The Dissertation Committee for Syed Hamid Ali Tirmizi certifies that this is the approved version of the following dissertation:

Ontology as a means for Systematic Biology

Committee:

Daniel Miranker, Supervisor

Don Batory

Kristen Grauman

Robin Gutell

Bruce Porter

Ontology as a means for Systematic Biology

by

Syed Hamid Ali Tirmizi, B.S., M.S.

Dissertation

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Doctor of Philosophy

The University of Texas at Austin

May 2011

بِسْمِ اللهِ الرَّحْمٰنِ الرَّحِيْمِ
اَلْحَمْدُ لِلّٰهِ رَبِّ الْعَالَمِيْنَ

Dedicated to my parents

and to the memory of Hakeem Muhammad Saeed

Acknowledgments

My deepest gratitude goes to my advisor Professor Daniel Miranker. This dissertation and any other accomplishments during the course of my Ph.D. would not have been possible without his guidance, encouragement and support.

I would like to thank Professor Don Batory for his guidance, in particular during the early stages of my doctoral studies. I also thank him and the rest of my dissertation committee: Professors Bruce Porter, Robin Gutell and Kristen Grauman, for their insightful comments and discussions on my work.

My thanks also go to all past and present members of my research group who have helped me in my studies and research, particularly to Kerin Claeson, Juan Sequeda, Smriti Ramakrishnan, Rui Mao, Willard Briggs and Lee Thompson. I also thank Professor Timothy Rowe and Dr. Julian Humphries for educating me about systematic biology and Professor Paula Mabee for providing me with the data from the *Cypriniformes* Tree of Life project that was critical to the evaluation of my work.

I thank all my teachers from school and college for all the knowledge that forms the foundation of my research. I especially thank Professors Waseem Ikram, Aftab Maroof and Ayub Alvi for making computer science enjoyable for me during my undergraduate years, and seeding the idea of a doctorate degree.

I have a number of great friends whose company I have enjoyed over the years and who have generously helped me in my various endeavors. I thank all of my friends. In particular, Tauseef Rab, Bilal Janjua and Abbas Hassan deserve my gratitude for making my daily life as a graduate student enjoyable. I also thank: Omar Zia, Omer Shahid, Ramoza Ahsan and Farheen Omar for their help during my graduate admission process; Adnan Wasim and Ali Ibrahim for helping me practice my dissertation proposal; and Rashid Kaleem, Amber Hassaan, Faisal Iqbal and Owais Khan for their time and comments on my dissertation defense rehearsal.

Ever since my arrival in the USA for graduate studies, my eldest aunt *Khala Sahab's* place has been my home away from home. I have looked forward to visiting her and her family during my vacations, during tough phases of my studies, and even for having a surgery. They have always welcomed me and cared for me. I thank *Khala Sahab* and each member of her family.

The greatest blessings in my life are my parents. Each of my successes is a tribute to their infinite and unconditional love. I hope to be a source of comfort for them now and for the rest of my life. I thank *Ammi* and *Abba Jan*, and the rest of my family: *Dadi Jan*, *Aapi*, Akif *bhai*, *Bhaiya*, Samira, Marium and Hammad, for their love, support and patience. They have enabled me to accomplish my goals.

Syed Hamid Ali Tirmizi

Ontology as a means for Systematic Biology

Syed Hamid Ali Tirmizi, Ph.D.

The University of Texas at Austin, 2011

Supervisor: Daniel P. Miranker

Biologists use ontologies as a method to organize and publish their acquired knowledge. Computer scientists have shown the value of ontologies as a means for knowledge discovery. This dissertation makes a number of contributions to enable systematic biologists to better leverage their ontologies in their research.

Systematic biology, or phylogenetics, is the study of evolution. “Assembling a Tree of Life” (AToL) is an NSF grand challenge to describe all life on Earth and estimate its evolutionary history. AToL projects commonly include a study a taxon (organism) to create an ontology to capture its anatomy. Such anatomy ontologies are manually curated based on the data from morphology-based phylogenetic studies. Annotated digital imagery, morphological characters and phylogenetic (evolutionary) trees are the key components of morphological studies.

Given the scale of AToL, building an anatomy ontology for each taxon manually is infeasible. The primary contribution of this dissertation is automatic inference and concomitant formalization required to compute anatomy ontologies. New anatomy ontologies are formed by applying transformations on an existing anatomy ontology for a model organism. The conditions for the transformations are derived from observational data recorded as morphological characters. We automatically created the *Cypriniformes* Gill and Hyoid Arches Ontology using the morphological character data provided by the *Cypriniformes* Tree of Life (CTOL) project.

The method is based on representing all components of a phylogenetic study as an ontology using a domain meta-model. For this purpose we developed Morphster, a domain-specific knowledge acquisition tool for biologists.

Digital images often serve as proxies for natural specimens and are the basis of many observations. A key problem for Morphster is the treatment of images in conjunction with ontologies. We contributed a formal system for integrating images with ontologies where images either capture observations of nature or scientific hypotheses. Our framework for image-ontology integration provides opportunities for building workflows that allow biologists to synthesize and align ontologies.

