Qamar

College of Electrical and Mechanical Engineering

National University of Science and Technology

May 2013

This page is intentionally left blank

ACKNOWLEDGEMENTS

"*Those who do not thank people, they do not thank Allah*." *(Al-Tirmidhi 1878).*

There is an old saying ‘It takes a village to raise a child’ and I am not an exception. It is with the grace of Almighty that I was led into the company of the following people, whose generosity, enthusiasm, and good shepherding sustained me in producing this work: Dr. *Shoab Ahmed Khan*, Dr. *Usman Qamar*, Dr. *Saad Rehman*, Dr. *Muhammad Abbass* and all those people who generously shared their knowledge and expertise with me. I am beholden to my family for their love and continuing encouragement. My friends always backed me up very strongly. Therefore, I thank every single one of them for their support.

To my Parents, Advisors and colleagues.

ABSTRACT

Considering today's surge of information, the need for well organized knowledge bases is increasing rapidly for providing simplified access to knowledge and its further processing. In biomedical domain, heaps of information is buried in scientific publications and online forums. This calls for representing this information in a more expressive semantic way by determining and storing relational information into a machine readable form. So, the primary goal of this research endeavor has been to build a knowledge base on entities and relations containing amass of formalized background knowledge suitable for supporting reasoning in biomedical domain.

In this work, we introduce a way for easily accessing the knowledge about body parts and symptoms of human diseases, along with environmental, social, nutritional and diagnostic factors that cause these diseases. The information for this knowledge base is extracted from the controlled vocabulary thesaurus "Medical Subject Headings" (MeSH), which is published by National Library of Medicine.

The result is a semantic graph of typed entities and relations between diseases, their symptoms, affected body parts and determining factors, with emphasis on environmental, social, nutritional and diagnostic factors. The facts stored in our ontology are provided to the user in a visual web interface.

Currently, our ontology contains 53020 individuals, 96835 synonymous terms and 197731 facts related to seven pre-defined categories of our biomedical ontology. In this way, it fulfils an identified need to provide detailed semantic knowledge regarding different biomedical sub-domains at one place through one core KB.

Keywords – *Biomedical ontology, Ontology creation, Knowledge integration, Semantic knowledge, Biomedical sub-domains*.

TABLE OF CONTENTS

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviation	**Illustration**
ACM	Association for Computing Machinery
GAD	Genetic Association Database
GO	Gene Ontology
IEEE	Institute of Electrical and Electronic Engineers
KB	Knowledge Base
KEGG	Kyoto Encyclopedia of Genes and Genomes
MeSH	Medical Subject Headings
MIPS	Mammalian Protein – Protein Interaction Database
NIH	National Institute of Health
NLM	National Library of Medicine
OMIM	Online Mendelian Inheritance in Man
OWL	Web Ontology Language
PHSkb	Public Health Surveillance Knowledge Base
RDBMS	Relational Database Management System
RDF	Resource Description Framework
SQL	Structured Query Language
UMLS	Unified Medical Languages System
US	United States
W3C	World Wide Web Consortium
XML	Extensible Markup Language

Chapter 1
INTRODUCTION

Where is the wisdom we have lost in knowledge?

Where is the knowledge we have lost in information?

('Choruses from the Rock' by T.S.Eliot - 1934)

Knowledge is of no use if it is not represented and shared in a quality way. Sharing and passing knowledge helps in achieving the advancements without reinventing the wheel and thus results in advancement of humanity as a whole. In past few years, amount of research work has grown to great lengths in some fields. Especially, a lot of researches have been carried out in biomedical domain and spreading of this knowledge to masses, made an important contributory step towards civilization. But most of this knowledge is present in scientific publications and only very few people know how to access it. If this knowledge can be made accessible to the common people then a lot of advantages can be achieved including, diseases' prevention, early and more accurate disease diagnosis, more effective treatment and many more".

1.1 Background and Motivation

Based on the huge pile of health information available on the internet, web has the potential of being the ultimate encyclopedic source. But effective retrieval of required results from web has always been problematic, due to which we are still far from exploiting this potential [1]. Users have to undergo vast amount of difficulties in finding the exact precise information from this huge pile of health data. The existing generic search engines (e.g. Google), generic catalogues (e.g. Yahoo) and free text based search engines have not been capable of solving this issue [2]. Most of the time, the results returned by them are innumerable and completely irrelevant, which require a lot of manual working on user's part. A major reason behind this incapability of search engines is lack of semantic organization of data available on the web [3]. So, when the size of the web expands exponentially, it then becomes difficult to retrieve the useful information due to this lack of organization.

Especially, most of the information in biomedical research area is not yet captured in databases today, but rather present in structured form in scientific publications and also in semi-structured or unstructured form on web. If this knowledge gets effectively provided to the population then many diseases can be prevented, diagnosed earlier, and more accurately, and thus treated better, and cured more effectively; even epidemics and pandemics could be avoided.

Researchers have long been trying to find the solution to this problem and according to them [1] [4] [5] [6] [7]; the best possible way to solve this issue is to move to the concept of semantic web. But due to the diversity of terms and their definitions between groups; adding semantics to web is not a straight forward task. It asks for achieving a shared and common understanding of a domain. So, this leads us to the creation of an ontology, which can be applied to various contexts for variety of purposes. Ontologies serve as the backbone of semantic web by facilitating knowledge exchange across people and application systems [3][4].

In recent years, very large common-sense knowledge bases (KB) have been generated automatically. They were built by extracting entity-relationship-oriented facts from Web sources. Examples thereof are the large collaborative KB Freebase as well as True Knowledge, providing a question-answering platform on the commercial side. On the research side, DBpedia and YAGO constitute well developed representatives, both containing factual information extracted from the Wikipedia. All of these have formal knowledge representations, using the Resource Description Framework (RDF) data model. The development of RDF started as a project of the World Wide Web Consortium (W3C) in the 1990s. In the early 2000s, the first complete implementation of RDF was published and it has prevailed as a general method for conceptual description or modeling of information implemented in Web sources. Further improvements and developments on RDF are undertaken by W3C. RDF provides a graph-based data model, making statements about resources in the form of so-called RDF triples, “subject-predicate-object”. For example, the statement “Asthma has the symptom coughing” can be presented in RDF as the triple “asthma” denoted by a subject, “has the symptom” by a predicate, and “coughing” by an object. An RDF-based data model is more suited to certain kinds of knowledge representation than the relational model, as a collection of RDF statements generally is represented as a labeled and directed multi-graph. Thus, they can be deployed as semantic services such as question answering, reasoning and explanation, and knowledge discovery.

In the biomedical domain, in fact, there already are plenty of biomedical knowledge collections available, conveying at least one of these two following kinds of characteristics:

1) They solely focus on highly specialized aspects, e.g., protein interactions, gene expression, and metabolic pathways, but lack general determinants. Some examples of these general determinants include environmental factors (e.g. noise pollution), social factors (e.g. Adolescent Behavior, Identity crisis, huger, etc.), nutritional factors (e.g. Proteins, Deoxyglucose, etc.) and diagnostic factors (e.g. Angiography, Autopsy). This kind of knowledge collection, targeting specialized aspects, is covered by The MIPS Mammalian Protein-Protein Interaction Database, Gene Ontology, or Kyoto Encyclopedia of Genes and Genomes among others.
2) In essence, most of these mentioned collections are hand-crafted and extensively curated by human experts.

But these features are not suitable for our purpose which is to gather information regarding environmental, social, nutritional, and behavioral factors of diseases as well, besides just collecting the knowledge about diseases, symptoms and body parts. A biomedical ontology would be more useful if it contains semantically rich knowledge regarding different sub-domains e.g. information about diseases, their symptoms, and contributing factors; all in one place. Such ontology would have to be of high quality and be based on an authentic information source. It would have to consist of not only the concepts and named entities, but also relations among them like, subclassof, typeof and means etc. Along with being extensible and reusable, it would have to be application independent as well. Availability of such an application can really open the horizon towards more effective and useful applications in biomedical domain.

Thus, we want to dedicate ourselves to the accessibility of the medical expertise for the populace by building a biomedical knowledge base on entities and relations regarding diseases, their symptoms, affected body parts and determining factors, with emphasis on environmental, social, nutritional and diagnostic factors.

1.2 Objective and Contribution

This thesis is aimed to build up an ontology that is based on a widely trusted biomedical vocabulary thesaurus named MeSH. We intend to create such a KB that contains information regarding body parts and symptoms of human diseases, along with environmental, social, nutritional and diagnostic factors that cause these diseases.

Rather than using the information extraction methods, our approach is to make use of the fact that MeSH has tree numbers for each of its vocabulary term. Tree numbers describe the level of the term in the taxonomy and make it quite easy to access the taxonomic or hierarchical details regarding that term. These tree numbers help us gather concepts, entities and relations keeping in view the MeSH classification. For usefulness of an ontology, one of the basic requirements is to arrange the concepts in taxonomy. Though MeSH also has a hierarchy but it cannot be used as it is for our ontological needs, because the entities which we want to gather for our KB categories are spanned to more than one category in MeSH hierarchy. For example: constituent entities for "Nutritional Factors" category of our KB are spanned in three MeSH categories named, Chemical and Drugs, Phenomena and Processes, and Food and beverages. Therefore we need to re-classify the vocabulary terms of MeSH. Our KB is based on data model of entities and binary relations. Three categories of relations which we want to collect include subclassof, typeof and means relationship.

So, our contribution is two-fold:

1) A core Ontology: We describe how we integrated data from different MeSH categories to obtain our core ontology as the storage backend for our system. The structure of our KB follows the RDF data model and is inspired by YAGO.
2) Providing an access point to the knowledge base: The ontology can be queried comfortably by using the web browsing interfaces. The browsing interface takes a single query input and returns all relations with the respective entities stored in the KB.

1.3 Outline

Chapter 2 discusses the related work. In chapter 3, we describe in detail the knowledge extraction approach that we utilized for gathering the data from MeSH thesaurus. We present the design and construction of backbone of our system, the KB, in chapter 4. Chapter 5 presents an overview of the visual interface that was developed for accessing the knowledge base, contains discussion on the results of our system and also presents the potential applications of this knowledge base in different research areas. Finally, chapter 6 concludes this thesis and proposes an outlook of possible extensions, modifications, and improvements as future work.

1.4 Summary:

The problems of effective retrieval of required results from web and lack of availability of biomedical research information in databases have long been prevailing in the research community. Researchers have long been trying to find the solution to this problem and proposed the concept of semantic web as a solution for them. But due to the diversity of terms and their definitions between groups; adding semantics to web is not a straight forward task. It asks for achieving a shared and common understanding of a domain. So, this leads us to the creation of an ontology, which can be applied to various contexts for variety of purposes. In the biomedical domain, there already are plenty of biomedical knowledge collections available, but none of them provides the knowledge regarding different biomedical sub domains at one place. This is what served as a motivation for this research work.

Chapter 2

LITERATURE REVIEW

In this chapter, we give a brief overview on the state-of-the-art of existing knowledge collections. We considered the work on the general domain as well as on the more specific ones, especially the biomedical domain. We describe their functionality and elucidate how they lack the certain level of details regarding different causative factors of diseases and what is lacking in them which is provided through our KB.

2.1 Overview

Knowledge representation and capturing relationships has long been a topic of interest in the field of artificial intelligence. Since Cyc in 1980s, a lot of knowledge bases have been built which differ from each other on bases of varying factors, most common of which are size and target domains.

In recent years, a lot of common sense knowledge bases have been built up automatically by extracting entity-relationship targeted facts from web sources. Facts can be described as instances of unary, binary or higher-artery relationships. Where, these three kinds of relations can be described as:

- Unary Relations: categorization of individual entities into semantic classes
- Binary Relations: typed relations between entities with binary relations' instances
- Higher-Artery Relations: non-binary relations

For our KB, we have focused on binary relations which connect two entities that are related in a predefined way.

2.2 General Domain Knowledge Bases

In general domain, the work on Knowledge bases (KBs) can be categorized into two parts, i.e. industrial research side and academic research side [6]. Some of the examples of well developed KBs in industrial research side include Freebase [7] and TrueKnowledge [8]. Both of these are based on entity relationship oriented facts from varying web sources [9].

Freebase contains a huge structured data collection regarding individuals of many miscellaneous domains, like, music, media, location, literature. It takes its information from varying web sources including info boxes of Wikipedia, free text, online news centers and even approved entries of its community members.

True knowledge provides a flexible question answering platform to its users. It collects the knowledge from user submissions and also from external databases, like Wikipedia. Both true knowledge and freebase cover almost same kind of domains, but the major difference among them lies in the ways of access provided by them. In contrast to Freebase, True knowledge provides a user friendly way of accessing the knowledge i.e. through natural language access. For example, in response to user query "*When was Michelle Obama born?*" as well as for the query "*Michelle Obama birth date*", the user is provided with the correct result "*January 17th 1964*".

Both KBs, i.e. Freebase and True Knowledge, contain partial biomedical domains' information. If the user tries to query the Freebase for gathering information on "*Dyslexia*", then he just gets a list of some of the symptoms and risk factors, as shown in table 2.1. Alternatively, True Knowledge allows its user to search through queries like "*What causes Dyslexia?*", "*What are risk factors of Dyslexia?*" and "*What are symptoms of Dyslexia?*". The answers are shown in table 2.1. It is evident from the outputs from both of these databases, that they provide details regarding just a few triggers and symptoms, while causes were completely omitted in True Knowledge.

Table 2.1: Results to example query "Dyslexia" in Freebase and True Knowledge. (The query was performed in 2013)

	Risk Factors	Causes	Symptoms
Freebase	– Family History of Dyslexia	- -	– Speech Disorder – Delayed reading ability
True Knowledge	- - -	- - -	– Speech Disorder – Delayed reading ability

In the same way, the results for query example “asthma” are shown in Table 2.2. All of the results detailed in this table are just risk factors and symptoms. Only a few causes were provided by Freebase, and they also were just from the chemical domain.

Table 2.2: Results to example query “asthma” in Freebase and True Knowledge. (The query was performed in 2013)

	Risk Factors	**Causes**	**Symptoms**
Freebase	– Filipino American – Native Hawaiians – Passive smoking – Puerto Ricans in the United States – Overweight – African American – Poverty – Exposure to allergens – Family history of asthma – Small for gestational age – Family history of atopic disease	– Chronic Inflammation of airways – Reversible bronchoconstriction	– Dyspnea – Wheeze – Cyanosis – Pectus carinatum – Short stature – Pulsus paradoxus – Cardiac arrest – Bronchospasm – Chest Tightness – Nocturnal Cough
True Knowledge	– Adenosine – Histamine – Mannitol – Hippuric acid – Cotinine – Adenosine phosphosulfate – Cholesteryl	- - -	– Cough – Cyanosis – Wheez – pectus carinatum – Bronchospasm – Cardiac arrest – Nocturnal Cough – Pulsus paradoxus – short stature – Dyspnea – Chest Tightness – Chest hyperinflation – Chest expansion poor

Both of these above described query examples as well as many other queries that we have performed on both of these databases, have shown that, these databases do not provide information regarding different causative factors of diseases, including environmental, social, nutritional and diagnostic factors. Freebase provides few environmental factors for some diseases, like it provided an environmental factor “*exposure to allergens*” for query example “*asthma*”, but omits the remaining factors completely. On the other side, True Knowledge completely omits all these factors. Besides that, another issue that was found in True Knowledge was of repetitive result entries, e.g. in the output of query example “*asthma*”, it shows the symptom “*Nocturnal Cough*” six times, which shows a lacking in True Knowledge KB on data cleaning part.

On academic research side, the examples of well developed and well kept KBs are DBpedia [10] and YAGO [11] [12] [13] [14]. Both of these are based on RDF subject-property-object triples from web sources like Wikipedia.

DBpedia provides structured information regarding people, planet, color, language, etc. Its gathers this knowledge from Wikipedia info boxes in up to 97 different languages by utilizing the rule based parsing techniques.

YAGO is a joint research project aimed at extracting structured information regarding people, organization, cities, etc. This extensible and large ontology has been built up by unifying entities and facts which were derived from both Wikipedia and WordNet. The relational knowledge for YAGO were automatically extracted from the category system and info boxes of the Wikipedia and unified with taxonomic relations from WordNet.

YAGO2 is an extension to YAGO and it has incorporated another knowledge source as well, which is GeoNames. Besides that, entities, facts and events in this KB are additionally anchored in both time and space. For example, the result of performing a query “*Albert Einstein*” in YAGO2 also provides the details regarding date of birth and place, like, *wasBornOnDate 1879-03-14* and *wasBornIn Ulm*.

All of the above described knowledge bases are based on RDF data model so they serve as a very useful asset for semantic services, like reasoning and explanation, knowledge discovery, and question answering.

Considering the vast amount of research work in this area, we can present only a few most meaningful and recent ones. For a more detailed overview, we refer the reader to the paper titled, “From Information to Knowledge: Harvesting Entities and Relationships from Web Sources” by Weikum et al.

2.3 Biomedical Domain Knowledge Bases

Judging the effectiveness of KBs in other research fields, several have already been developed in biomedical domain as well. Most of them are well developed and well kept by United States National Library of Medicine (NLM), which is one of the 27 institutions of the U.S National Institute of Health (NIH). However, most of the existing biomedical collections contain information regarding just some specialized areas like protein to protein interactions, gene expressions, etc. or only provide some crude taxonomy that lacks the semantic relations. Examples of such collections which are focused on a specialized domain, include:

- PPI [15]: The Mammalian Protein to Protein Interaction database is a manually curated collection by Munich Information Center for Protein Sequences (MIPS),and is based on data that is extracted by expert curators from existing research material.
- GO [16][17]: GeneOntology is a structured and controlled collection of vocabularies and terms which represent the genes and genes product properties across all species.
- KEGG [18][19]: Kyoto Encyclopedia of Genes and Genomes is developed mainly for providing information regarding gene functions.

Examples of collections which have a taxonomic detailing, but lack useful semantic relations, include:

- MeSH [20]: Medical Subject Headings is a poly hierarchical vocabulary thesaurus, which is developed for indexing the articles for Pubmed/MEDLINE database [21].
- UMLS [22]: Unified Medical Languages System is a collection of biomedical vocabularies which is developed from MeSH, GO and OMIM along with many other sources.

Besides this, some research work has also been carried out regarding relations between diseases and their contributing factors. This research has been conducted in different contexts, including relations between genotype and drug response phenotype [23], disease gene associations, and also disease and etiological factors [24]. Examples of such collections are:

- OMIM [25][26]: Online Mendelian Inheritance in Man provides up to date information regarding human genes, heritable diseases and genetic disorders.
- GAD [27]: Genetic Association Database provides information regarding associations between human genes and complex diseases and disorders.
- Diseasome [28][29]: Focuses on phenotype and disease gene associations and OMIM is one of its information sources.

Besides these KBs, another one named Public Health Surveillance Knowledgebase (PHSkb) is aimed at supporting diseases surveillance by providing easy access to knowledge.

Only a small subset of trusted existing biomedical KBs is described above. Most of these are hand crafted and are manually curated by human experts. Although, a lot of information extraction techniques like pattern matching, natural language processing and statistical learning [30, 31, 32, 33, 34, 35, 36] have been introduced recently for creating high quality ontology, but they have still not been able to surpass the quality of manually curated ones. However the manually integrated KBs have to undergo the challenges of low coverage, high integration cost, quality assurance and fast aging [11].