Biologists building ontologies often had to choose between two ontology systems: Open Biomedical Ontologies (OBO) or the Semantic Web. It was critical to bridge the gap between the two systems to leverage biological ontologies for inference. We created a methodology and a lossless round-trip mapping for OBO ontologies to the Semantic Web. Using the Semantic Web as a guide to organize OBO, we developed a mapping system which is now a community standard.

Table of Contents

Chapter 1

Introduction

The use of ontologies in biology can be recognized since at least as far back as the work of Carolus Linnæus (*Systema Naturæ*, 1735-1758 [1, 2]) on the classification of organisms in the form of a taxonomy now known as the Linnaean Taxonomy, and in Charles Darwin's sketches of evolutionary trees in his notebook on Transmutation of Species in 1837-1838 [3]. However, the use of the term 'ontology' among biologists and their efforts to express biological knowledge in ontological form seem to have gained traction since the Gene Ontology project [4].

In this context, ontologies are an artificial intelligence tool for capturing the concepts within a domain, as well as the attributes of the concepts and the relationships among the concepts. They are a convenient tool for representing scientific knowledge [5, 6]. Ontology development is an expensive and error prone process that often requires the involvement of trained knowledge engineers to solicit knowledge from domain experts, understand it, and encode it into an ontology [7].

1.1 State of Systematic Biology

Systematic biology or phylogenetics is the study of biological diversity and its origins. It focuses on understanding evolutionary relationships among taxa (organisms,

singular: taxon). One of the grand challenges in this domain is "Assembling a Tree of Life" (AToL), i.e. to describe all life on Earth and estimate its evolutionary history [8].

Currently the most significant use of ontologies in systematic biology is to capture the knowledge about taxon anatomies [9, 10]. These ontologies are called anatomy ontologies or *Nomina Anatomica*. This compares with molecular biology where ontologies such as the Gene Ontology are aimed at standardizing gene representation and attributes across databases. Building anatomy ontology for the taxon under study is often a significant part of an AToL project. However, building the ontology is a laborious process.

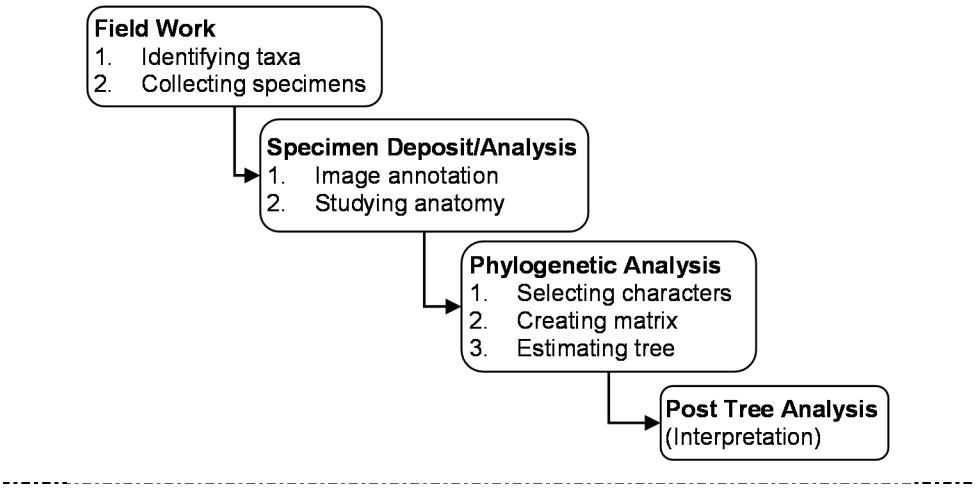


Figure 1.1: Typical workflow for systematic biologists.

In morphology-based phylogenetic studies, it is common for systematic biologists to go out in the field to collect specimens of taxa under study (Figure 1.1). These specimens are often deposited into museums or other natural history collections, and digital proxies created for further study. The digital proxies consist of 2D images and 3D models obtained from *computed tomography* (CT) scans. This imagery is then carefully

segmented and annotated, and becomes the primary means for further study. Using this imagery, biologists study anatomies of organisms, make scientific observations and identify features of interest, called *morphological characters and character states*. Characters and character states from multiple taxa are aligned in the form of a data matrix (or *character matrix*). A tree reconstruction algorithm uses the data matrix to produce one or more *phylogenetic trees*, each representing possible evolutionary relationships among the taxa under consideration. These trees are then used to draw conclusions in the diverse areas of biology, environment, medicine, agriculture etc.

Computer technology is extensively employed in each aspect of the workflow of phylogenetics. This includes the use of popular graphics tools such as Adobe Photoshop and Illustrator for image annotations, ontology editors such as OBO-Edit [11] and Protégé [12] for building ontologies for domains such as taxon anatomies (*Nomina Anatomica*), and building character matrices and generating phylogenetic trees using tools like Mesquite [13] and an array of tree reconstruction algorithms based on parsimony and maximum likelihood etc. [14, 15].

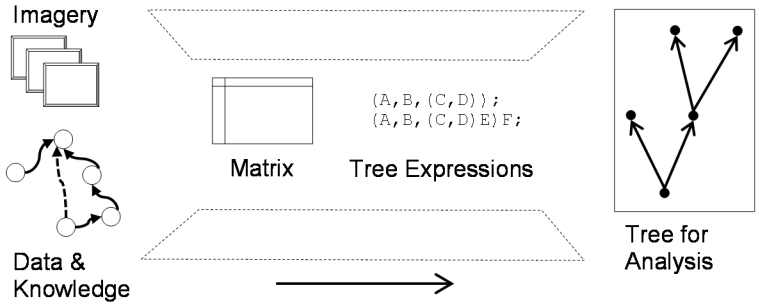


Figure 1.2: The workflow for systematic biologists starts with a lot of imagery and explicit evidence, which remains disconnected from the results of the later stages in the workflow.

While significant progress has been made in the use of technology in this domain, each aspect of the workflow remains disconnected from the rest. To date, the images, ontology and comparative studies all remain in isolated repositories or literature text, and hence disconnected from each other. Consider the example of NIH designated model organism *Danio rerio* (common name: zebrafish): as a model organism zebrafish has been extensively studied. A mass of digital imagery has been produced for zebrafish. An authoritative anatomy ontology called Zebrafish Anatomy ontology (ZFA) [9] has been created and published. As a model organism, it has also been used in comparative studies for building character matrices and generating trees.

The divide exists due to a lack of consideration for broader issues such as data representation, storage, and integration for the overall workflow. Annotations are often embedded onto the images. These images form the persistent authoritative definitions of character states. Character states are subsequently integer coded and organized into character matrices. Tree reconstruction algorithms use these matrices to generate trees labeled only with integer coded features. Hence the computed representation loses all the biological semantics in the process. In addition, ontologies such as *Nomina Anatomica* are considered a final publication mechanism for the acquired knowledge, which is in contrast to computer science or artificial intelligence where ontologies are a means for inference and knowledge integration. As a result, the components and results of a study are disconnected from the original images and observations, i.e. the evidence for the results (Figure 1.2).

Biomedical ontology building projects are faced with a choice between the available ontology languages provided by ontology-based systems Open Biomedical Ontologies (OBO) [16] and the Semantic Web [17]. OBO emerged from the Gene

Ontology (GO) project [4], and hosts over 100 biomedical ontologies including model ontologies such as Zebrafish Anatomy [9], Teleost Anatomy Ontology [10] and Adult Mouse Gross Anatomy [18]. The Semantic Web is an evolving extension of the World Wide Web based on ontologies. Some important biomedical ontologies such as NCI Thesaurus [19] and BioPAX [20] have been modeled using Web Ontology Language (OWL), the ontology language for the Semantic Web. The lack of a bridge between the two ontology systems has been responsible for preventing biologists working with OBO from reusing knowledge from existing ontologies in OWL, and vice versa.

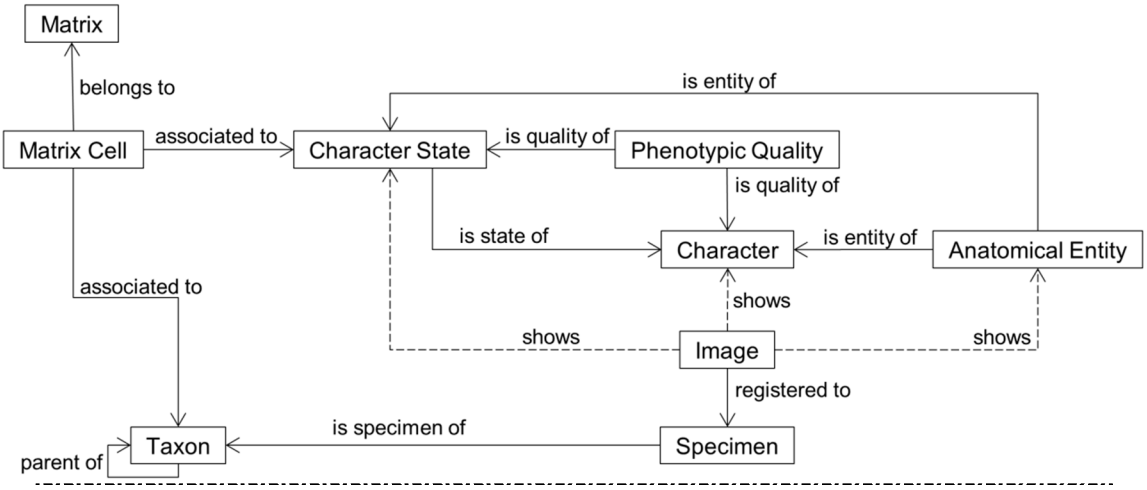


Figure 1.3: A meta-model for phylogenetic studies.