One thing that is obvious from the above described lists of biomedical KBs is that there is no such ontology that can provide the complete detailed information regarding most of the main factors (including, environmental, social, nutritional and diagnostic factors) of a disease at one place without focusing on a specialized context. If anyone has to access all of these varying factors, then he must have to rely on more than one KB, as no core KB having such integrated knowledge about contributing factors of diseases have yet been built. And one of the major issues that can arise while utilizing more than one KBs together is the mapping of diversified entities covered in KBs. So we dedicate ourselves to this problem by building up a core biomedical knowledge base containing all of these major causative factors of a disease through one KB.

A summarization of main characteristics of all the above described KBs is presented in Table 2.3.

2.4 Summary

After going through an overview of the state-of-the-art of existing knowledge collections of general as well as biomedical domain, it gets clear that none of them is capable of providing certain level of details regarding different causative factors of diseases at one place through one core KB, without focusing on a specialized context. So, this leads us to the idea of creating a core KB, that can provide integrated knowledge regarding, diseases, symptoms, body parts and causative factors (environmental, social, nutritional, diagnostics factors) of diseases.

Table 2.3: Summarization of characteristics of major existing knowledge bases

KB	Domain	Source	Size
		General Domain	
Freebase	Music, literature, location, media, etc	– Wikipedia free text and info boxes – online news centers – community contributions – other domain specific pages	Entities: > 10 million Facts: > 358 million
True Knowledge	Music, literature, location, media, etc.	– Wikipedia – WordNet – GeoNames – Community contributions	Entities: >13.3 million Facts: >437.8 million
DBpedia	People, Eucaryotes, Disease, Planet, Color, Language, Event, Award, etc.	Wikipedia in up to 97 languages	Entities: >3.5 million Facts: >672 million[1]
YAGO	People, Organizations, Cities, etc.	– Wikipedia – WordNet	Entities: >10 million Facts: >80 million
YAGO2	Domain of YAGO + time, place	– Wikipedia – WordNet – GeoNames	Entities: ~9.8 million Facts: >80 million
		Biomedical Domain	
PPI db of MIPS	Protein to protein interaction	Scientific literature	N.A
GO GenBank	Genes, gene product genes	Scientific publications, direct submissions from individual laboratories, bulk submissions from large-scale sequencing centers	N.A
KEGG	Genomes, Enzymatic pathways, chemicals	N.A	>11 mil genes, ~134,000 pathway maps, 375 human diseases, 9.336 drugs
OMIM	genes, genetic disorders, including phenotype description and body parts	Manually generated by scientists and physicians	~18,597 genes
MeSH	General medical subjects for indexing articles for PubMed database	PubMed publications	~25,186 entities
Diseasome	Human disease network	OMIM	>4,213 diseases, >91,182 genes
UMLS	General biomedicine	GO, OMIM, MeSH, etc	Entities: >1 million Facts: >12 million

[1] 286 million of these 672 million facts were extracted from English Wikipedia, while remaining 386 million were gathered from other languages

Chapter 3

DATA GATHERING

3.1 Problem Definition

The first and most important step for effective implementation of an ontology is to make-ready the data. This chapter will focus on identifying our data source and its attributes. Besides, the problem of data reliability will also be addressed here.

Good data leads to good results and bad data is always misleading. Especially for building an ontology, the quality and reliability of data matters a lot, as this is what eventually results in effectiveness, reliability and coverage of an ontology. The data we begin with takes the form of an input and this input serves as the backbone of ontology. Our target requirement out of this data is to present it in our ontology in such a semantic way that it triggers a learning process that describe the concept that is '*intelligible'* in that it can be understood, discussed, and disputed; '*operational'* in that it can be applied to actual examples. With this in mind, vast varieties of biomedical data sources publically available were considered. But after careful preliminary analysis, research was narrowed down to one data source named MeSH, controlled by NLM. This data source was selected because it is free to use, well documented, well updated and well maintained with huge set of attributes for a pile of diseases. Moreover, accuracy and reliability of entities recorded from this source is very high because it is a medical vocabulary thesaurus that is well controlled by NLM and contains medical subjects in a well organized hierarchical structure. All these advantages have made our knowledge and relation extraction easy.

Considering the availability of a few other reliable sources like UMLS and OMIM, we could have gone a bit further by incorporating data from these sources as well and would have extended the coverage of our ontology, but effective knowledge modeling of our ontology was preferred over increasing the sample-space and incorporating the data from these other sources into our well modeled and well based knowledge base is now left as a future work.

Data issue is bit tricky; therefore, we will consider the data problem in two stages. First stage is how to retrieve the data from MeSH and second is how to manage retrieved data in a Relational Database (RDBMS), while maintaining our ontological semantic requirements, i.e. extracting the relationships between gathered entities. This chapter will focus on describing the first stage only, and second stage will be explained in next chapter named “Knowledge Base”.

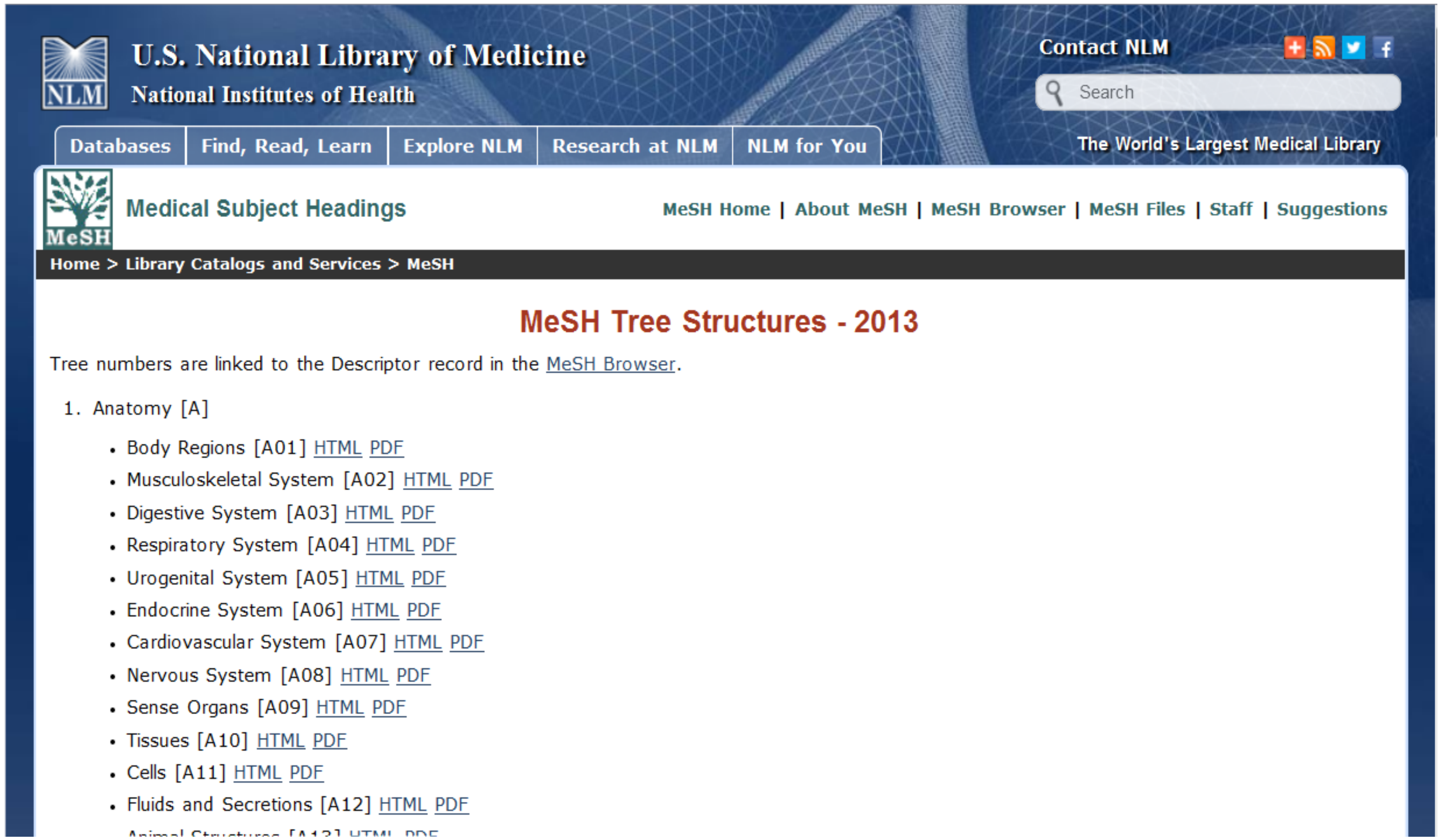
Without going in to the details of data taxonomy that is needed in our ontology, we will focus on the problem of gathering the data from MeSH.

MeSH [20] is a controlled biomedical vocabulary thesaurus that is published and continuously revised and updated by NLM. It was created by NLM, more than 40 years ago, for indexing and searching MEDLINE database of journal articles. It enables retrieval systems, such as NLM's PubMed, to provide subject searching of the data. MeSH consists of sets of terms naming descriptors in a hierarchical structure that permits searching at various levels of specificity. MeSH descriptors are arranged in both an alphabetic and a hierarchical structure. At the most general level of hierarchical structure are very broad headings such as "Anatomy" or "Mental Disorders." More specific headings are found at more narrow levels of the twelve-level hierarchy, such as "Ankle" and "Conduct Disorder." There are 26,853 descriptors in 2013 MeSH. All of this MeSH information can be accessed through hierarchically maintained MeSH Tree Structure available at their official website, whose snapshot and web link is mentioned in figure 3.1. All of the detailed information regarding each constituent descriptor of this MeSH Tree is described in descriptor records, as depicted in figure 3.2.

Some of the most common and important constituents of these descriptor records are presented and explained as under:

1) MeSH Heading: A preferred unique term for the descriptor that is used to represent it in other elements
2) Unique Id: Seven-character alpha-numeric string uniquely identifying a descriptor
3) Tree Number: Alpha-numeric string referring to location of a term naming descriptor within a MeSH Descriptor hierarchy, along with providing a way to access the details of all of its root descriptors. For example, the tree number for body region foot is “A01.378.610.250”. This tree number lets you access the details of all of its root

descriptors by telling you the tree number for all of those root descriptors, as depicted in table 3.1.

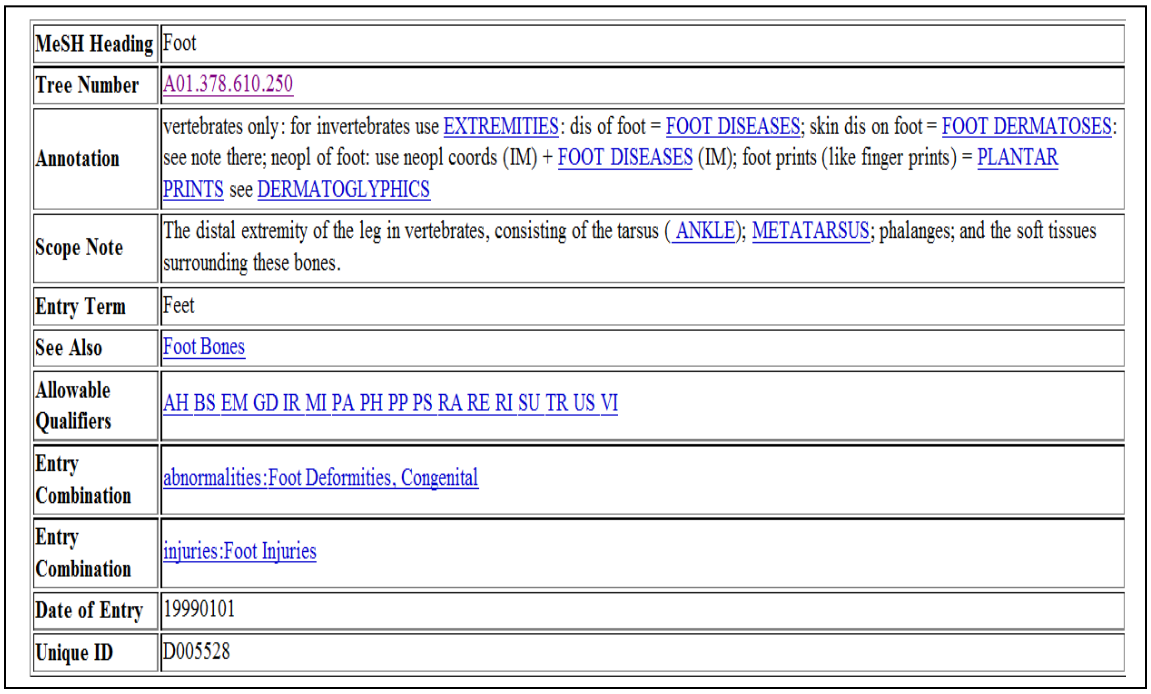

U.S. National Library of Medicine
National Institutes of Health
Contact NLM
Search
Databases
Find, Read, Learn
Explore NLM
Research at NLM
NLM for You
The World's Largest Medical Library
Medical Subject Headings
MeSH Home | About MeSH | MeSH Browser | MeSH Files | Staff | Suggestions
Home > Library Catalogs and Services > MeSH

MeSH Tree Structures - 2013

Tree numbers are linked to the Descriptor record in the MeSH Browser.

1. Anatomy [A]
   - Body Regions [A01] HTML PDF
   - Musculoskeletal System [A02] HTML PDF
   - Digestive System [A03] HTML PDF
   - Respiratory System [A04] HTML PDF
   - Urogenital System [A05] HTML PDF
   - Endocrine System [A06] HTML PDF
   - Cardiovascular System [A07] HTML PDF
   - Nervous System [A08] HTML PDF
   - Sense Organs [A09] HTML PDF
   - Tissues [A10] HTML PDF
   - Cells [A11] HTML PDF
   - Fluids and Secretions [A12] HTML PDF

Figure 3.1: A snapshot of MeSH Tree Structure, Accessed on 5/6/2013 (http://www.nlm.nih.gov/mesh/trees.html)



MeSH Heading	Foot
Tree Number	A01.378.610.250
Annotation	vertebrates only: for invertebrates use EXTREMITIES: dis of foot = FOOT DISEASES; skin dis on foot = FOOT DERMATOSES: see note there; neopl of foot: use neopl coords (IM) + FOOT DISEASES (IM); foot prints (like finger prints) = PLANTAR PRINTS see DERMATOGLYPHICS
Scope Note	The distal extremity of the leg in vertebrates, consisting of the tarsus (ANKLE); METATARSUS; phalanges; and the soft tissues surrounding these bones.
Entry Term	Feet
See Also	Foot Bones
Allowable Qualifiers	AH BS EM GD IR MI PA PH PP PS RA RE RI SU TR US VI
Entry Combination	abnormalities:Foot Deformities, Congenital
Entry Combination	injuries:Foot Injuries
Date of Entry	19990101
Unique ID	D005528

Figure 3.2: MeSH Descriptor Record of Body Region, Foot

Table 3.1: Tree numbers of all Root Descriptors extracted from Tree Number of one MeSH Descriptor, i.e. Foot

MeSH Hierarchy Level	Tree Number	MeSH Heading
Level 1	[A01]	Body Regions
Level 2	[A01.378]	Extremities
Level 3	[A01.378.610]	Lower Extremity
Level 4	[A01.378.610.250]	Foot

This whole process of chunking out the tree numbers of all root descriptors is explained in detail in Figure 3.3.

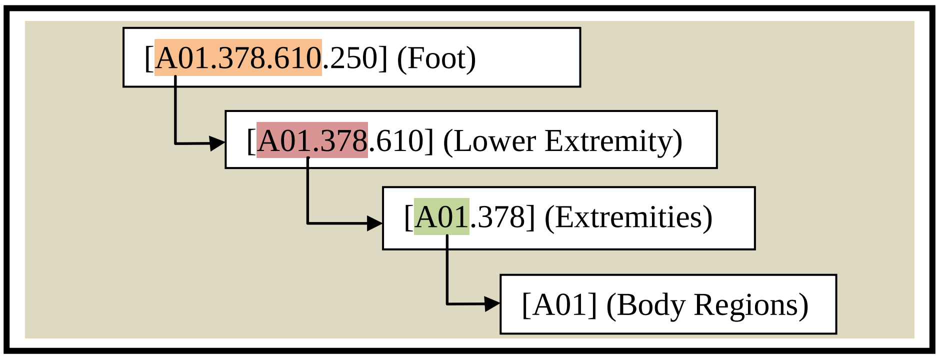


Figure 3.3: Chunking Tree numbers of root descriptors from tree number of a MeSH descriptor named "Foot"

Tree numbers are also used for browsing the MeSH vocabulary and for inclusive searches by retrieval systems using MeSH. Moreover, MeSH follows the phenomena of polyhierarchy, so according to this, a descriptor can have more than one Tree Numbers [37], [38], [39], which means that a descriptor can belong to more than one category at a time.

4) Annotation: Contains free-text information regarding a term naming descriptor
5) Entry Term: Entry Terms in MeSH are names of substances that are considered equivalent to a term naming descriptor for retrieval purposes. They are provided in descriptor records, if exist.
6) See Also: Free-text element which refers a user from a Descriptor to other terms which have related roots

MeSH provides us the data in two ways; in well maintained and well documented XML files; and through MeSH online browser. First of all we tried to extract data from XML files through different tools, but did not get successful in that, because of two main issues:

1. MeSH data and its attributes that we needed to extract were scattered in different MeSH XML files, which were quite large in size. None of the existing XML files reading/ editing tools had even been able to successfully read such large files, let alone edit them.
2. Besides that, we needed to extract data with its attributes according to our hierarchical needs. For example, we wanted to collect data from a few manually decided subclasses of MeSH category "*Diseases*" and combine it with one of the subclass of a MeSH category named "Psychiatry and Psychology". Gathering data in such a way from the predefined MeSH taxonomy was quite difficult to do with any tool.

So, we opted for crawling MeSH data from the MeSH tree structure available through their official website, whose snapshot and web link are already depicted above in figure 3.1. We programmed a customized web crawler for this task and extracted the required data with its attributes from that browser. At this stage, one MeSH attribute that helped us a lot in crawling the required data effectively, is MeSH "*Tree Numbers*". MeSH taxonomy is maintained by this "*Tree Number*" attribute, as each MeSH entry term has this attribute and it shows the level of occurrence of that term in MeSH hierarchy. We build up our crawler logic around this fact and crawled data by following this "*Tree Number*" attribute. All of the crawled data was gathered in an SQL database table. The data from this table was then utilized for second data stage that was mentioned earlier in this chapter, i.e. how to manage retrieved data in a Relational Database (RDBMS), while maintaining our ontological semantic requirements, i.e. extracting the relationships between gathered entities. This second stage is deferred till next chapter.

3.2 Tools – Microsoft SQL Server 2008, Microsoft Visual Studio 2010

Microsoft SQL server 2008 database served as the backend of our customized crawler and the actual crawler was programmed through Microsoft Visual Studio 2010, in C#.

A code snippet for the customized data crawler that was written as a console application in .Net is presented in Appendix A.1.

3.3 Summary

Good data leads to good results and bad data is always misleading. Especially for building an ontology, the quality and reliability of data matters a lot, as this is what eventually results in effectiveness, reliability and coverage of an ontology. The data we begin with takes the form of an input and this input serves as the backbone of ontology. Keeping this in mind, we selected the most trusted biomedical thesaurus MeSH as our Information source and collected all of the required biomedical entities' details from its web browser into an SQL database, through our customized crawler.

Chapter 4

KNOWLEDGE BASE

The main aim of this research endeavor was to build up a core biomedical ontology that provides information regarding diseases, body parts, symptoms of human diseases, along with environmental, social, nutritional and diagnostic factors that cause these diseases. So, in this chapter, we present a detailed description of structure of our KB and then explain that how we integrated data from different MeSH categories to obtain our core ontology as the storage backend for our system. And then finally, the data cleaning step for our KB is presented.