1.2 GOALS AND SCOPE OF RESEARCH

This dissertation entails a vision where systematic biologists conduct their work, find additional data, utilize relevant scientific data and publish results in an integrated fashion. We consider ontologies to be a malleable and powerful tool suitable for achieving these goals. Furthermore, we believe that this use of ontologies allows

systematic biologists to infer new knowledge that is contained but not obvious in their data.

Maxilla, anterior process, presence: absent (0); present (1)

Maxilla, maxillary fenestra, shape: circular (0); oval (1)

Calyx, circumference: 1-2 cm (0); 3-5 cm (1)

Figure 1.4: Examples of character statements taken from biology literature [21].

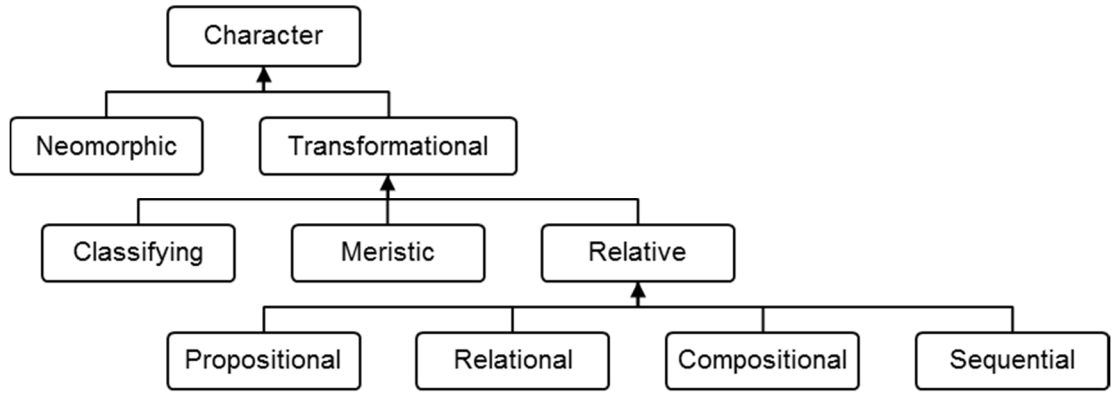


Figure 1.5: A taxonomy of character types.

In this dissertation, we focus on using ontologies for knowledge discovery in systematic biology. In order to achieve this goal, we also resolve issues regarding the representation and integration of data in this domain.

Our main contribution is a framework for automatic inference of new *Nomina Anatomica* ontologies based on an existing model organism anatomy and the knowledge contained in a phylogenetic study. There are two key aspects to this problem.

First, in order to perform inference on the data in a phylogenetic study, it needs to be represented as a unified knowledgebase, i.e. an ontology. We have created a meta-

model (see Figure 1.3) for phylogenetic studies that enables us to represent all aspects of phylogenetic studies, i.e. anatomical entities, characters, character states etc., as a single ontology. Character statements, i.e. characters and character states, are critical concepts in this domain, typically expressed in natural language (Figure 1.4). It is critical for a representation suitable for knowledge inference to provide means for precisely capturing the meaning of each character statement. We realized that character statements can be classified into different types. Hence, a novel contribution in this work is a taxonomy of character types (Figure 1.5), and their corresponding logical signatures in the ontology. A character statement represented in its appropriate signature is well-defined, i.e. no information is lost in its transformation to its logical form, and its natural language statement may be understood from its ontology representation.

Second, we have devised an algorithm for inferring a new anatomy ontology by applying transformations on a model anatomy ontology. The knowledge in a morphological character matrix informs the algorithm of the features of interest, i.e. the features that may need to be transformed, and a phylogenetic tree provides the knowledge of the exact evolutionary changes and hence the sequence of transformations for the algorithm. A fully detailed explanation of this work is present in Chapter 3.

We have used our inference mechanism on the morphology data curated by the biologists on the Cypriniformes Tree of Life (CTOL) project, and created an anatomy ontology for *Cypriniformes* gill and hyoid arches from the model ontology Teleost Anatomy Ontology.

In order to enable biologists to express their data in appropriate ontological form for such knowledge inference opportunities, we have created a knowledge acquisition tool, Morphster [22]. Our meta-model for phylogenetic studies is the underlying data

model for Morphster. A more detailed introduction to the Morphster project is present in Chapter 4. Towards the creation of Morphster, we have made the following contributions.

We have created a formal system for integrating images with ontologies where images either serve as exemplars for phylogenetic observations or for capturing hypotheses in the workflow of a phylogenetic study (Chapter 5). Our framework for image-ontology integration provides opportunities for building workflows that allow biologists to build and align their ontologies without the involvement of knowledge engineers. It also improves upon the image retrieval capabilities of existing ontology based image retrieval systems.

We have also contributed a methodology and a lossless round-trip mapping for OBO ontologies to the Semantic Web [23], bridging the gap between the two ontology systems (Chapter 6). We use the organization of the Semantic Web as a guide to study and organize OBO, hence making it easy to identify straightforward mappings as well as the differences between the two ontology languages. We contributed our methodology and the mapping towards the development of the official standard mapping of the Gene Ontology project and the standard for biomedical ontologies. Our Java implementation of the standard mapping is a part of the open-source Gene Ontology repository and used in major ontology tools in this domain.

Chapter 2

Background

2.1 INTRODUCTION TO ONTOLOGIES

In philosophy, ontology is the study of existence. In knowledge-based systems, it is a vocabulary of a set of objects and the describable relationships among them [24].

One of the definitions of an ontology in computer science is as follows:

> *An ontology defines the terms used to describe and represent an area of knowledge. Ontologies are used by people, databases, and applications that need to share domain information (...). Ontologies include computer-usable definitions of basic concepts in the domain and the relationships among them (...). They encode knowledge in a domain and also knowledge that spans domains. In this way, they make that knowledge reusable. [25]*

Ontologies are usually expressed in formal languages that allow detailed and accurate descriptions of concepts and relationships. Another definition of ontology states:

> *In such an ontology, definitions associate the names of entities in the universe of discourse (e.g., classes, relations, functions, or other objects) with human-readable text describing what the names are*

meant to denote, and formal axioms that constrain the interpretation and well-formed use of these terms. [24]

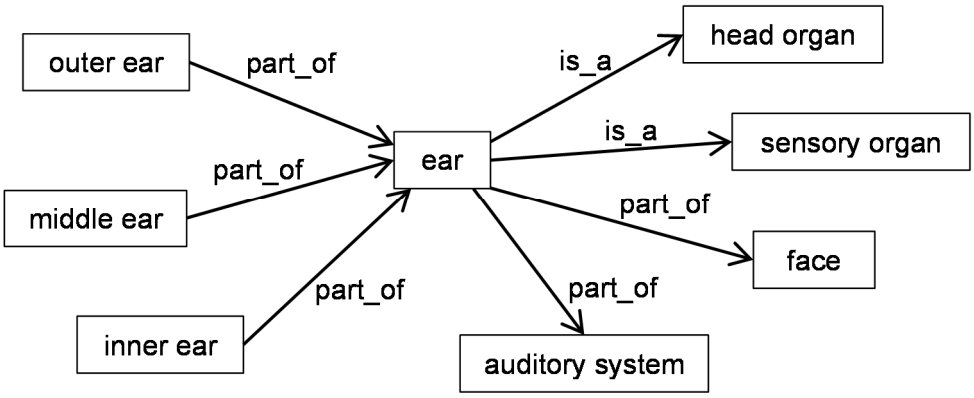


Figure 2.1: A part of the mouse adult gross anatomy ontology that depicts classes and relationships among them. This portion of the ontology describes *ear*: it is a part of the face and the auditory system of mouse, it is also a head organ and a sensory organ, and it has 3 parts.

Figure 2.1 provides an example of a simple ontology. It has been taken from Mouse Adult Gross Anatomy ontology [18] and shows classes or terms (*ear*, *face*, *auditory system* etc.) in boxes, and their relationships (*is_a*, *part_of* etc.) as labeled directed edges.

Ontologies range from light taxonomies and classifications to fully axiomatized theories. Recently, ontologies have been adapted in many research and business communities as a tool for sharing and expressing domain knowledge. Among scientific domains, ontologies are extensively used in areas like artificial intelligence [5, 6], the Semantic Web [25, 26] and biology [4, 9, 27] as a form of knowledge representation in intelligent systems.

Semantic Web Technology

The Semantic Web is an ontology-based extension of the World Wide Web. While the current Web focuses on the interchange of documents, the Semantic Web vision aims to create a universal medium for integration of data. In order to achieve this goal, the Semantic Web provides expressive languages for recording information about the data, and their relationships. This allows humans and machines to find, share and integrate information easily.

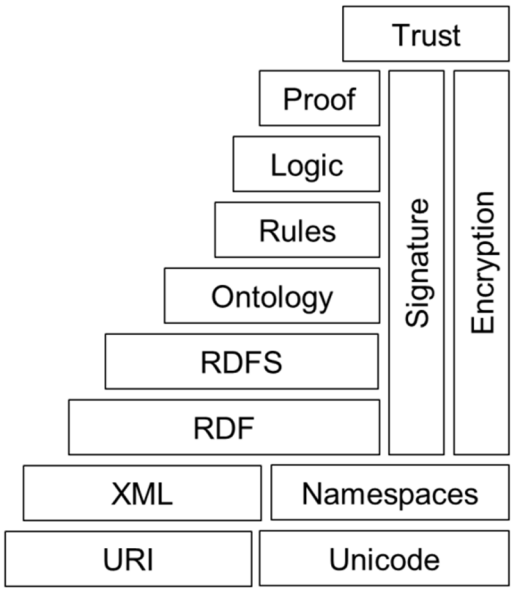


Figure 2.2: The Semantic Web layer cake that lists and presents the organization of the technologies provided by the Semantic Web.