4.1 Overview

We built up our biomedical ontology in Microsoft SQL server 2008. A formal definition and description of the knowledge structure of our KB is described as under.

4.1.1 Problem Definition

We aim to build up such a biomedical ontology that contains following information:

- Individuals: Including candidates for body parts, symptoms, diseases, environmental factors, social factors, nutritional factors and diagnostic factors
- Classes: all of the individuals are arranged in a hierarchy and are connected to each other through classes
- Relations: Individuals are linked to each other through three kind of binary relations, including, subclassof, typeof and means relations.
- Facts: Facts about individuals are recorded in our KB. Whenever two individuals are connected through a relation then the resultant is called a fact. The constituent individuals of a fact are called its arguments. Some examples of the kind of facts that shall be included in our KB are listed in Table 4.1.

Table 4.1: Facts Example

Argument 1	Relation	Argument 2
Dyslexia, Acquired	subclassof	Delirium, Dementia, Amnestic
Dyslexia, Acquired	subclassof	Cognitive Disorders
Dyslexia, Acquired	subclassof	Dyslexia
Dyslexia, Acquired	typeof	Communication Disorders
Dyslexia, Acquired	typeof	Language Disorders
Dyslexia, Acquired	typeof	Learning Disorders
Dyslexia, Acquired	typeof	Neurobehavioral Manifestations
Dyslexia, Acquired	means	Reading Disability, Acquired
Dyslexia, Acquired	means	Alexia, Acquired
Dyslexia, Acquired	means	Word Blindness, Acquired

4.1.2 The Knowledge Model

Knowledge representation has long been a topic of interest in AI and many models have been proposed up till now ranging from Frames, KL-ONE to description logics, RDFS and OWL [37][38]. Among these existing data models, web ontology language (OWL) and its basis RDFS are considered the state-of-the-art formalism in knowledge representation. Considering this, we designed our KB in SQL, following the RDF-format, as it was adapted in YAGO [11].

YAGO expresses the entities and relations between entities, i.e. facts, in a consistent RDF style semantic graph. According to knowledge model of YAGO, all objects are expressed as entities, which are further grouped into predefined classes. These entities are then linked through a type relation between classes and their instances and also by subclassof relation between class and subclass. Besides that, binary relation can hold between two entities.

For example, to say this that "*Dyslexia, Acquired* is a sub class of the class named *Cognitive Disorders*", the recorded binary relation will be, "*Dyslexia, Acquired*" stands in "*subclassof*" relation with the entity "*Cognitive Disorders*". This triple of relation and entities is called fact, as presented below:

subClassOf(Dyslexia, Acquired, Cognitive Disorders)

The two entities which are part of a fact are called arguments of the fact.

4.1.3 Contribution

Our KB has two main tables named, individuals and facts. All of the distinct entities are included in individuals table, while all of the facts, connecting these individuals through relations, are recorded in facts table. Besides that, we also introduced another table named synonymous terms, which according to its name is meant to record all of the possible naming variants of an entity.

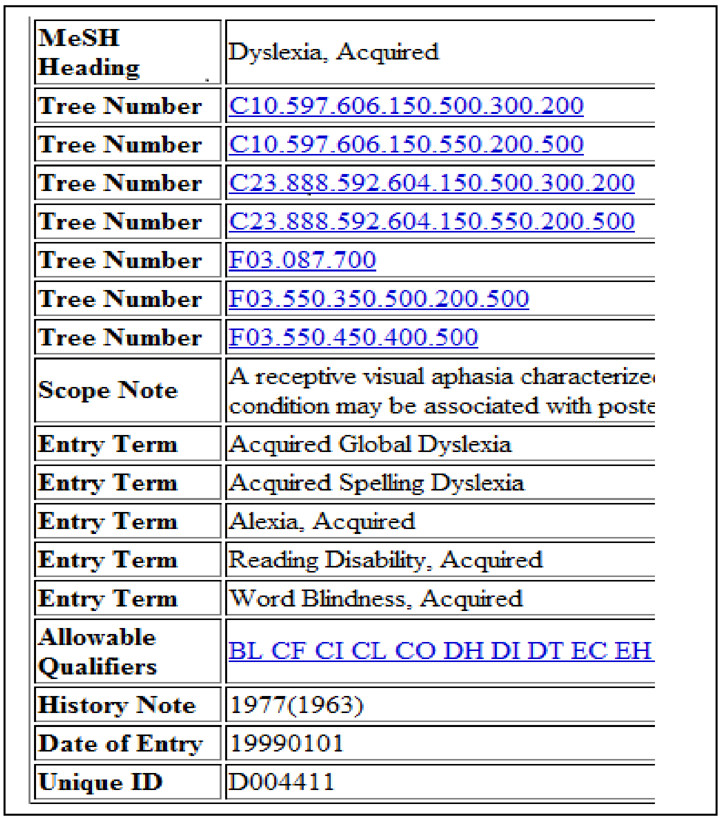
The synonymous terms are mostly missing in most of the KBs and hence affect the effectiveness and usage of those KBs to a great extent, in text mining tasks like, named entity recognition. This issue arises because of the diversity of biomedical entry terms. So, by gathering and recording these synonymous terms in our KB, we aim to cope with this issue, so that our KB does not have this limitation and it can then be utilized for achieving effective named entity recognition in text mining tasks. This will result in improving the effectiveness and usage of our KB.

4.1.4 Sources

Our KB contains information regarding different sub domains. Along with diseases and its symptoms, it also provides details regarding different causative factors of diseases, including environmental, social, nutritional and diagnostic factors. Besides the normal English representations, the special names and even codes are also recorded in our KB.

As we wanted to extract all of the information regarding diseases and all of these variant factors, so no other source could have served as a better starting point for our KB than a biomedical thesaurus like MeSH. Therefore, we chose MeSH to be the backbone of our KB. MeSH [20] is a controlled biomedical vocabulary thesaurus that is published and continuously revised and updated by NLM. It was created by NLM, more than 40 years ago, for indexing and searching MEDLINE database of journal articles. It enables retrieval systems, such as NLM's PubMed, to provide subject searching of the data.

All the information in MeSH is described in descriptor records as depicted in figure 4.1.



MeSH Heading	Dyslexia, Acquired
Tree Number	C10.597.606.150.500.300.200
Tree Number	C10.597.606.150.550.200.500
Tree Number	C23.888.592.604.150.500.300.200
Tree Number	C23.888.592.604.150.550.200.500
Tree Number	F03.087.700
Tree Number	F03.550.350.500.200.500
Tree Number	F03.550.450.400.500
Scope Note	A receptive visual aphasia characterize condition may be associated with poste
Entry Term	Acquired Global Dyslexia
Entry Term	Acquired Spelling Dyslexia
Entry Term	Alexia, Acquired
Entry Term	Reading Disability, Acquired
Entry Term	Word Blindness, Acquired
Allowable Qualifiers	BL CF CI CL CO DH DI DT EC EH
History Note	1977(1963)
Date of Entry	19990101
Unique ID	D004411

Figure 4.1: MeSH Descriptor Record of Dyslexia, Acquired

It is evident from this figure 4.1, that each MeSH entity has Unique ID, MeSH Heading, and Tree Number. As is evident from the name, Unique ID is the unique identifier of a descriptor, MeSH Heading is the preferred unique term for the descriptor that is used to represent it in other elements and Tree Number represents the level or position of an entity within the MeSH taxonomy. Besides that, Entry Terms are also provided in descriptor records, if they exist. Entry Terms in MeSH are names of substances that are considered equivalent for retrieval purposes. Moreover, MeSH follows the phenomena of polyhierarchy, so according to this, a descriptor can have more than one Tree Numbers [39], which means that a descriptor can belong to more than one category at a time.

4.2 Knowledge Base Construction

4.2.1 Knowledge Base Structure

The structure for our knowledge base is depicted in detail in figure 4.2, which also shows that how the information from MeSH descriptor record is mapped to the different columns of three of our KB tables named *Individuals*, *Synonymous Terms* and *Facts*.

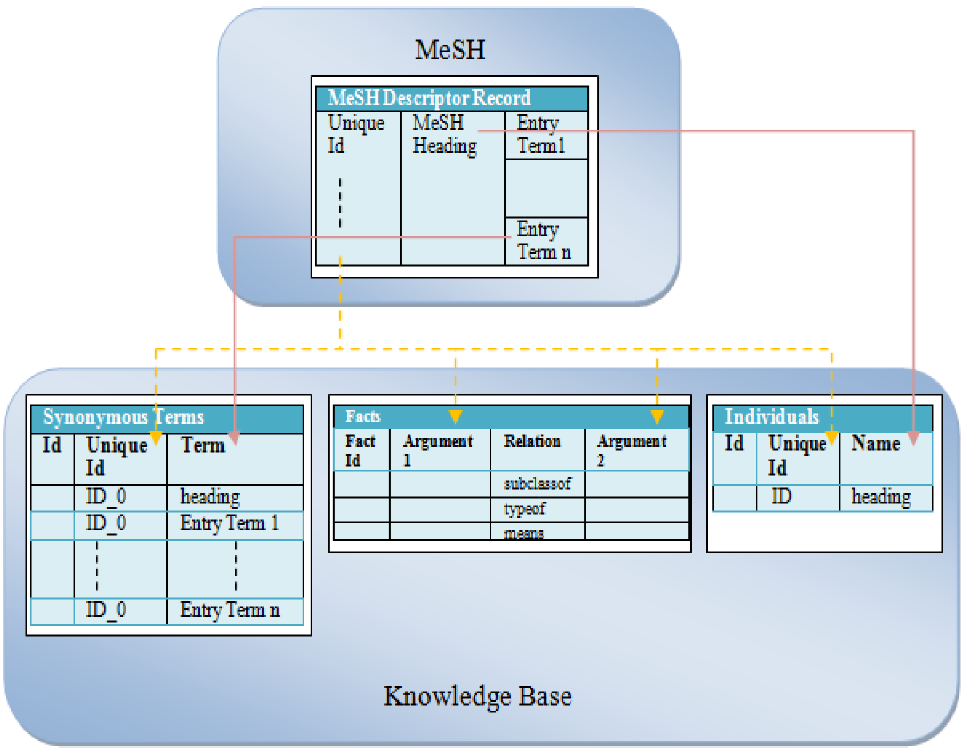


Figure 4.2: Knowledge Base Structure and Steps for Integration of Data from MeSH

4.2.2 Steps for Knowledge Extraction from MeSH

All of the MeSH entities are categorized in 16 classes, as depicted in figure 4.3. Based on the sub domains required for our KB, we grouped the required MeSH entities in Diseases, Symptoms, Body Parts, Environmental Factors, Social Factors, Nutritional Factors and Diagnostic Factors. We manually selected the constituent MeSH classes for each of our KB classes and noted down their tree numbers for accessing the required data of each of those MeSH classes from the sql database in which we crawled the MeSH data (as described in chapter 3). A data selection criterion for each of our KB categories based on Tree Numbers is described in detail in Table 4.2.

The basic steps which we followed for storing MeSH entities data into Individuals and Synonymous Terms tables of our KB are described as under:

1) The Unique ID and MeSH Heading of each required entity is recorded in Individuals table as a concept.

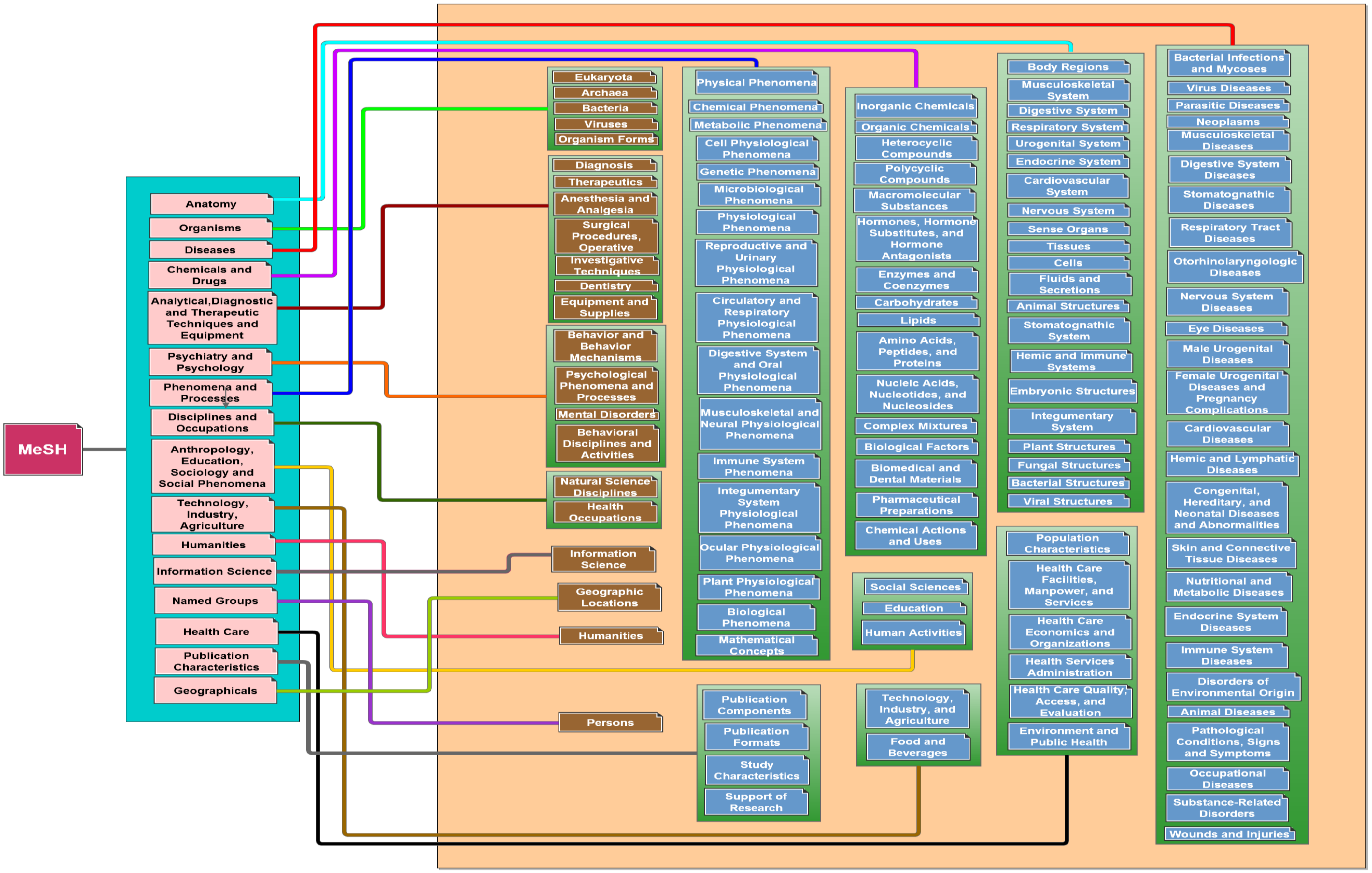
Table 4.2: KB Categories and Their Constituents from MeSH Identified With Tree Numbers

Category	Constituents
Body Parts	All sub classes of Anatomy[A01…..A21]
Symptoms	Signs and symptoms[C23.888]
Diseases	1.Mental Disorders [F03] 2.All subclasses of Diseases Except Signs and symptoms [C01…C22, C23.300, C23.550, C24, C25, C26]
Environmental Factors	1.Sub classes of Chemicals and Drugs [D01…D05, D20, D23, D25, D26, D27] 2.Following subclasses of phenomena and processes: a. Astronomical Phenomena [G01.060] b. Geological Phenomena [G01.311] c. Radiation [G01.750] 3.All sub classes of organisms [B01…B05] 4.Environment and Public Health [N06] 5.Technology, industry and agriculture [J01]
Social Factors	1. Subclasses of Health Care [N01….N05] 2. Subclasses of Psychiatry and Psychology [F01, F02, F04] 3. Sub classes of Anthropology, Education, Sociology and Social Phenomena [I01… I03]
Nutritional Factors	1. Subclasses of Chemicals and Drugs [D06, D08, D09, D10,D12, D13] 2. Following subclasses of phenomena and processes: a. Nutritional Physiological Phenomena [G07.610] b. Nutrition Processes [G07.700.620] 3. Food and Beverages [J02]
Diagnostic Factors	Sub classes of Analytical, Diagnostic and Therapeutic Techniques and Equipment [E01….E07]

2) For effectively recording all the synonyms in our KB, we stored Unique ID, MeSH Heading and all the Entry Terms of a descriptor record into the SynonymousTerms table as instances of a concept. For Example, entries made in Synonymous Terms table for Unique ID D004411 are depicted in Table 4.3:

Table 4.3: Synonyms for Unique Id D004411

Unique Id	Term
D004411_0	Acquired Global Dyslexia
D004411_0	Acquired Spelling Dyslexia
D004411_0	Alexia, Acquired
D004411_0	Dyslexia, Acquired
D004411_0	Reading Disability, Acquired
D004411_0	Word Blindness, Acquired


MeSH
Anatomy
Organisms
Diseases
Chemicals and Drugs
Analytical,Diagnostic and Therapeutic Techniques and Equipment
Psychiatry and Psychology
Phenomena and Processes
Disciplines and Occupations
Anthropology, Education, Sociology and Social Phenomena
Technology, Industry, Agriculture
Humanities
Information Science
Named Groups
Health Care
Publication Characteristics
Geographicals
Eukaryota
Archaea
Bacteria
Viruses
Organism Forms
Diagnosis
Therapeutics
Anesthesia and Analgesia
Surgical Procedures, Operative
Investigative Techniques
Dentistry
Equipment and Supplies
Behavior and Behavior Mechanisms
Psychological Phenomena and Processes
Mental Disorders
Behavioral Disciplines and Activities
Natural Science Disciplines
Health Occupations
Information Science
Geographic Locations
Humanities
Persons
Physical Phenomena
Chemical Phenomena
Metabolic Phenomena
Cell Physiological Phenomena
Genetic Phenomena
Microbiological Phenomena
Physiological Phenomena
Reproductive and Urinary Physiological Phenomena
Circulatory and Respiratory Physiological Phenomena
Digestive System and Oral Physiological Phenomena
Musculoskeletal and Neural Physiological Phenomena
Immune System Phenomena
Integumentary System Physiological Phenomena
Ocular Physiological Phenomena
Plant Physiological Phenomena
Biological Phenomena
Mathematical Concepts
Publication Components
Publication Formats
Study Characteristics
Support of Research
Inorganic Chemicals
Organic Chemicals
Heterocyclic Compounds
Polycyclic Compounds
Macromolecular Substances
Hormones, Hormone Substitutes, and Hormone Antagonists
Enzymes and Coenzymes
Carbohydrates
Lipids
Amino Acids, Peptides, and Proteins
Nucleic Acids, Nucleotides, and Nucleosides
Complex Mixtures
Biological Factors
Biomedical and Dental Materials
Pharmaceutical Preparations
Chemical Actions and Uses
Social Sciences
Education
Human Activities
Technology, Industry, and Agriculture
Food and Beverages
Body Regions
Musculoskeletal System
Digestive System
Respiratory System
Urogenital System
Endocrine System
Cardiovascular System
Nervous System
Sense Organs
Tissues
Cells
Fluids and Secretions
Animal Structures
Stomatognathic System
Hemic and Immune Systems
Embryonic Structures
Integumentary System
Plant Structures
Fungal Structures
Bacterial Structures
Viral Structures
Population Characteristics
Health Care Facilities, Manpower, and Services
Health Care Economics and Organizations
Health Services Administration
Health Care Quality, Access, and Evaluation
Environment and Public Health
Bacterial Infections and Mycoses
Virus Diseases
Parasitic Diseases
Neoplasms
Musculoskeletal Diseases
Digestive System Diseases
Stomatognathic Diseases
Respiratory Tract Diseases
Otorhinolaryngologic Diseases
Nervous System Diseases
Eye Diseases
Male Urogenital Diseases
Female Urogenital Diseases and Pregnancy Complications
Cardiovascular Diseases
Hemic and Lymphatic Diseases
Congenital, Hereditary, and Neonatal Diseases and Abnormalities
Skin and Connective Tissue Diseases
Nutritional and Metabolic Diseases
Endocrine System Diseases
Immune System Diseases
Disorders of Environmental Origin
Animal Diseases
Pathological Conditions, Signs and Symptoms
Occupational Diseases
Substance-Related Disorders
Wounds and Injuries
Figure 4.3: MeSH Categories and their constituents

3) For differentiating between concepts and instances in our KB, a "_0" suffix was concatenated with each instance's Unique ID. For example, if:

Concept:	D004411	Dyslexia, Acquired

Then:

Instance:	D004411_0	Dyslexia, Acquired

4.2.3 Relations' Extraction

For populating the Facts table, we followed the MeSH taxonomy given by Tree Numbers and took following steps regarding each kind of relation:

4.2.3.1 The Subclassof Relation

The subclassof relation is meant to associate each entity with its associated parent entities. As explained earlier in previous chapter, this is accomplished through "*Tree Number*" attribute of an entity term, which helps us in identifying all of the parents. For example, the parent of an entity "*Foot*" having Tree number "A01.378.610.250" is an entity named "Lower Extremity" whose tree number is "A01.378.610", which can be chunked out from Tree number of "Foot" in the following way, as shown in figure 4.4:

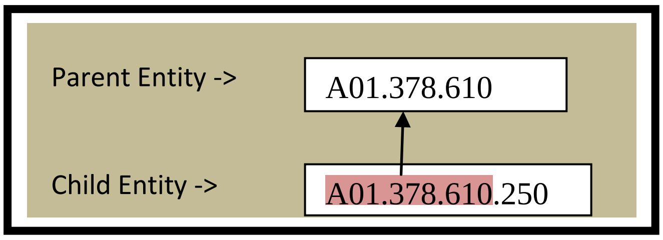


Figure 4.4: Identifying the Parent of an Entity

We kept the above scenario in mind and wrote out the SQL script (as shown in Appendix A.2) for collecting the subclassof relations for our KB from the MeSH crawled data which was stored in a separate SQL table (explained in chapter 3).

With the help of this script, the Unique ID of each entity class is connected to the directly associated upper classes by subclassof relation. For Example: entries made in Facts table for Unique ID D004411 are depicted in Table 4.4, which intend to state that Dyslexia, Acquired(D004411) is subclass of Dyslexia (D004410) and Delirium, Dementia, Amnestic, Cognitive Disorders (D019965).

Table 4.4: Facts for Unique Id D004411With Subclassof Relation

Fact Id	Argument 1	Relation	Argument 2
29312	D004411	subclassof	D004410
29313	D004411	subclassof	D019965

4.2.3.2 The Typeof Relation

The typeof relation is meant to associate each entity with all of its associated root entities. By root entities, we mean all of the predecessor entities of that entity term. This is also accomplished through "*Tree Number*" attribute of an entity term, which helps us in identifying all of the associated root entities also called instances. For example, the instances of an entity "*Foot*" having Tree number "A01.378.610.250" are entities: "Lower Extremity [A01.378.610]", "Extremities [A01.378]", "Body Regions [A01]", which can be chunked out from Tree number of "Foot" in the following way, as shown in figure 4.5:

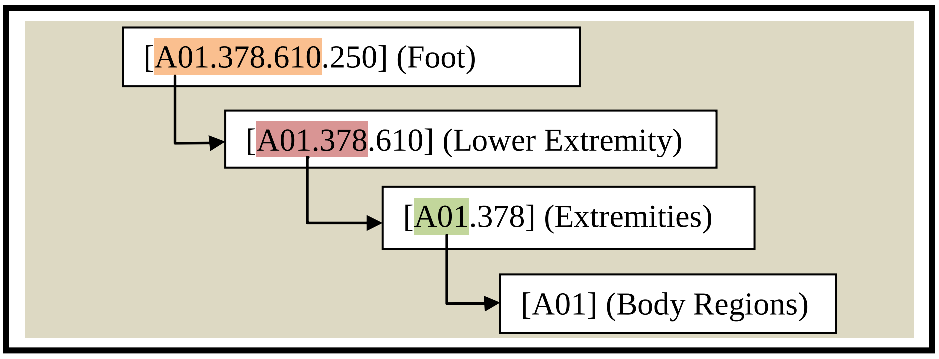


Figure 4.5: Identifying the instances of an Entity

We kept the above scenario in mind and wrote out the SQL script (as shown in Appendix A.2) for collecting the typeof relations for our KB from the MeSH crawled data which was stored in a separate SQL table (explained in chapter 3).

With the help of this script, the Unique Id of each entity class is connected to its instances by typeof relation. By instance, we mean the upper classes of an entity class as well as all of the predecessors of those upper classes. For example: entries made in Facts table for Unique ID D004411 are depicted in Table 4.5. All of these entries intend to state that Dyslexia, Acquired (D004411) is a typeof Mental Disorders

(D001523), Communication Disorders (D003147), Dyslexia (D004410), Language Disorders (D007806), Learning Disorders (D007859), Nervous System Diseases (D009422), Neurologic Manifestations (D009461), Signs and Symptoms (D012816), Mental Disorders Diagnosed in Childhood (D019952), Neurobehavioral Manifestations (D019954) and Delirium, Dementia, Amnestic, Cognitive Disorders (D019965).

Table 4.5: Facts for Unique Id D004411With Typeof Relation

Fact Id	Argument 1	Relation	Argument 2
29314	D004411	typeof	D001523
29315	D004411	typeof	D003147
29316	D004411	typeof	D004410
29317	D004411	typeof	D007806
29318	D004411	typeof	D007859
29319	D004411	typeof	D009422
29320	D004411	typeof	D009461
29321	D004411	typeof	D012816
29323	D004411	typeof	D019952
29324	D004411	typeof	D019954
29325	D004411	typeof	D019965

4.2.2.3 The Means Relation

This relation is intended to associate all of the synonymous terms. This was accomplished by connecting the Unique ID of each entity class to its corresponding MeSH Heading and Entry Terms by means relation. For this purpose, we utilized the Individuals and Synonymous Terms table and write out an SQL script (presented in Appendix A.2) for gathering these relations. For example, entry made in Facts table for Unique ID D004411 is: D004411 means D004411_0. In this entry, D004411 corresponds to Dyslexia, Acquired and D004411_0 corresponds to the instances of D004411from Individuals and Synonymous Terms table. So, according to this logic, D004411 means (i) Dyslexia, Acquired, (ii) Acquired Global Dyslexia, (iii) Acquired Spelling Dyslexia, (iv) Alexia, Acquired, (v) Reading Disability, Acquired, and (vi) Word Blindness, Acquired.

4.2.3 Data Cleaning

After gathering all of this data in our KB, we moved to the data cleaning step and removed all of the duplicates from all three of our tables.

4.3 Summary

After getting the data from our information source MeSH into an SQL table, the KB structure was built up and then all the biomedical entities were organized in “individuals” table. After that, all of the synonymous terms were effectively recorded in “synonymous terms” table and then finally, the “subclassof”, “typeof” and “means” relations regarding the gathered biomedical entities of our KB, were collected through SQL scripting.

Chapter 5

RESULTS AND DISCUSSION

Now when we have already gone through the data extraction and relation gathering steps, this chapter will focus on evaluating the effectiveness of the resulting ontology with reference to its comparison to the existing KBs.

5.1 Overview

After getting done with all of the knowledge extraction process, the resulting KB contains the required data. The total number of individuals, synonymous terms and facts recorded in our KB are presented in Table 5.1.

Table 5.1: Total Number Of Individuals, Synonymous Terms And Facts In Our KB

Individuals	53020
Synonymous Terms	96835
Facts	197731

Considering the availability of a few reliable sources other than MeSH; like UMLS and OMIM, we could have gone a bit further and would have extended the coverage of our ontology, but effective knowledge modeling of our ontology was preferred over increasing the sample-space and incorporating the data from these other sources into our well modeled and well based knowledge base is now left as a future work.

5.2 Comparison to Major Exiting Biomedical KBs

A hefty list of existing biomedical ontologies was presented in chapter 2. All of them focus on different biomedical sub domains. None of them provided an integrated semantic knowledge regarding different biomedical sub-domains at one place. E.g. knowledge regarding most of the main factors (including, environmental, social, nutritional and diagnostic factors) of a disease has not yet been collected at one place without focusing on a specialized context. If anyone has to access all of these varying factors, then he must have to rely on more than one KB, as no core KB having such integrated knowledge about contributing factors of diseases have yet been built.

And one of the major issues that can arise while utilizing more than one KBs together, is the mapping of diversified entities covered in KBs. This issue was the main driving force and motivation that lead us to this research work. So, in contrast to all of the existing KBs including:

1) ones that are focused on a specialized domain (e.g. PPI, GO, KEGG)

2) that have a taxonomic detailing but lack useful semantic relation (e.g. MeSH, UMLS)

3) that contain details regarding relations of specialized contexts, like relations between genotype and drug response phenotype, or disease gene associations (e.g. OMIM, GAD, Diseasome).

Our ontology:

1) does not focus on any specialized domain and caters to all the diseases, their symptoms and causative factors (including environmental, social, nutritional and diagnostic factors) on the whole.

2) besides just classifying the biomedical data in predefined categories, provides knowledge regarding three kind of semantic relations, including, *subclassof*, *typeof* and *means* relation

3) instead of just focusing on relations or causative factors of some specialized context like, genotype and drug response phenotype, or disease gene associations; contains an integrated knowledge regarding the general and most common causative factors of diseases i.e. environmental, social, nutritional and diagnostic factors

Most of these above mentioned KBs are hand crafted and are manually curated by human experts. A lot of information extraction techniques like pattern matching, natural language processing and statistical learning [30, 31, 32, 33, 34, 35, 36] have been introduced recently for creating high quality ontology, but they have still not been able to surpass the quality of manually curated ones. However the manually integrated KBs have to undergo the challenges of low coverage, high integration cost, quality assurance and fast aging [11]. Keeping this in view, the selection of a quality ontology that has reliable information is quite a challenging task. United States National Library of Medicine (NLM) is one of the 27 institutions of the U.S National Institute of Health (NIH) and is considered one of the best in the research domain of biomedicine and bioinformatics for developing and controlling several most frequently used and trusted biomedical KBs. MeSH is also controlled by NLM and is frequently updated and revised, so this leverages for our concerns regarding the reliability, quality assurance and fast aging issues regarding MeSH being our knowledge source. Therefore, in contrast to the above mentioned KBs, our ontology has not been manually curated and is based on data from a trusted medical thesaurus MeSH.

A summarization of main differences between our KB and other existing biomedical KBs, is presented in table 5.2.

5.3 Accessing the Knowledge Base

For effective visualization of the data gathered in our KB, we provide a web browsing interface. The interface makes sure that the knowledge stored in our KB is clearly represented. In this chapter it is explained in detail with the help of different example scenarios.

5.3.1 Tools – Microsoft Sql Server 2008, Microsoft Visual Studio 2010

As is detailed in chapter 4, our KB was built up in Microsoft SQL server 2008, so this database served as the backend of our browsing tool and the actual browsing tool was programmed through Microsoft Visual Studio 2010, in C#.

5.3.2 Knowledge Base Exploration

Through this browsing tool, users can comfortably access the information regarding constituents of different categories of our KB. User can simply enter the name of that entity which he wants to search for and as a result he obtains fact's details regarding all three kind of relation categories i.e. subclassof, typeof and means relation. Besides that, it is also mentioned that to which KB category does this entered search term belongs. For instance, if the user enters the search term "Dyslexia, Acquired", then system will render results as depicted in figure 5.1.

User can even click one of the entities from the results and then that entity becomes the search term and all its related information is displayed.

For a comprehensive description of the kind of knowledge currently accessible through our KB, some examples of the possible queries on our KB are described in natural language as follows:

- *Dyslexia, Acquired*IsA ?

  The answer to this query will include all entities which are in the "subClassOf" relationship with "Dyslexia, Acquired".

- *Dyslexia, Acquired*HasSynonyms ?

OR

Dyslexia, Acquired Means ?

OR

*Dyslexia, Acquired*IsAlsoCalled ?

OR

*Dyslexia, Acquired*IsTheSameAs ?

The answer to these queries will include all entities which are in the “means” relationship with “Dyslexia, Acquired”.

- *Dyslexia, Acquired*IsRelatedTo ?

  OR

  Dyslexia, Acquired BelongsTo ?

  The answer to these queries will include all entities which are in the “TypeOf” relationship with “Dyslexia, Acquired”.

Sample results for all of the categories of our ontology are presented in Appendix A.3.

5.4 Application Areas of our Knowledge Base

5.4.1 Overview

In past few years, use of ontological background knowledge has increased manifolds in lot of domains. By judging its effectiveness in other domains, ontologies have become a hot topic in biomedical domain as well. But, until now, most of the ontologies were focused on a few sub domains of biomedicine. Considering this, our KB is intended to provide information regarding varying biomedical sub domains.

As we all know that information on web is scattered and mixed with pile of noisy unstructured texts and media [1]. This serves as a hindrance in effectively utilizing this vast amount of medical literature. For example, if user initiates a simple web search regarding a disease and wants to know its symptoms and contributory factors, then all he

Table5.2: comparison of our KB with other major existing biomedical KBs

KB	Domain	Source	Size	Semantic Relations	Causative Factors of Diseases
Our Core Ontology	Diseases, Symptoms, Body Parts, Nutritional Factors, Environmental Factors, Social Factors, Diagnostic Factors	MeSH	53020 Individuals, 96835 Synonymous terms, 197731 Facts	Included	Included, i.e. Nutritional Factors, Environmental Factors, Social Factors, Diagnostic Factors
PPI db of MIPS	Protein to protein interaction	Scientific literature	N.A	Not included	Not included
GO GenBank	Genes, gene product genes	Scientific publications, direct submissions from individual laboratories, bulk submissions from large-scale sequencing centers	N.A	Contains relations of specialized context	Not included
KEGG	Genomes, Enzymatic pathways, chemicals	N.A	>11 mil genes, ~134,000 pathway maps, 375 human diseases, 9.336 drugs	Contains relations of specialized context	Not included
OMIM	genes, genetic disorders, including phenotype description and body parts	Manually generated by scientists and physicians	~18,597 genes	Contains relations of specialized context, e.g. relations between genotype and drug response phenotype	Contains causative factors of specialized context, e.g. only genetic disorders that can lead to a disease
MeSH	General medical subjects for indexing articles for PubMed database	PubMed publications	~25,186 entities	Not included	Included as a separate entry term
Diseasome	Human disease network	OMIM	>4,213 diseases, >91,182 genes	Contains relations of specialized context	Not included

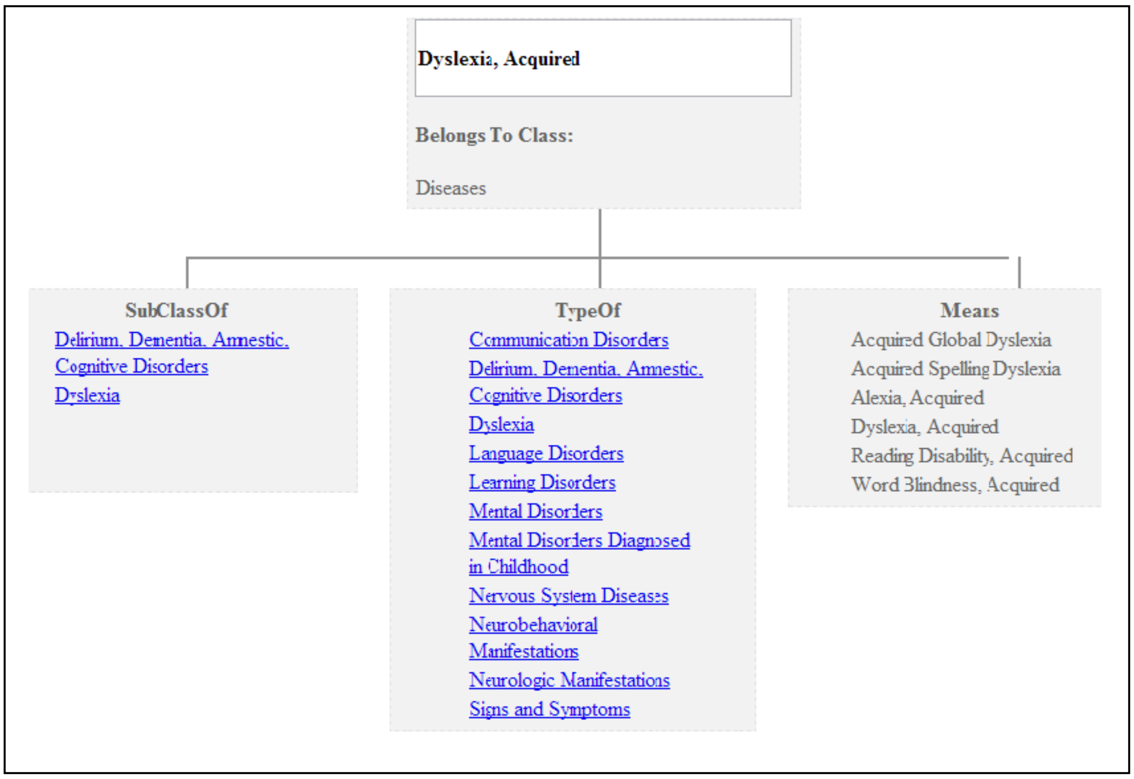


Figure 5.1: Browsing Interface for Our KB

is provided is with a huge pile of articles and papers. Even over viewing those links is bound to take him lot of months, let alone to extract the required answers. So, if all of this information regarding body parts, diseases, their symptoms and contributory factors is all compiled in one machine readable KB, then it can have enormous benefits.

5.4.2 Major Application Areas

Some of those prospective applications and benefits of our KB are described as under:

1) A comprehensive machine readable encyclopedia that can be queried in a highly precised and expressive way just like a semantic database.
2) Helping in document classification by integrating supervised or semi supervised learning techniques with our background KB.
3) Providing a way to carry out entity relationship oriented semantic web search, by helping in two ways: (i) detection of entities and relations from web pages, (ii) reasoning about detected entities and relations in probabilistic logics.
4) Enabling effective entity disambiguation or word sense disambiguation (e.g. [40]) by accurately and quickly mapping the text phrases onto named entities in our KB.

5) A medium for effectively gathering more related information from biomedical literature and growing our KB.
6) Utilizing our ontological knowledge structure for the purpose of data cleaning (e.g for data warehouses) [41] and record linkage (also known as entity resolution) [42].
7) A comprehensive source for providing aid in machine translation (e.g. [43]) related tasks.
8) Serving as a backbone for natural language question answering browsing (e.g. [44] [45]).

Chapter 6

CONCLUSION AND FUTURE WORK

6.1 Overview

Knowledge is of no use if it is not represented and shared in a quality way. Sharing and passing knowledge helps in achieving the advancements without reinventing the wheel and thus results in advancement of humanity as a whole. In past few years, amount of research work has grown to great lengths in some fields. Especially, a lot of researches have been carried out in biomedical domain and spreading of this knowledge to masses, made an important contributory step towards civilization. But most of this knowledge is present in scientific publications and only very few people know how to access it. If this knowledge can be made accessible to the common people then a lot of advantages can be achieved including, diseases' prevention, early and more accurate disease diagnosis, more effective treatment and many more".