Ontologies allow specification of semantics of data in a way that is usable by Web applications and intelligent software agents. Therefore, ontologies can be used to improve existing Web applications and to provide new ways of leveraging the content available on the Web [17].

The key technologies in the Semantic Web infrastructure (see Figure 2.2) [28] are:

- Extensible Markup Language (XML) [29] is a language that provides arbitrary structure to documents by allowing user-defined markup tags.
- Resource Description Framework (RDF) [30] is used to express meaning of data using *triples*. A triple is a binary predicate and defines a relationship between two entities. RDF triples can be expressed using XML.
- Universal Resource Identifiers (URIs) [31] identify entities, either classes or relationship types, present on the Semantic Web. This means that each entity gets a globally unique identifier that can be accessed by everyone on the Web.
- RDF Schema (RDF-S) [32] and Web Ontology Language (OWL) [33, 34] are used for describing ontologies. RDF Schema allows description of valid classes and relationship types, and some properties like subclasses, domains, ranges etc. OWL provides constructs for describing richer content and provides ontology and concept level annotations, set combinations, equivalences, cardinalities, deprecations etc.
- Languages like SPARQL are available for querying RDF-based ontologies [35].
- Other important components in Semantic Web vision are rule languages and inference methods [36, 37].

Ontologies in Biomedicine

Ontologies are very important to scientific research and discovery in biomedicine. Over 200 biomedical ontologies are available on the NCBO BioPortal [38]. Typically, these ontologies are used either for publishing results or as controlled vocabularies of

standard terms for use across biological studies. In particular, an early project on making a controlled vocabulary was EcoCyc [39, 40], with the aim of providing a comprehensive encyclopedia of *Escherichia coli* biology.

Open Biomedical Ontologies (OBO) Foundry [16] is an effort under the US National Center for Biomedical Ontology (NCBO) to create and share ontologies for use across different biomedical domains. OBO has its own ontology language that supports many important ontologies including the Gene Ontology [4] and anatomies of model organisms such as zebrafish [9] and mouse [18]. Most AToL funded projects investigate evolutionary relationships among a group of related organisms. Building ontologies is often central to these projects [9, 10, 41].

An ontology in the OBO format consists of two parts; the first part is the header that contains tag-value pairs describing the ontology, and the second part contains the domain knowledge described using *term* and *typedef* (more commonly known as a relationship type) stanzas [42]. A stanza may define and describe a term, a typedef or an instance using a collection of tag-value pairs. The terms and typedefs defined in an OBO ontology are assigned local identifiers and namespaces. Relationships between different terms are expressed using the 'relationship' tag.

The OBO format is human friendly. Therefore, it is easy for domain experts to understand it and express their knowledge in this language. Useful GUI-based tools like OBO-Edit [11] are available for building ontologies in the OBO format.

As OBO continues to evolve as a language and hosted content, there is emphasis on formalizing the syntax and semantics of OBO format. Also, given the ongoing adoption of ontologies by the biomedical community and emerging new ontology building projects, OBO Foundry has developed standard ontologies such as the Relations

Ontology [27], which provide consistent and unambiguous formal definitions of the typedefs used in such ontologies. While this effort is designed to assist developers and users in avoiding errors in ontology building, it also promises to simplify the process of ontology alignment in the future for the OBO community.

Ontology Engineering and Tools

Ontology engineering is a hard problem that requires interaction between subject matter experts and knowledge engineers [7]. Subject matter experts are the primary source of knowledge and knowledge engineers are trained in encoding the knowledge into a formal ontological form. Hence, knowledge extracted from the subject matter experts is encoded into ontologies by knowledge engineers. This two-step process of ontology engineering is expensive in terms of time and efficiency, and sometimes knowledge engineers may not fully understand the knowledge that can introduce errors in the knowledge base.

Currently, common ontology editing tools work by providing a tree-like view of the ontology. A knowledge engineer editing the ontology locates appropriate parts of the ontology by navigating through its tree representation. In such a system, fundamental ontology editing tasks become cumbersome and prone to errors. For example, in order to add a new concept to the ontology, a class needs to be created independent of other parts of the ontology, and then connected to various concepts in the ontology using predefined or user defined relationship types. As the ontology grows, it becomes increasingly difficult to keep track of the progress or to find out if an error has occurred.

Some ontology editing tools that are in common use today are the following:

- OBO-Edit [11] is an open source ontology editor written in Java, developed by the Berkeley Bioinformatics and Ontologies Project, and is funded by the

Gene Ontology Consortium. It has been optimized for OBO ontology language. It features a tree-based ontology editing interface, a graph visualizer and search capabilities.

- The Protégé [12] system is an environment for ontology development capable of running on multiple platforms. It supports customized user-interface extensions through plug-ins. Protégé supports Semantic Web technologies like OWL to build ontologies that can be made accessible to the Web. Protégé implements a rich set of knowledge-modeling structures that support visualization and editing of ontologies.