Based on the huge pile of health information available on the internet, web has the potential of being the ultimate encyclopedic source. But effective retrieval of required results from web has always been problematic, due to which users are still far from exploiting this potential. Users have to undergo vast amount of difficulties in finding the exact precise information from this huge pile of health data.

Especially, most of the information in biomedical research area is not yet captured in databases today, but rather present in structured form in scientific publications and also in semi-structured or unstructured form on web. If this knowledge gets effectively provided to the population then many diseases can be prevented, diagnosed earlier, and more accurately, and thus treated better, and cured more effectively; even epidemics and pandemics could be avoided.

Researchers have long been trying to find the solution to this problem and according to them [1] [4] [5] [6] [7]; the best possible way to solve this issue is to move to the concept of semantic web. But due to the diversity of terms and their definitions between groups; adding semantics to web is not a straight forward task. It asks for achieving a shared and common understanding of a domain. So, this leads us to the creation of an ontology, which can be applied to various contexts for variety of purposes. Ontologies serve as the backbone of semantic web by facilitating knowledge exchange across people and application systems.

Thus, we dedicated ourselves to the accessibility of the medical expertise for the populace by building a biomedical knowledge base on entities and relations regarding diseases, their symptoms, body parts and determining factors, with emphasis on environmental, social, nutritional and diagnostic factors.

6.2 Conclusion

In this research work, we have described the whole process of creation of a biomedical knowledge base containing entities and relationships (subclassof, typeof, means) regarding body parts, diseases, their symptoms and causative factors. The varying causative factors which are covered through this KB include environmental factors, social factors, nutritional factors and diagnostic factors.

The problems of effective retrieval of required results from web and lack of availability of biomedical research information in databases have long been prevailing in the research community. Researchers have long been trying to find the solution to this problem and proposed the concept of semantic web as a solution for them. But due to the diversity of terms and their definitions between groups; adding semantics to web was not a straight forward task. It asks for achieving a shared and common understanding of a domain. So, these lead us to the creation of an ontology, which can be applied to various contexts for variety of purposes.

In the biomedical domain, there already were plenty of biomedical knowledge collections available. After going through an overview of the state-of-the-art of existing knowledge collections of general as well as biomedical domain, we noticed that none of them was capable of providing certain level of details regarding different causative factors of diseases at one place through one core KB, without focusing on a specialized context. So, this gave birth to this idea of creating a core KB, that can provide integrated knowledge regarding, diseases, symptoms, body parts and causative factors (environmental, social, nutritional, diagnostics factors) of diseases.

Good data leads to good results and bad data is always misleading. Especially for building an ontology, the quality and reliability of data matters a lot, as this is what eventually results in effectiveness, reliability and coverage of an ontology. The data we begin with takes the form of an input and this input serves as the backbone of ontology. Keeping this in mind, we selected the most trusted biomedical thesaurus MeSH as our Information source and collected all of the required biomedical entities' details from its web browser into an SQL database, through our customized crawler.

After crawling the data from our information source, MeSH theasaurus, into an SQL database, the KB structure was built up and then all the biomedical entities from that crawled data were organized in "individuals" table. After that, all of the related entry terms for these organized entities of our KB, were effectively recorded in "synonymous terms" table and then finally, the "subclassof", "typeof" and "means" relations regarding the gathered biomedical entities of our KB, were collected through SQL scripting. Facts or relations regarding each MeSH entity were quickly and accurately collected by effectively querying the MeSH Tree Numbers of entities.

For providing a detailed understanding of ontologies to the readers, we presented a comprehensive introduction to the existing knowledge bases and knowledge representation techniques in both general and biomedical domain. Besides tha, the data model and source for our KB were also clearly mentioned and explained in detail to avoid any confusions.

Nonetheless, our KB can still be further improved and extended increasing its effectiveness and coverage.

6.3 Future Work

One of the major improvement areas is source of the KB. Currently, our KB is utilizing only MeSH thesaurus as the core source, so it would be quite interesting and useful to do the refined integration of data from some other trusted knowledge sources as well. This might result in improvement regarding coverage of our KB, in a way that more diseases, symptoms, body parts and factors get collected as well. Besides that, some improvement in knowledge model of our KB might also result in a more faithful knowledge representation from respective taxonomies.

In addition to that, an important contribution would be to record the relations between those entities which belong to different classes, e.g. to record relations between entities of class "Diseases" and class "Diagnostic Factors". This kind of relations can be effectively collected from scientific research material by utilizing some text mining techniques.

Last but not least, additional visualization capabilites can be implemented to provide better access to the knowledge base. Among others, this includes a redesigning of our Web interface towards a clearer and more sophisticated layout.

APPENDICES

Appendix A.1: Code Snippet for our Customized Crawler

<u>Crawler.cs</u>

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Net;
using System.IO;
using System.Collections.Specialized;
using ExtractThesis.Functions;


namespace Utility.Parser
{
    public class CCrawler
    {
        private CookieContainer _cookieContainer;

        public CCrawler()
        {
            _cookieContainer = new CookieContainer();
        }

        public byte[] ProcessPostResponseGetBinary(Dictionary<string, string> post, string _url)
        {
            if (_url == String.Empty)
                throw new Exception("ProcessPostResponse: URL not provided");

            ASCIIEncoding encoding = new ASCIIEncoding();
            StringBuilder postData = new StringBuilder();
            bool first = true;

            foreach (KeyValuePair<string, string> p in post)
            {
                if (first)
                {
                    postData.Append(p.Key + "=" + p.Value);
                    first = false;
                }
                else
                    postData.Append("&" + p.Key + "=" + p.Value);
            }

            byte[] data = encoding.GetBytes(postData.ToString());

            // Prepare web request...
            HttpWebRequest myRequest = (HttpWebRequest)WebRequest.Create(_url);
            myRequest.Method = "POST";

            myRequest.CookieContainer = _cookieContainer;

            myRequest.ContentType = "application/x-www-form-urlencoded";
            myRequest.UserAgent = "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; .NET CLR 1.0.3705;)";
            myRequest.Accept = "text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8";
            myRequest.Headers.Add("Accept-Language: en-us,en;q=0.5");
            myRequest.Headers.Add("Accept-Encoding: gzip,deflate");
            myRequest.Headers.Add("Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7");
            myRequest.KeepAlive = true;
            myRequest.Headers.Add("Keep-Alive: 300");

            myRequest.ContentLength = data.Length;
            Stream newStream = myRequest.GetRequestStream();

            // Send the data.
            newStream.Write(data, 0, data.Length);
            newStream.Close();

            myRequest.CookieContainer = _cookieContainer;

            HttpWebResponse myResponse = (HttpWebResponse)myRequest.GetResponse();
            _cookieContainer = myRequest.CookieContainer;

            Stream remoteStream = myResponse.GetResponseStream();

            MemoryStream localStream = new MemoryStream();
```

```
        byte[] buffer = new byte[1024];
        int bytesRead;
        int bytesProcessed = 0;
        do
        {
            bytesRead = remoteStream.Read(buffer, 0, buffer.Length);
            localStream.Write(buffer, 0, bytesRead);
            bytesProcessed += bytesRead;
        } while (bytesRead > 0);

        byte[] output = new byte[localStream.Length];

        localStream.Position = 0;

        localStream.Read(output, 0, Convert.ToInt32(localStream.Length));
        return output;
    }

    public string ProcessPostResponse(Dictionary<string, string> post, string _url)
    {
        if (_url == String.Empty)
            throw new Exception("ProcessPostResponse: URL not provided");

        ASCIIEncoding encoding = new ASCIIEncoding();
        StringBuilder postData = new StringBuilder();
        bool first = true;

        foreach (KeyValuePair<string, string> p in post)
        {
            if (first)
            {
                postData.Append(p.Key + "=" + p.Value);
                first = false;
            }
            else
                postData.Append("&" + p.Key + "=" + p.Value);
        }

        byte[] data = encoding.GetBytes(postData.ToString());

        // Prepare web request...
        HttpWebRequest myRequest = (HttpWebRequest)WebRequest.Create(_url);
        myRequest.Method = "POST";

        myRequest.CookieContainer = _cookieContainer;

        myRequest.ContentType = "application/x-www-form-urlencoded";
        myRequest.UserAgent = "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; .NET CLR 1.0.3705;)";
        myRequest.Accept = "text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8";
        myRequest.Headers.Add("Accept-Language: en-us,en;q=0.5");
        myRequest.Headers.Add("Accept-Encoding: gzip,deflate");
        myRequest.Headers.Add("Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7");
        myRequest.KeepAlive = true;
        myRequest.Headers.Add("Keep-Alive: 300");

        myRequest.ContentLength = data.Length;
        Stream newStream = myRequest.GetRequestStream();
        // Send the data.
        newStream.Write(data, 0, data.Length);
        newStream.Close();

        myRequest.CookieContainer = _cookieContainer;

        HttpWebResponse myResponse = (HttpWebResponse)myRequest.GetResponse();
        _cookieContainer = myRequest.CookieContainer;
        StreamReader reader = new StreamReader(myResponse.GetResponseStream());
        return reader.ReadToEnd().Trim();
    }

    public string ProcessPostResponseNoHeaders(Dictionary<string, string> post, string _url)
    {
        if (_url == String.Empty)
            throw new Exception("ProcessPostResponse: URL not provided");

        ASCIIEncoding encoding = new ASCIIEncoding();
        StringBuilder postData = new StringBuilder();
        bool first = true;

        foreach (KeyValuePair<string, string> p in post)
        {
            if (first)
            {
                postData.Append(p.Key + "=" + p.Value);
                first = false;
            }
            else
                postData.Append("&" + p.Key + "=" + p.Value);
        }

        byte[] data = encoding.GetBytes(postData.ToString());

        // Prepare web request...
        HttpWebRequest myRequest = (HttpWebRequest)WebRequest.Create(_url);
        myRequest.Method = "POST";

        myRequest.CookieContainer = _cookieContainer;

        myRequest.ContentType = "application/x-www-form-urlencoded";
        myRequest.UserAgent = "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; .NET CLR 1.0.3705;)";
        myRequest.KeepAlive = true;
        myRequest.Headers.Add("Keep-Alive: 300");

        myRequest.ContentLength = data.Length;
        Stream newStream = myRequest.GetRequestStream();

        // Send the data.
        newStream.Write(data, 0, data.Length);
        newStream.Close();

        myRequest.CookieContainer = _cookieContainer;

        HttpWebResponse myResponse = (HttpWebResponse)myRequest.GetResponse();
        _cookieContainer = myRequest.CookieContainer;
        StreamReader reader = new StreamReader(myResponse.GetResponseStream());
        return reader.ReadToEnd().Trim();
    }

    public string ProcessPostResponseNoHeaders(List<NVP> post, string _url)
    {
        if (_url == String.Empty)
            throw new Exception("ProcessPostResponse: URL not provided");

        ASCIIEncoding encoding = new ASCIIEncoding();
```

```
StringBuilder postData = new StringBuilder();
bool first = true;

foreach (NVP p in post)
{
    if (first)
    {
        postData.Append(p.Name + "=" + p.Value);
        first = false;
    }
    else
        postData.Append("&" + p.Name + "=" + p.Value);
}


byte[] data = encoding.GetBytes(postData.ToString());

// Prepare web request...
HttpWebRequest myRequest = (HttpWebRequest)WebRequest.Create(_url);
myRequest.Method = "POST";

myRequest.CookieContainer = _cookieContainer;

myRequest.ContentType = "application/x-www-form-urlencoded";
myRequest.UserAgent = "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; .NET CLR 1.0.3705;)";
myRequest.KeepAlive = true;
myRequest.Headers.Add("Keep-Alive: 300");

myRequest.ContentLength = data.Length;
Stream newStream = myRequest.GetRequestStream();

// Send the data.
newStream.Write(data, 0, data.Length);
newStream.Close();

myRequest.CookieContainer = _cookieContainer;

HttpWebResponse myResponse = (HttpWebResponse)myRequest.GetResponse();
_cookieContainer = myRequest.CookieContainer;
StreamReader reader = new StreamReader(myResponse.GetResponseStream());
return reader.ReadToEnd().Trim();
```

```
}

public string ProcessGetResponse(Dictionary<string, string> post, string _url)
{
    if (_url == String.Empty)
        throw new Exception("ProcessGetResponse: URL not provided");

    ASCIIEncoding encoding = new ASCIIEncoding();
    StringBuilder queryString = new StringBuilder();
    bool first = true;

    foreach (KeyValuePair<string, string> p in post)
    {
        if (first)
        {
            queryString.Append("?" + p.Key + "=" + p.Value);
            first = false;
        }
        else
            queryString.Append("&" + p.Key + "=" + p.Value);
    }

    // Prepare web request...
    HttpWebRequest myRequest = (HttpWebRequest)WebRequest.Create(_url + queryString.ToString());
    myRequest.Method = "GET";
    myRequest.CookieContainer = _cookieContainer;

    myRequest.UserAgent = "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; .NET CLR 1.0.3705;)";
    myRequest.Accept = "text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8";
    myRequest.Headers.Add("Accept-Language: en-us,en;q=0.5");
    myRequest.Headers.Add("Accept-Encoding: gzip,deflate");
    myRequest.Headers.Add("Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7");
    myRequest.KeepAlive = true;
    myRequest.Headers.Add("Keep-Alive: 300");
    myRequest.ContentType = "application/x-www-form-urlencoded";

    HttpWebResponse myResponse = (HttpWebResponse)myRequest.GetResponse();
    _cookieContainer = myRequest.CookieContainer;

    StreamReader reader = new StreamReader(myResponse.GetResponseStream());
    return reader.ReadToEnd().Trim();
```

```
}
public string ProcessResponse(string _url)
{
    if (_url == String.Empty)
        throw new Exception("ProcessResponse: URL not provided");

    // Prepare web request...
    HttpWebRequest myRequest = (HttpWebRequest)WebRequest.Create(_url);
    myRequest.Method = "GET";
    myRequest.CookieContainer = _cookieContainer;

    myRequest.UserAgent = "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; .NET CLR 1.0.3705;)";
    myRequest.Accept = "text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8";
    myRequest.Headers.Add("Accept-Language: en-us,en;q=0.5");
    myRequest.Headers.Add("Accept-Encoding: gzip,deflate");
    myRequest.Headers.Add("Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7");
    myRequest.KeepAlive = true;
    myRequest.Headers.Add("Keep-Alive: 300");
    myRequest.ContentType = "application/x-www-form-urlencoded";


    HttpWebResponse myResponse = (HttpWebResponse)myRequest.GetResponse();
    _cookieContainer = myRequest.CookieContainer;

    StreamReader reader = new StreamReader(myResponse.GetResponseStream());
    return reader.ReadToEnd().Trim();
}

public string ProcessResponseNoHeaders(string _url)
{
    if (_url == String.Empty)
        throw new Exception("ProcessResponse: URL not provided");

    // Prepare web request...
    HttpWebRequest myRequest = (HttpWebRequest)WebRequest.Create(_url);
    myRequest.Method = "GET";
    myRequest.CookieContainer = _cookieContainer;

    myRequest.UserAgent = "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; .NET CLR 1.0.3705;)";
    HttpWebResponse myResponse = (HttpWebResponse)myRequest.GetResponse();
    _cookieContainer = myRequest.CookieContainer;

    StreamReader reader = new StreamReader(myResponse.GetResponseStream());
    return reader.ReadToEnd().Trim();
}

public string ProcessResponseCCHeadliner(string _url)
{
    if (_url == String.Empty)
        throw new Exception("ProcessResponse: URL not provided");

    // Prepare web request...
    HttpWebRequest myRequest = (HttpWebRequest)WebRequest.Create(_url);
    myRequest.Method = "GET";
    myRequest.CookieContainer = _cookieContainer;

    myRequest.UserAgent = "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; .NET CLR 1.0.3705;)";
    myRequest.Accept = "text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8";
    myRequest.Headers.Add("Accept-Language: en-us,en;q=0.5");
    myRequest.Headers.Add("Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7");
    myRequest.KeepAlive = true;
    myRequest.Headers.Add("Keep-Alive: 300");
    myRequest.ContentType = "application/x-www-form-urlencoded";


    HttpWebResponse myResponse = (HttpWebResponse)myRequest.GetResponse();
    _cookieContainer = myRequest.CookieContainer;

    StreamReader reader = new StreamReader(myResponse.GetResponseStream());
    return reader.ReadToEnd().Trim();
}

public string ProcessResponseAuthenticateNTLMSecurity(string username, string password, string url)
{
    string ReturnValue = "";

    HttpWebRequest request = (HttpWebRequest)WebRequest.Create(url);
    request.Method = "GET";
    request.KeepAlive = true;
```

```
            request.Accept = @"*/*";

            if (string.IsNullOrEmpty(username) == false && string.IsNullOrEmpty(password) == false)
            {
                NetworkCredential credential = new NetworkCredential(username, password);
                CredentialCache credentialCache = new CredentialCache();
                credentialCache.Add(new Uri(url), "NTLM", credential);
                request.Credentials = credentialCache;
            }