Ontology Matching or Alignment

Ontology matching or ontology alignment is the process of determining correspondences between concepts across ontologies [43, 44]. Historically, the need for ontology alignment rose out of the need to integrate independently developed heterogeneous databases [45, 46]. With the advancement in ontology technology in the form of the Semantic Web and OBO, and growing ontology content, aligning ontologies is key to interoperability among heterogeneous resources.

2.2 MORPHOLOGY-BASED PHYLOGENETIC STUDIES

Definitions

- **Taxon (pl. taxa):** A taxon may be a single organism or a group of taxa that is considered a unit by a systematic biologist. A taxon may or may not be named. Usually, it is a group of organisms that are inferred to be phylogenetically related and have characteristics in common.

	Has vertebrae?	Has bony skeleton?	Has four limbs?	Produces amniotic egg?	Has hair?	Has two post-orbital fenestrae?
Sharks & relatives	Yes	No	No	No	No	No
Ray-finned fishes	Yes	Yes	No	No	No	No
Amphibians	Yes	Yes	Yes	No	No	No
Primates	Yes	Yes	Yes	Yes	Yes	No
Rodents & rabbits	Yes	Yes	Yes	Yes	Yes	No
Crocodiles & relatives	Yes	Yes	Yes	Yes	No	Yes
Dinosaurs & birds	Yes	Yes	Yes	Yes	No	Yes

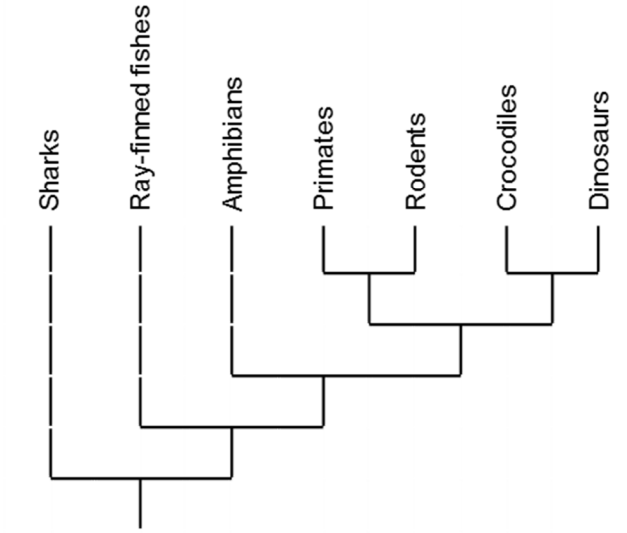


Figure 2.3: Examples of a data matrix based on morphological characters, and a phylogenetic tree obtained from that matrix. Example data courtesy evolution.berkeley.edu

- **Model Organism:** A model organism is one that is extensively studied and considered a representative for a class of organisms. For example, mouse and zebrafish are model vertebrates, fruit fly is a model invertebrate, rice is a

model plant etc. Newly discovered organisms are often studied based on a comparison with a model organism.

- **Homology:** Homology refers to a similarity among characteristics of taxa that is due to their shared ancestry. For instance, anatomical structures that perform the same function in different taxa and evolved from the same structure in some ancestor taxon are homologous. Phylogenetic studies often focus on identifying homologies among taxa in order to estimate better evolutionary relationships.
- **Convergent Evolution and Homoplasy:** Convergent evolution refers to the acquisition of similar biological structures in unrelated lineages (in contrast to homology, which has a common origin). Similarity in structures that evolve through convergent evolution is called homoplasy.
- **Character and Character State:** A character is an observable trait or feature that may be of interest to a systematic biologist. A character state is a specific value taken by a character in a specific taxon. For example, a character 'skin color' may have states 'black' and 'brown' etc. Systematic biologists often carefully choose characters that are potential indicators of homologies among the taxa under study, in order to build more accurate evolutionary lineages.
- **Data Matrix:** A data matrix presents character state assignments to all the taxa in a comparative study (see example in Figure 2.3). Each row usually represents a taxon, and each column represents a character. A cell in the matrix specifies the state assignment for a specific character to a specific taxon. The row for each taxon represents its state vector. The matrix is also called a morphological character matrix or simply character matrix.

- **Phylogenetic Tree:** A phylogenetic tree (or simply phylogeny, tree) represents evolutionary relationships among the group of taxa contained in the tree. Terminal nodes of such a tree usually represent extant organisms, whereas internal nodes represent ancestral taxa. The ancestral nodes in the phylogeny may be hypothetical. Figure 2.3 shows an example of a phylogenetic tree.
- **Tree Estimation:** Phylogenetic trees represent some form of hierarchical clustering over the character states of taxa in a data matrix. Various techniques for clustering the character states have been developed for estimation of phylogenetic trees. Common approaches are based on maximum parsimony, maximum likelihood and Bayesian methods [14, 15], which are applied to the state vectors in the data matrix.
- **Internally Labeled Phylogeny (ILP):** Even though the internal nodes of a tree may be hypothetical, some tree algorithms also assign taxa to internal (ancestral) nodes by inferring their state vectors when estimating the tree [47]. A phylogenetic tree which has all terminal and ancestral nodes labeled with state vectors is called an internally labeled phylogeny (ILP) [48].