            HttpWebResponse response = null;
            string res = "";
            try
            {
                response = (HttpWebResponse)request.GetResponse();
                StreamReader reader = new StreamReader(response.GetResponseStream());
                res = reader.ReadToEnd().Trim();
            }
            catch
            {
                throw;
            }

            return res;
        }

        public CookieContainer CookieContainer
        {
            get { return _cookieContainer; }
            set { _cookieContainer = value; }
        }
    }
}
```

Process.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using ExtractThesis.Functions;
using Utility.Parser;
using HtmlAgilityPack;
using System.Web;
using ExtractThesis;
using MoreLinq;
using System.Threading;

namespace ExtractThesis.NLM
{
    public class Process
    {
        public static int MAX_RECURSION = 950;
        public static int RECURSION = 0;
        public static ExtractThesisEntities DBContext;
        public static List<NVP> filteredList = new List<NVP>();

        public static CCrawler craw = new CCrawler();

        public static void Work()
        {
            DBContext = new ExtractThesisEntities();

            //Step 1 : Read All links from Page 1
            List<NVP> linkCollection = DoMainPage();


            List<NVP> completeItemList = new List<NVP>();
            //Step 2 : Read out every page from Step 1
            foreach (NVP p in linkCollection)
            {
                completeItemList.AddRange(DoSecondPage(p));
            }
```

```
              //Step 3: Read through Entire Link List and Build Category Chart in filteredList
            foreach (NVP i in completeItemList)
            {
                RecursiveReadAllItems(i.Value);
            }
        }

        public static List<NVP> DoMainPage()
        {
            string url = "http://www.nlm.nih.gov/mesh/trees.html";
            string html = craw.ProcessGetResponse(new Dictionary<string, string>(), url);
            List<NVP> linkCollection = new List<NVP>();

            HtmlDocument doc = new HtmlDocument();
            doc.LoadHtml(html);

            HtmlNodeCollection treeItems = doc.DocumentNode.SelectNodes("html/body/div[@id='wrapper-fluid']/div[@id='container-
fluid']/div[@id='main-body']/div[@id='body']/ol/li");



              //Read from Tree Nodes and fill Name Value Pairs with Links and Name
            foreach (HtmlNode nTree in treeItems)
            {
                HtmlNodeCollection ns = nTree.SelectNodes("ul/li");
                foreach (HtmlNode n in ns)
                {
                    string Name = n.InnerText;
                    string Link = "";
                    if (n.SelectSingleNode("a[1]").Attributes["href"] != null)
                        Link = "http://www.nlm.nih.gov" + n.SelectSingleNode("a[1]").Attributes["href"].Value;
                    linkCollection.Add(new NVP("", Name, Link));
                }
            }

            return linkCollection;
        }
        public static NVP[] DoSecondPage(NVP page)
         {
             string html = craw.ProcessGetResponse(new Dictionary<string, string>(), page.Value);
             HtmlDocument doc = new HtmlDocument();
             doc.LoadHtml(html);

             HtmlNode ul = doc.DocumentNode.SelectSingleNode("html/body/div[@id='wrapper-fluid']/div[@id='container-
fluid']/div[@id='main-body']/div[@id='body']/ul[@class='Level1']");
             List<NVP> allItems = ReadSecondPageRecursive(1, ul);

             return allItems.ToArray();
         }

         public static List<NVP> ReadSecondPageRecursive(int Level, HtmlNode ul)
         {
             NVP listItem = null;
             List<NVP> SecondPageList = new List<NVP>();
             HtmlNodeCollection lis = ul.SelectNodes("li");
             foreach (HtmlNode li in lis)
             {
                 string Name = "";
                 string Link = "";

                 if (li.SelectSingleNode("a").Attributes["href"] != null)
                 {
                     Name = li.SelectSingleNode("a").InnerText;
                     Link = li.SelectSingleNode("a").Attributes["href"].Value;
                     listItem = new NVP(Level.ToString(), Name, Link);
                 }

                 HtmlNode innerUL = li.SelectSingleNode("ul");
                 if (innerUL != null)
                 {
                     listItem.InnerValues = ReadSecondPageRecursive(++Level, innerUL);
                 }

                 if (listItem != null)
                 {
                     SecondPageList.Add(listItem);
                 }
             }
```

```
    return SecondPageList;
}

public static string GetTermFromURL(string url)
{
    string term = "";
    string[] urlSplits = url.Split(new char[] { '?', '&', '=' }, StringSplitOptions.RemoveEmptyEntries);
    for (int i = 0; i < urlSplits.Length; i++)
    {
        string termSearcher = urlSplits[i].ToLower();
        if (termSearcher.Equals("term") && urlSplits.Length > i)
        {
            term = urlSplits[i + 1];
        }
    }

    return term.ToLower();
}

public static void RecursiveReadAllItems(string url)
{
    if (++RECURSION > MAX_RECURSION)
    {
        --RECURSION;
        return;
    }

    if (String.IsNullOrEmpty(url))
    {
        --RECURSION;
        return;
    }
    else
    {
        string html = "";
        bool readSuccess = false;

        for (int o = 0; o < 10; o++)
        {
            try
        {
            html = craw.ProcessGetResponse(new Dictionary<string, string>(), url);
            readSuccess = true;
        }
        catch
        {
            readSuccess = false;
            Thread.Sleep(5000);
        }
        if (readSuccess)
            break;
    }

    if (!readSuccess)
    {
        throw new Exception("Internet Error, Cannot Proceed...");
    }

    HtmlDocument doc = new HtmlDocument();
    doc.LoadHtml(html);
    HtmlNodeCollection infoTableTRs = doc.DocumentNode.SelectNodes("html/body/table/tr");

    string term = GetTermFromURL(url);

    if (infoTableTRs != null)
    {
        NVP details = new NVP();

        foreach (HtmlNode tr in infoTableTRs)
        {
            string th = tr.SelectSingleNode("th") != null ? tr.SelectSingleNode("th").InnerText.Trim() : "";
            string td = tr.SelectSingleNode("td") != null ? tr.SelectSingleNode("td").InnerText.Trim() : "";

            if (th.ToLower().Trim().StartsWith("mesh heading"))
            {
                details.MeshHeading = td;
            }
            else if (th.ToLower().Trim().StartsWith("tree number"))
            {
                details.TreeNumber = td;
            }
    else if (th.ToLower().Trim().StartsWith("annotation"))
    {
        details.Annotation = td;
    }
    else if (th.ToLower().Trim().StartsWith("scope note"))
    {
        details.ScopeNote = td;
    }
    else if (th.ToLower().Trim().StartsWith("date of entry"))
    {
        details.DateOfEntry = td;
    }
    else if (th.ToLower().Trim().StartsWith("entry term"))
    {
        details.EntryTerm = td;
    }
    else if (th.ToLower().Trim().StartsWith("unique id"))
    {
        details.UniqueID = td;
    }
}

var u = DBContext.FilteredLists.FirstOrDefault(d => d.Term == term);

if (u == null)
{
    details.Term = term;
    details.URL = url;
    filteredList.Add(details);


 //Step 4 : Instantly Flush to Database
    foreach (NVP i in filteredList)
    {
        FilteredList l = new FilteredList();
        l.Category = i.Category;
        l.Term = i.Term;
        l.AllowableQualifiers = i.AllowableQualifiers;
        l.Annotation = i.Annotation;
        l.DateOfEntry = i.DateOfEntry;
        l.EntryCombination = i.EntryCombination;
```

```
                l.EntryTerm = i.EntryTerm;
                l.HistoryNote = i.HistoryNote;
                l.MeshHeading = i.MeshHeading;
                l.PreviousIndexing = i.PreviousIndexing;
                l.ScopeNote = i.ScopeNote;
                l.TreeNumber = i.TreeNumber;
                l.UniqueID = i.UniqueID;
                l.URL = url;
                DBContext.AddToFilteredLists(l);
            }

            DBContext.SaveChanges();
            filteredList.Clear();
        }
    }

    //Secondary Scan
    var scanCheck = DBContext.Scans.FirstOrDefault(d => d.Term == term);
    if (scanCheck == null)
    {
        Scan sc = new Scan();
        sc.Term = term;
        DBContext.AddToScans(sc);
    }
    DBContext.SaveChanges();

    HtmlNodeCollection tables = doc.DocumentNode.SelectNodes("html/body/table[2]");

    if (tables != null)
    {
        for (int i = 0; i < tables.Count; i++)
        {
            HtmlNodeCollection trs = tables[i].SelectNodes("tr");
            foreach (HtmlNode tr in trs)
            {
                HtmlNode tdLinkNode = tr.SelectSingleNode("td[@colspan='4']/a");
                if (tdLinkNode != null && tdLinkNode.Attributes["href"] != null)
                {
                    string newURL = "http://www.nlm.nih.gov" + tdLinkNode.Attributes["href"].Value;
                    string newterm = GetTermFromURL(newURL);
```

```
                                Scan check = DBContext.Scans.FirstOrDefault(d => d.Term == newterm);

                                if (check == null)
                                    RecursiveReadAllItems(newURL);
                            }
                        }
                    }
                }
            }
        }
    }
}
```

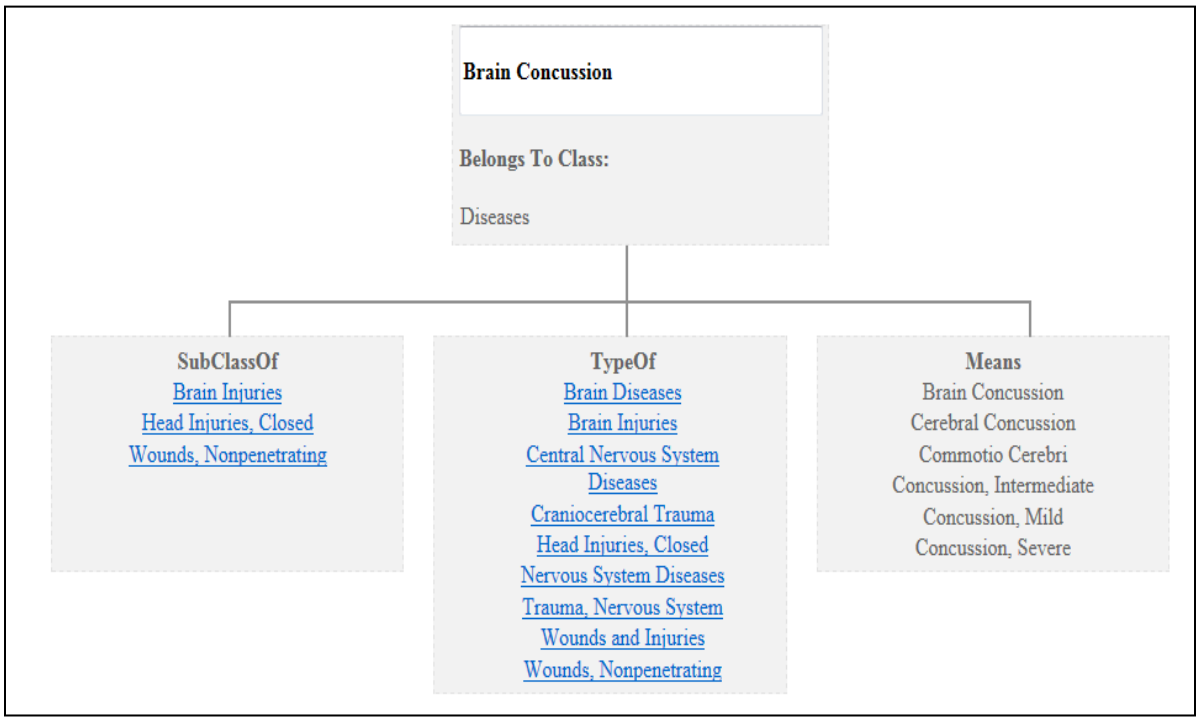
Appendix A.2: Script for Extracting “subclassof”, “typeof”, and “means” relation

```
DECLARE @tmp_MHeading nvarchar(MAX)
      DECLARE @tmp_MTreeNumber nvarchar(1000)
      DECLARE @MHeading_List cursor
      DECLARE @tmpTypeOfMHeading_Predecessor nvarchar(MAX)
      DECLARE @tmpTypeOfMHeading_Leaf nvarchar(MAX)
      DECLARE @TypeOfMHeading_List cursor
      DECLARE @tmp_MUniqueID_Leaf nvarchar(250)
      DECLARE @tmp_MUniqueID_Parent nvarchar(250)
      DECLARE @tmp_ParentMHeading nvarchar(250)
      DECLARE @tmp_TrimmedLeafTreeNumber nvarchar(1000)

      set @MHeading_List = CURSOR FOR
    select MeshHeading, TreeNumber from FilterListSorted where TreeNumber LIKE 'E07%'

      OPEN @MHeading_List

      FETCH NEXT FROM @MHeading_List into @tmp_MHeading,@tmp_MTreeNumber

      WHILE (@@FETCH_STATUS = 0)
      BEGIN
            --Getting the UniqueId of LeafMeSHHeading
            SELECT @tmp_MUniqueID_Leaf = UniqueId FROM FilteredList WHERE MeshHeading = @tmp_MHeading

            --INSERTING subclassof RELATIONS
            IF(len(@tmp_MTreeNumber) <> 3)
            BEGIN
                  SET @tmp_TrimmedLeafTreeNumber =substring(@tmp_MTreeNumber, 1, len(@tmp_MTreeNumber)-4)

                  SELECT @tmp_ParentMHeading = MeshHeading
                  FROM FilterListSorted
                  WHERE TreeNumber LIKE @tmp_TrimmedLeafTreeNumber

                  SELECT @tmp_MUniqueID_Parent = UniqueID
                  FROM FilteredList
                  WHERE MeshHeading = @tmp_ParentMHeading

                  INSERT INTO Relations(Argument1,Relation,Argument2)
                  VALUES (@tmp_MUniqueID_Leaf,'subclassof', @tmp_MUniqueID_Parent)
```

```
        END

        --INSERTING means RELATIONS
        INSERT INTO Relations(Argument1,Relation,Argument2)
        VALUES (@tmp_MUniqueID_Leaf,'means', @tmp_MUniqueID_Leaf + '_0')

        --INSERTING Typeof RELATIONS
        set @TypeOfMHeading_List = CURSOR FOR
        SELECT DISTINCT f2.MeshHeading AS MeSHHeading_Predecessor,f1.MeshHeading AS MeSHHeading_Leaf
        FROM FilterListSorted AS f1,FilterListSorted AS f2
        WHERE f1.TreeNumber LIKE Convert(nvarchar(500),f2.TreeNumber) + '%'
        AND f1.MeshHeading = @tmp_MHeading

        OPEN @TypeOfMHeading_List
        FETCH NEXT FROM @TypeOfMHeading_List into @tmpTypeOfMHeading_Predecessor, @tmpTypeOfMHeading_Leaf

        WHILE (@@FETCH_STATUS = 0)
        BEGIN
              IF(@tmpTypeOfMHeading_Predecessor <> @tmpTypeOfMHeading_Leaf)
              BEGIN
                    INSERT INTO Relations(Argument1,Relation,Argument2)
                    SELECT @tmp_MUniqueID_Leaf,'typeof',f1.UniqueID
                    FROM FilteredList f1
                    WHERE f1.MeshHeading = @tmpTypeOfMHeading_Predecessor
              END
        FETCH NEXT FROM @TypeOfMHeading_List into @tmpTypeOfMHeading_Predecessor, @tmpTypeOfMHeading_Leaf
        END

        FETCH NEXT FROM @MHeading_List into @tmp_MHeading,@tmp_MTreeNumber
  END

  CLOSE @MHeading_List
  DEALLOCATE @MHeading_List
```

Appendix A.3: Sample Results for Different Categories of Our Ontology

1. Diseases



Brain Concussion

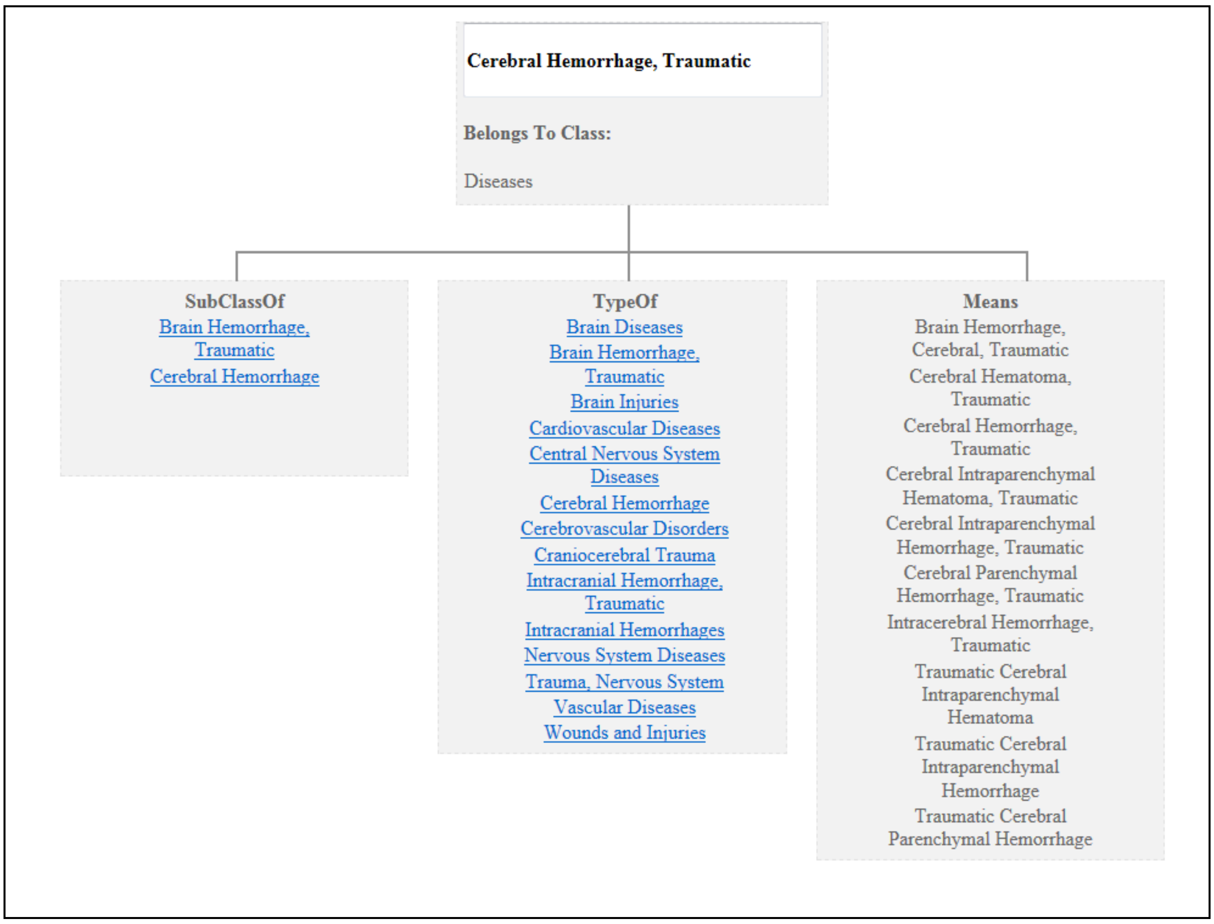
Belongs To Class:

Diseases

SubClassOf
Brain Injuries
Head Injuries, Closed
Wounds, Nonpenetrating

TypeOf
Brain Diseases
Brain Injuries
Central Nervous System Diseases
Craniocerebral Trauma
Head Injuries, Closed
Nervous System Diseases
Trauma, Nervous System
Wounds and Injuries
Wounds, Nonpenetrating

Means
Brain Concussion
Cerebral Concussion
Commotio Cerebri
Concussion, Intermediate
Concussion, Mild
Concussion, Severe



Cerebral Hemorrhage, Traumatic

Belongs To Class:

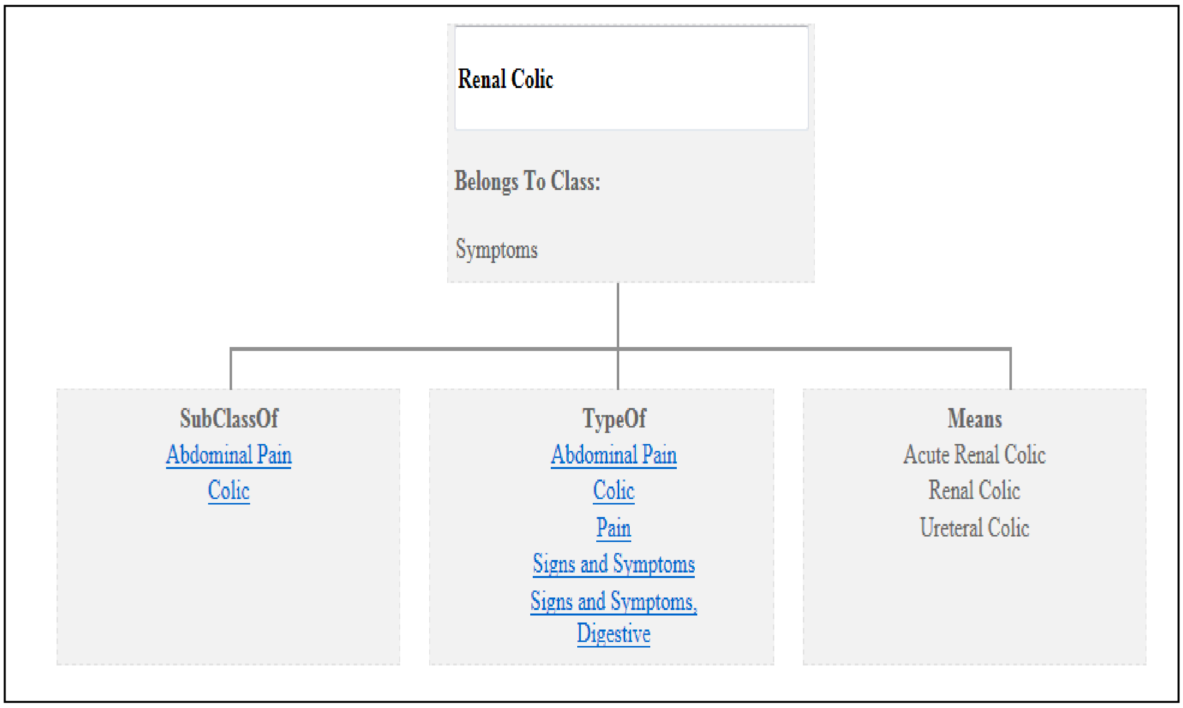
Diseases

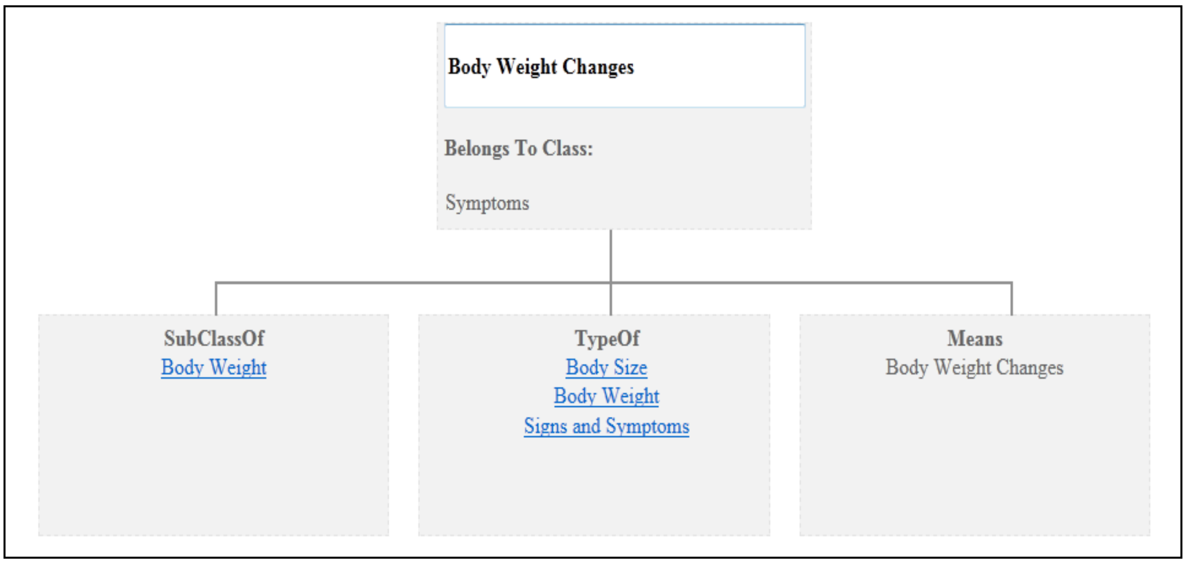
SubClassOf
Brain Hemorrhage, Traumatic
Cerebral Hemorrhage

TypeOf
Brain Diseases
Brain Hemorrhage, Traumatic
Brain Injuries
Cardiovascular Diseases
Central Nervous System Diseases
Cerebral Hemorrhage
Cerebrovascular Disorders
Craniocerebral Trauma
Intracranial Hemorrhage, Traumatic
Intracranial Hemorrhages
Nervous System Diseases
Trauma, Nervous System
Vascular Diseases
Wounds and Injuries

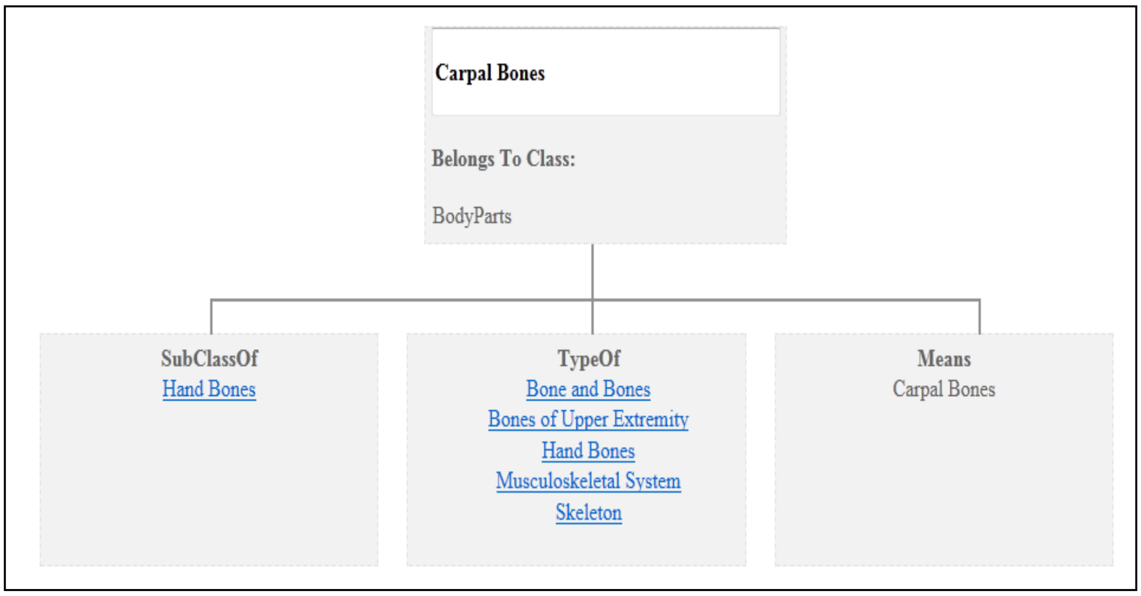
Means
Brain Hemorrhage, Cerebral, Traumatic
Cerebral Hematoma, Traumatic
Cerebral Hemorrhage, Traumatic
Cerebral Intraparenchymal Hematoma, Traumatic
Cerebral Intraparenchymal Hemorrhage, Traumatic
Cerebral Parenchymal Hemorrhage, Traumatic
Intracerebral Hemorrhage, Traumatic
Traumatic Cerebral Intraparenchymal Hematoma
Traumatic Cerebral Intraparenchymal Hemorrhage
Traumatic Cerebral Parenchymal Hemorrhage

2. Symptoms


Renal Colic
Belongs To Class:
Symptoms
SubClassOf
Abdominal Pain
Colic
TypeOf
Abdominal Pain
Colic
Pain
Signs and Symptoms
Signs and Symptoms, Digestive
Means
Acute Renal Colic
Renal Colic
Ureteral Colic


Body Weight Changes
Belongs To Class:
Symptoms
SubClassOf
Body Weight
TypeOf
Body Size
Body Weight
Signs and Symptoms
Means
Body Weight Changes

3. Body Parts


Carpal Bones

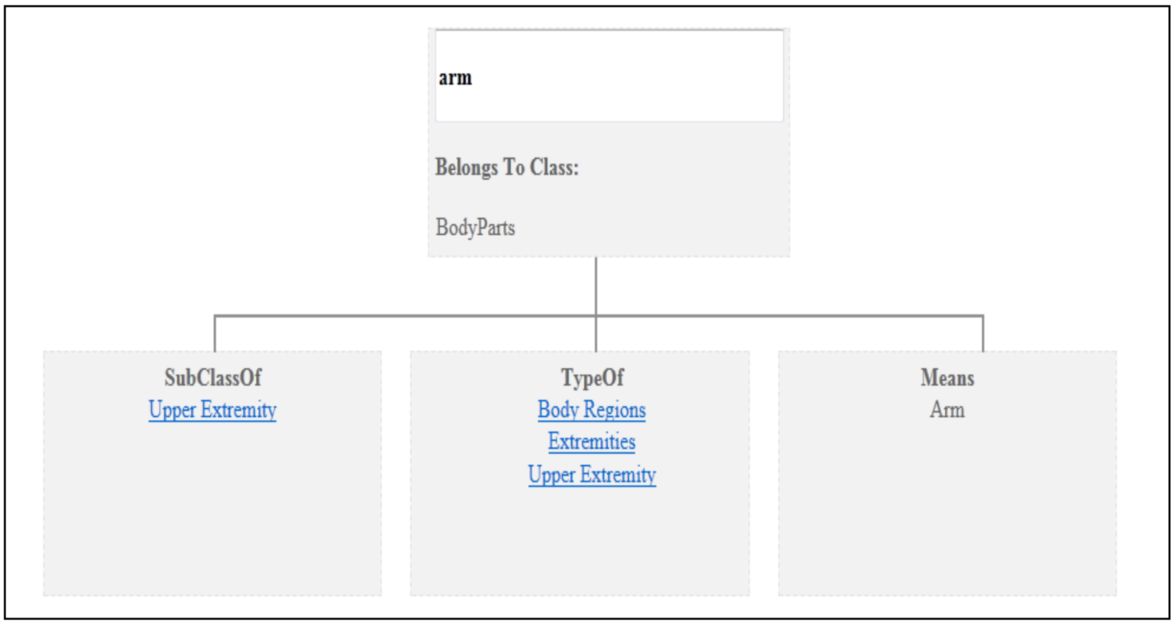
Belongs To Class:

BodyParts

SubClassOf
Hand Bones

TypeOf
Bone and Bones
Bones of Upper Extremity
Hand Bones
Musculoskeletal System
Skeleton

Means
Carpal Bones


arm

Belongs To Class:

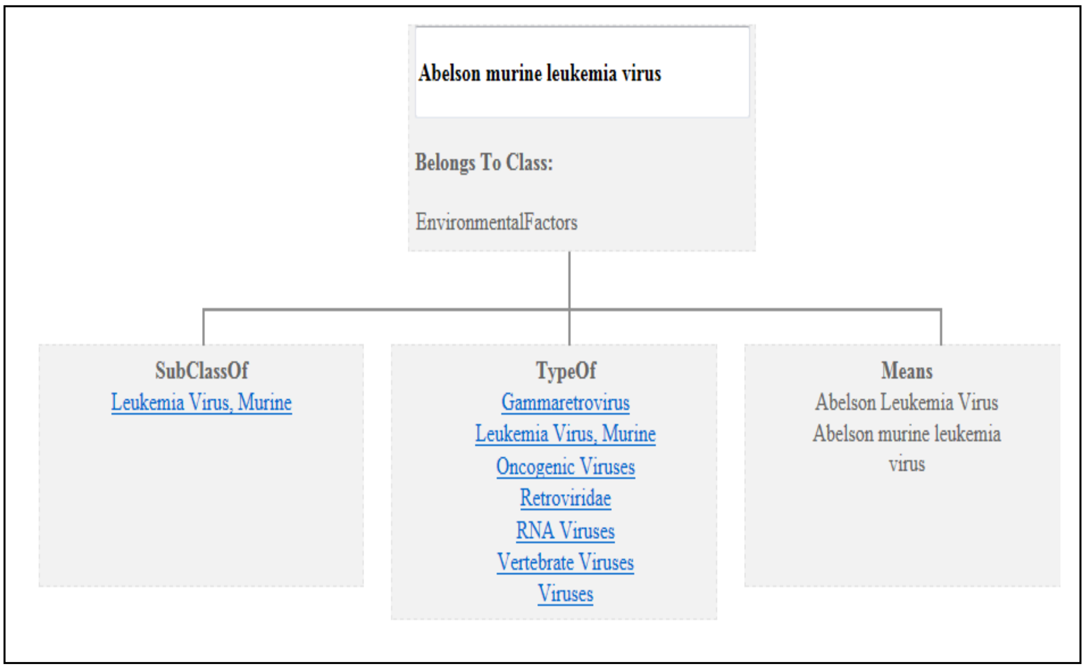
BodyParts

SubClassOf
Upper Extremity

TypeOf
Body Regions
Extremities
Upper Extremity

Means
Arm

4. Environmental Factors



Abelson murine leukemia virus

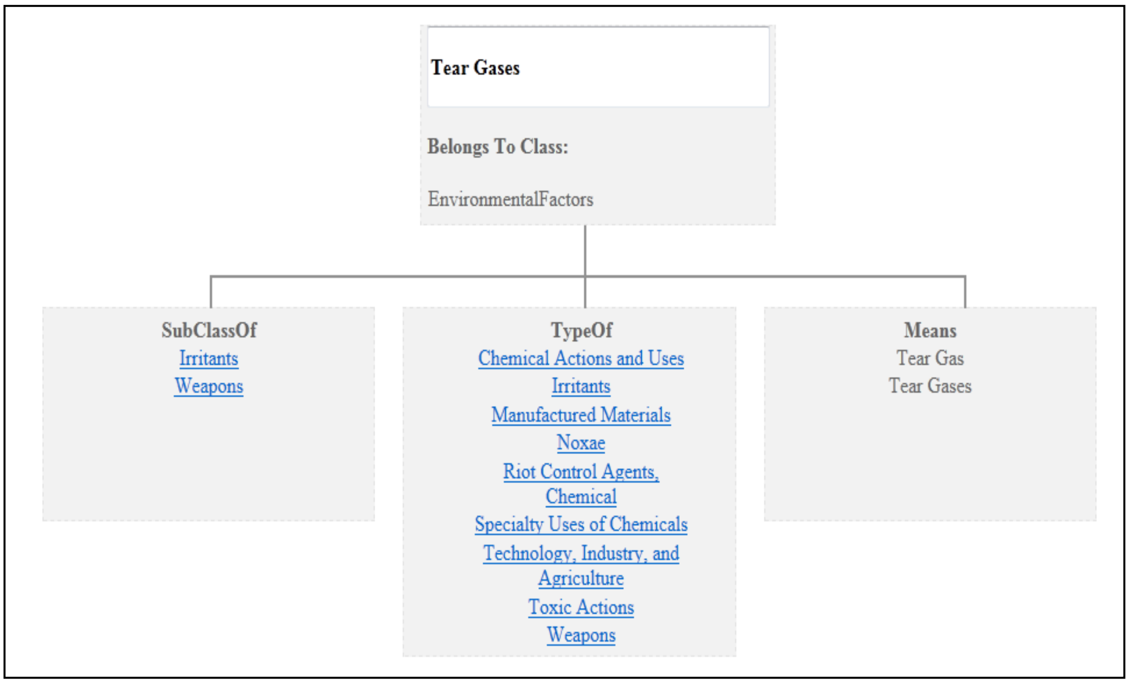
Belongs To Class:

EnvironmentalFactors

SubClassOf
Leukemia Virus, Murine

TypeOf
Gammaretrovirus
Leukemia Virus, Murine
Oncogenic Viruses
Retroviridae
RNA Viruses
Vertebrate Viruses
Viruses

Means
Abelson Leukemia Virus
Abelson murine leukemia virus



Tear Gases

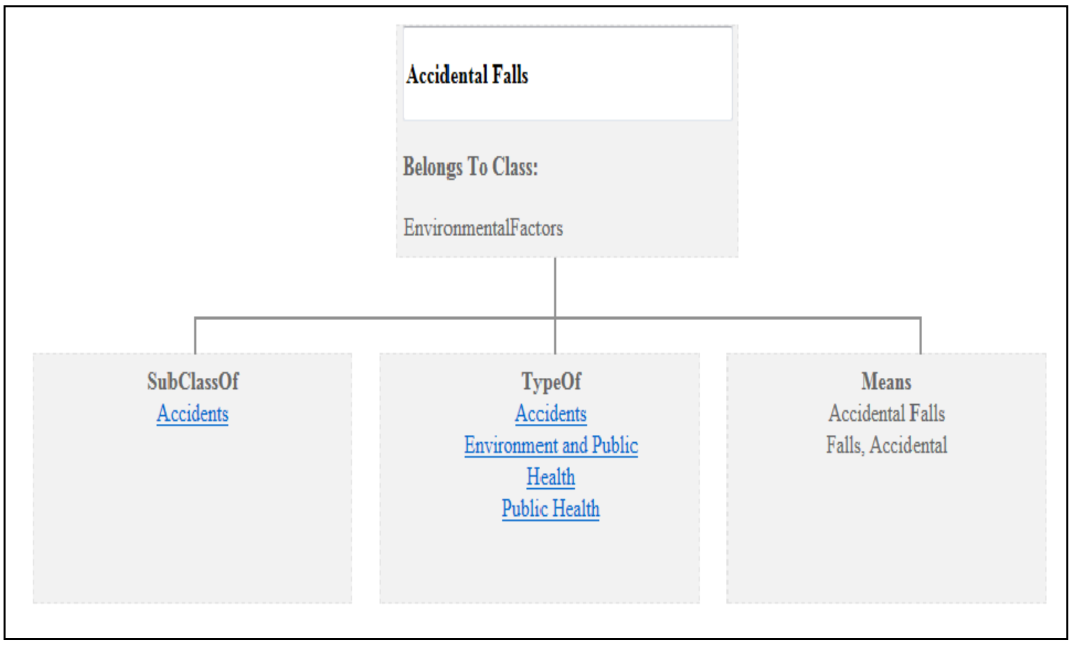
Belongs To Class:

EnvironmentalFactors

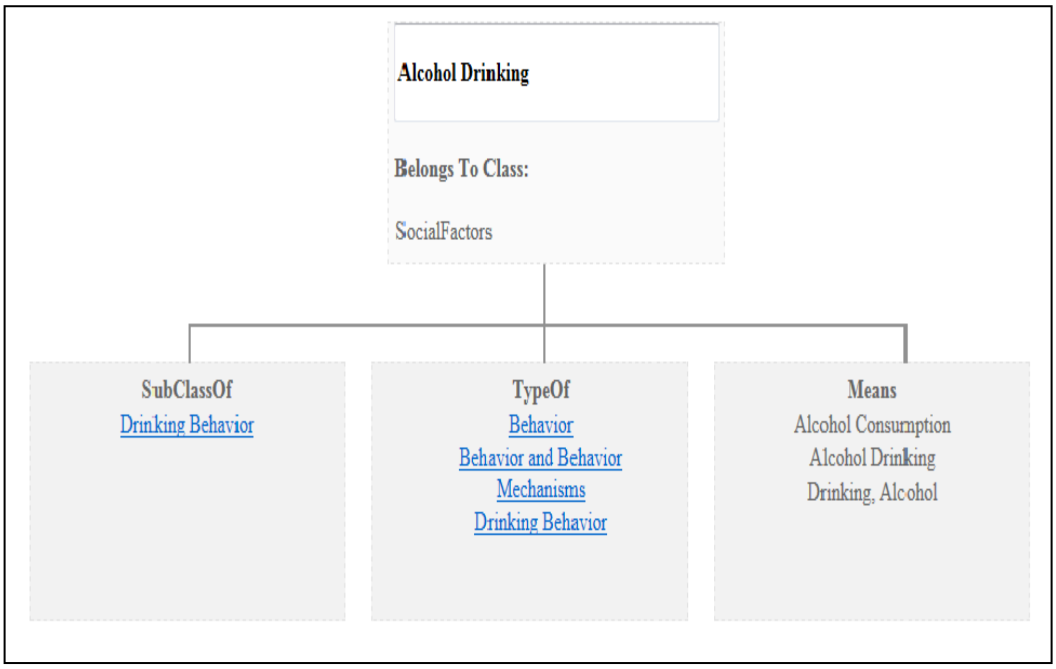
SubClassOf
Irritants
Weapons

TypeOf
Chemical Actions and Uses
Irritants
Manufactured Materials
Noxae
Riot Control Agents, Chemical
Specialty Uses of Chemicals
Technology, Industry, and Agriculture
Toxic Actions
Weapons

Means
Tear Gas
Tear Gases


Accidental Falls
Belongs To Class:
EnvironmentalFactors
SubClassOf
Accidents
TypeOf
Accidents
Environment and Public Health
Public Health
Means
Accidental Falls
Falls, Accidental

5. Social Factors


Alcohol Drinking

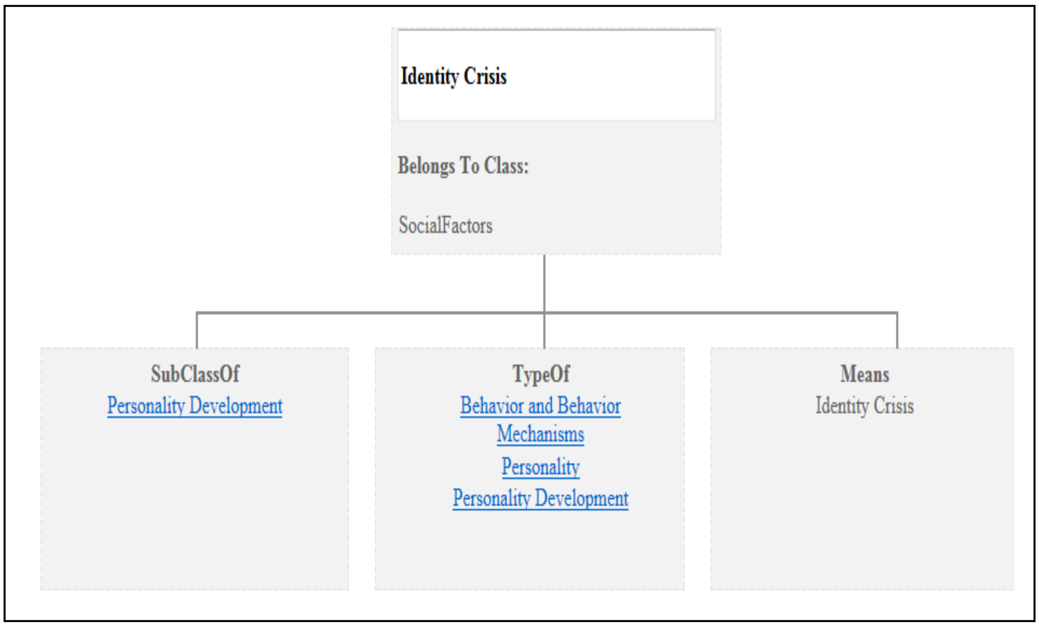
Belongs To Class:

SocialFactors

SubClassOf
Drinking Behavior

TypeOf
Behavior
Behavior and Behavior Mechanisms
Drinking Behavior

Means
Alcohol Consumption
Alcohol Drinking
Drinking, Alcohol


Identity Crisis

Belongs To Class:

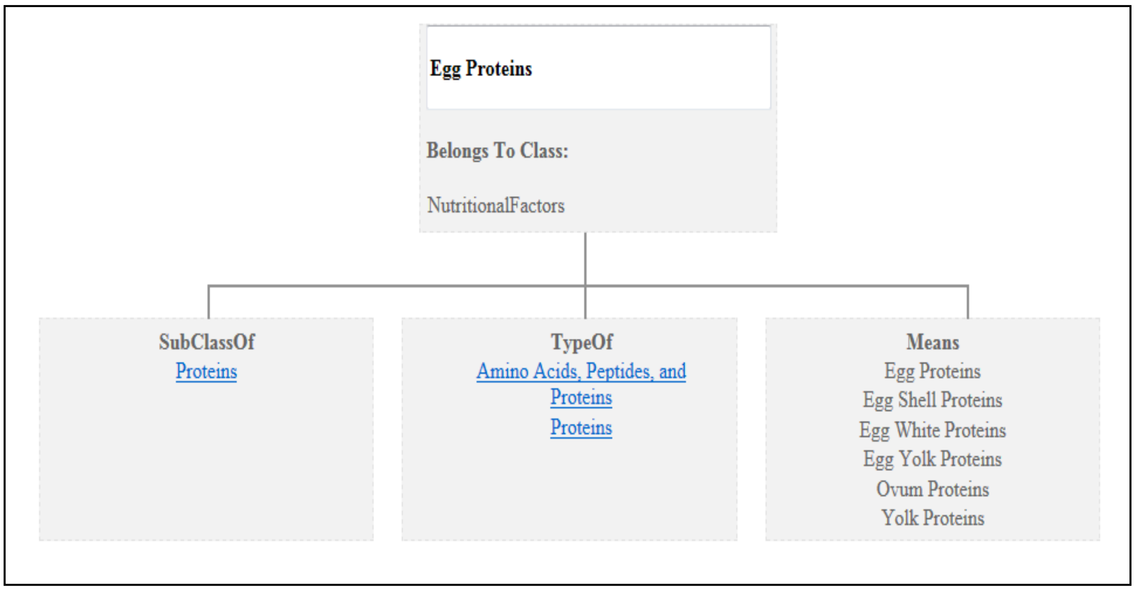
SocialFactors

SubClassOf
Personality Development

TypeOf
Behavior and Behavior Mechanisms
Personality
Personality Development

Means
Identity Crisis

6. Nutritional Factors


Egg Proteins

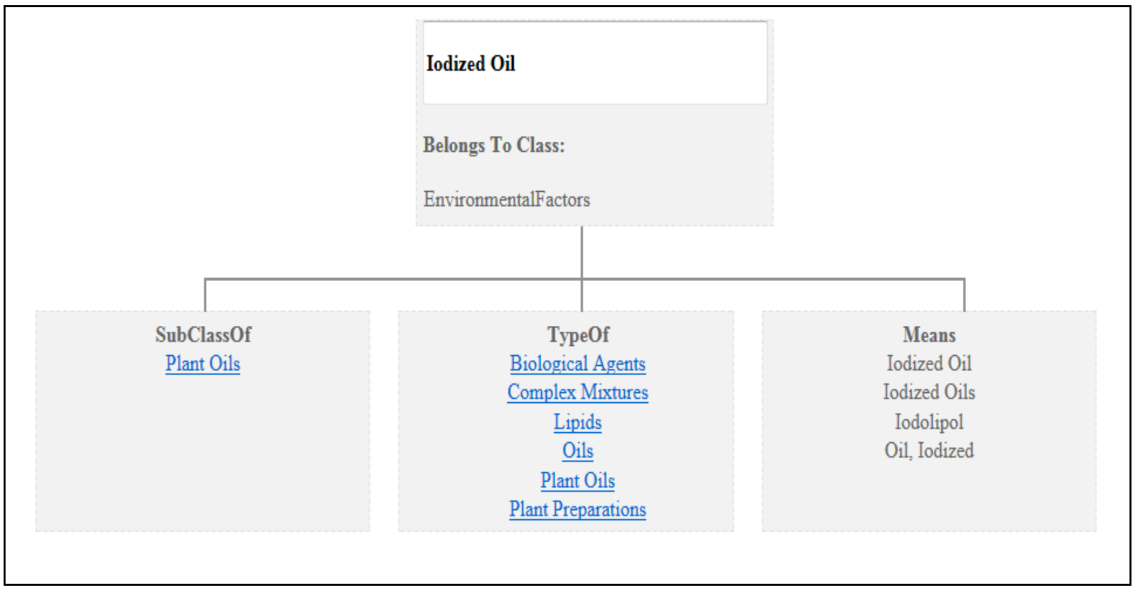
Belongs To Class:

NutritionalFactors

SubClassOf
Proteins

TypeOf
Amino Acids, Peptides, and Proteins
Proteins

Means
Egg Proteins
Egg Shell Proteins
Egg White Proteins
Egg Yolk Proteins
Ovum Proteins
Yolk Proteins


Iodized Oil

Belongs To Class:

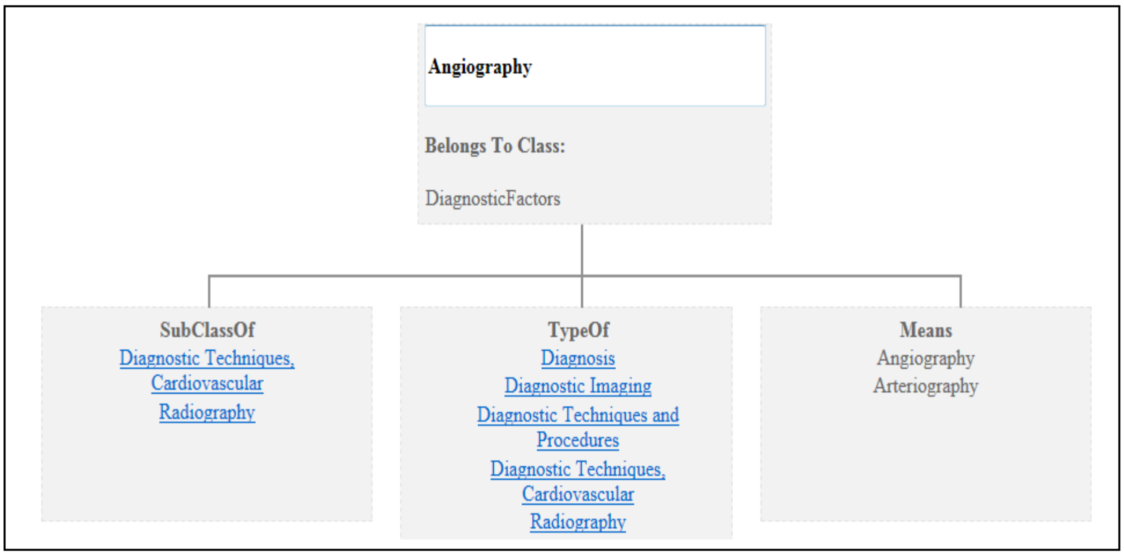
EnvironmentalFactors

SubClassOf
Plant Oils

TypeOf
Biological Agents
Complex Mixtures
Lipids
Oils
Plant Oils
Plant Preparations

Means
Iodized Oil
Iodized Oils
Iodolipol
Oil, Iodized

7. Diagnostic Factors



Angiography

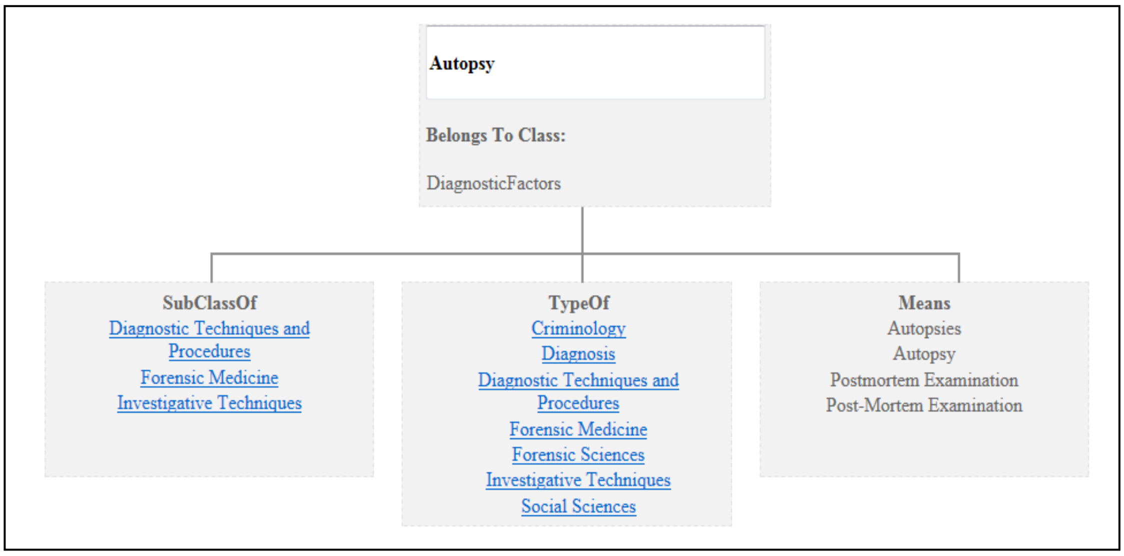
Belongs To Class:

DiagnosticFactors

SubClassOf
Diagnostic Techniques, Cardiovascular
Radiography

TypeOf
Diagnosis
Diagnostic Imaging
Diagnostic Techniques and Procedures
Diagnostic Techniques, Cardiovascular
Radiography

Means
Angiography
Arteriography



Autopsy

Belongs To Class:

DiagnosticFactors

SubClassOf
Diagnostic Techniques and Procedures
Forensic Medicine
Investigative Techniques

TypeOf
Criminology
Diagnosis
Diagnostic Techniques and Procedures
Forensic Medicine
Forensic Sciences
Investigative Techniques
Social Sciences

Means
Autopsies
Autopsy
Postmortem Examination
Post-Mortem Examination

BIBLIOGRAPHY

1. Weikum, Gerhard, and Martin Theobald. "From information to knowledge: harvesting entities and relationships from web sources." Proceedings of the twenty-ninth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems. ACM, 2010.
2. Soualmia, Lina Fatima, Christine Golbreich, and S. J. Darmoni. "Representing the MeSH in OWL: Towards a semiautomatic migration." Proceedings of the KR 2004 Workshop on Formal Biomedical Knowledge Representation. 2004..
3. Hu, Wenyang. Ontology-based web informatics system. Diss. University of Florida, 2002.
4. Flouris, Giorgos, et al. "Ontology change: Classification and survey." Knowledge Engineering Review 23.2 (2008): 117-152.
5. Lenat, Douglas B. "CYC: A large-scale investment in knowledge infrastructure." Communications of the ACM 38.11 (1995): 33-38.
6. Ye, Min, et al. "Text Mining for Building a Biomedical Knowledge Base on Diseases, Risk Factors, and Symptoms.", 2011.
7. Freebase: An entity graph of people, places and things. http://www.freebase.com/.
8. True knowledge: The internet answer engine. http://www.trueknowledge.com/.
9. Wikipedia: The Free Encyclopedia. http://www.wikipedia.org/.
10. Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives, “DBpedia: a nucleus for a web of open data”, In Proceedings of International Semantic Web Conference 2007, 2007.
11. Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. “Yago: a core of semantic knowledge”, In Proceedings of the 16th international conference on World Wide Web, WWW '07, pages 697-706. ACM, 2007. ACM ID: 1242667.

12. Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. “YAGO: a large ontology from wikipedia and WordNet”, Web Semantics: Science, Services and Agents on the World Wide Web, 6(3):203-217, September 2008.
13. Johannes Ho_art, Fabian M. Suchanek, Klaus Berberich, and Gerhard Weikum. “Yago2: A spatially and temporally enhanced knowledge base fromWikipedia”, Research Report MPI-I-2010-5-007, Max-Planck-Institut for Informatik, Saarbrucken, November 2010.
14. Wang, Yafang, et al. "Timely yago: harvesting, querying, and visualizing temporal knowledge from wikipedia." Proceedings of the 13th International Conference on Extending Database Technology. ACM, 2010.
15. Pagel, Philipp, et al. "The MIPS mammalian protein–protein interaction database." Bioinformatics 21.6 (2005): 832-834.
16. Harris, M. A., et al. "The Gene Ontology (GO) database and informatics resource." Nucleic acids research 32.Database issue (2004): D258.
17. Gene Ontology Consortium. The gene ontology (GO) project in 2006. Nucleic Acids Research, 34(Database issue):D322{326, January 2006. PMID:16381878.
18. Minoru Kanehisa and Susumu Goto. KEGG: kyoto encyclopedia of genes and genomes. Nucleic Acids Research, 28(1):27 {30, January 2000.
19. KEGG: The Kyoto Encyclopedia of Genes and Genomes. http://www.genome.jp/kegg/.
20. MeSH: Medical Subject Headings. http://www.nlm.nih.gov/mesh/.
21. PubMed: U.S. National Library of Medicine National Institutes of Health. http://www.ncbi.nlm.nih.gov/pubmed/.
22. Olivier Bodenreider. The uni_ed medical language system (umls): integrating biomedical terminiology. Nucleic Acids Research, 32(suppl 1):D267-D270, 2004.

23. Russ B Altman. PharmGKB: a logical home for knowledge relating genotype to drug response phenotype. Nature Genetics, 39, April 2007.
24. Yueyi Liu, Paul Wise, and Atul Butte. The "etiome": identification and clustering of human disease etiological factors. BMC Bioinformatics, 10(Suppl 2):S14, 2009.
25. Ada Hamosh, Alan F. Scott, Joanna S. Amberger, Carol A. Bocchini, and Victor A. McKusick. Online Mendelian Inheritance in Man (OMIM), a knowledgebase of human genes and genetic disorders. In Nucleic Acids Res, pages D514-517, January 2005
26. OMIM: Online Mendelian Inheritance in Man. http://www.ncbi.nlm.nih.gov/omim.
27. Kevin G. Becker, Kathleen C. Barnes, Tiffani J. Bright, and S. Alex Wang. The Genetic Association Database. Nature Genetics, 36(5):431-432, May 2004.
28. Albert-L_aszl_o Barab_asi. Network medicine - from obesity to the “diseasome”. New England Journal of Medicine, 357(4):404-407, 2007.
29. Kwang-Il Goh, Michael E. Cusick, David Valle, Barton Childs, Marc Vidal, and Albert-Laszlo Barabasi. The human disease network. PROCEEDINGS OF THE NATIONAL ACADEMY OF SCIENCES OF THE UNITED STATES OF AMERICA, 104(21):8685-8690, May 2007.
30. F. M. Suchanek, G. Ifrim, and G. Weikum. “Combining linguistic and statistical analysis to extract relations from web documents”, In KDD, 2006.
31. O. Etzioni, M. J. Cafarella, D. Downey, S. Kok, A.-M. Popescu, T. Shaked, S. Soderland, D. S. Weld, and A. Yates, “Web-scale information extraction in KnowItAll”. In WWW, 2004.
32. M. J. Cafarella, D. Downey, S. Soderland, and O. Etzioni, “KnowItNow: Fast, scalable information extraction from the web”, In EMNLP, 2005.

33. E. Agichtein and L. Gravano, “Snowball: extracting relations from large plain-text collections”, In ICDL, 2000.

34. R. Snow, D. Jurafsky, and A. Y. Ng, “Semantic taxonomy induction from heterogenous evidence”, In ACL, 2006.

35. P. Pantel and M. Pennacchiotti, “Espresso: Leveraging generic patterns for automatically harvesting semantic relations”, In ACL, 2006.

36. H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan, “GATE: A framework and graphical development environment for robust NLP tools and applications”, In ACL, 2002.

37. S. Russell and P. Norvig, “Artificial Intelligence: a Modern Approach”, Prentice Hall, 2002.

38. S. Staab and R. Studer, “Handbook on Ontologies”, Springer, 2004.

39. Spiteller, Verena, "Documentation of the project “MeSH”, Vienna University of Technology, 2007.

40. R. C. Bunescu and M. Pasca, “Using encyclopedic knowledge for named entity disambiguation”, In EACL, 2006.

41. S. Chaudhuri, V. Ganti, and R. Motwani, “Robust identification of fuzzy duplicates”, In ICDE, 2005.

42. W. W. Cohen and S. Sarawagi, “Exploiting dictionaries in named entity extraction: combining semi-markov extraction processes and data integration methods”, In KDD, 2004.

43. N. Chatterjee, S. Goyal, and A. Naithani, “Resolving pattern ambiguity for english to hindi machine translation using WordNet”, In Workshop on Modern Approaches in Translation Technologies, 2005.

44. W. Hunt, L. Lita, and E. Nyberg, “Gazetteers, wordnet, encyclopedias, and the web: Analyzing question answering resources”, Technical Report CMU-LTI-04-188, Language Technologies Institute, Carnegie Mellon, 2004.

45. G. Ifrim and G. Weikum, “Transductive learning for text classification using explicit knowledge models”, In PKDD, 2006.