Kinds of Phylogenetic Studies

There are two kinds of studies in morphology-based phylogenetics:

- **Single taxon studies:** As the name suggests, these studies focus on an individual taxon. Such studies produce annotated imagery as well as *Nomina Anatomica* that explains the anatomy of the taxon. Often, the *Nomina Anatomica* of a model organism is used to guide the development of a new *Nomina Anatomica*.

- **Multi-taxon studies:** These studies compare different taxa with the help of images as well as characters and character states. These character states are used to populate data matrices that may be input to a phylogenetic tree creation algorithm.

2.3 IMAGES IN LEARNING AND SCIENCE

Images have been an important instrument for human development and learning for a long time.

Use of representational pictures is supported by the research and theory on the potency of visual memory and the importance of providing examples when teaching concepts [49].

> *In 1994, elaborate cave art was discovered in Chauvet Cave, in France, art that is thought to be 35,000 years old. Far from being primitive, these animal paintings, engravings, and drawings were skillfully executed. As this find illustrates, from very early on people have created pictures. Perhaps these early paintings served as adjunct aids to storytellers, playing a role in humankind's development. Similarly, illustrations have been a part of our more recent development via the picture storybooks of our childhoods. [50]*

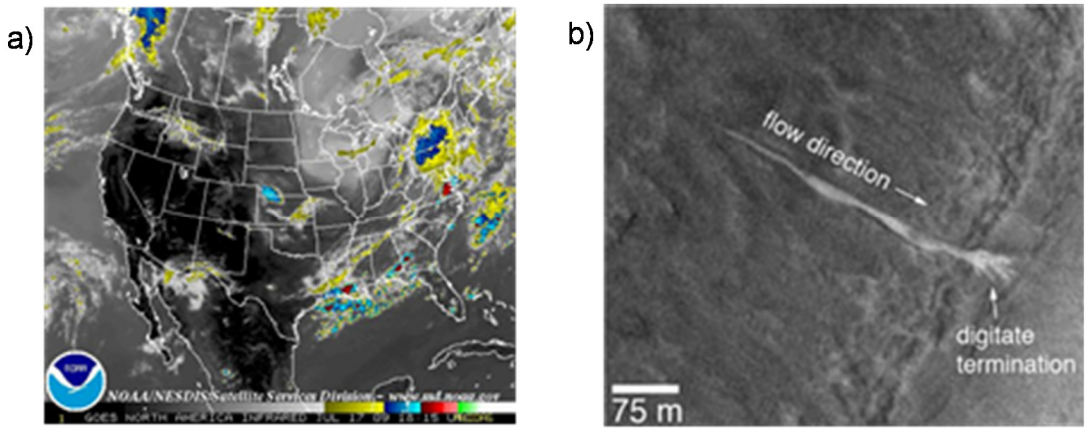


Figure 2.4: Kinds of images used in natural sciences such as meteorology and astronomy. The image on the left is an infrared satellite image taken by National Weather Service for weather forecast. The image on the right is a NASA photograph from its Mars mission that suggests presence of water on Mars.

In scientific studies, images are often treated as a definitive basis for a concept or an observation. In particular, such use of images is anticipated in natural sciences where the documentation of scientific observations is becoming increasingly reliant on digital imagery. For instance, astronomers rarely observe very large objects directly through telescopes, preferring imagery from sensitive electronic sensors than relying on the human eye [51]. Such use of image is common in other sciences like systematic biology [52, 53, 54], radiology [55], and Geographical Information Systems [56] etc. as well.

Figure 2.4 shows examples of images used in meteorology and astronomy. Meteorologists often look at images (like Figure 2.4a) to understand and predict weather patterns. Similarly, Figure 2.4b is an actual image from NASA's Mars mission [51], with labels that show how these images are used to build hypothesis such as: "Is there water on Mars?"

Chapter 3

Automatic Creation of New Anatomy Ontologies

Building *Nomina Anatomica* or anatomy ontology for a taxon is important to many AToL projects. Usually, this is a manual process that involves both subject matter experts and knowledge engineers. Given the scale of AToL, this is not a feasible methodology for building anatomy ontologies. We anticipated that the use of knowledge from model organisms and existing comparative phylogenetic studies may provide a way of automating the process of creating new anatomy ontologies.

Automating the creation of an anatomy ontology is a primary contribution of this dissertation. We have identified that they key problem to solve is to understand morphological character statements and representing them appropriately for knowledge inference. In this regard, we have created a taxonomy of character types and defined ontological signatures (or frames) for each type that allows us to precisely capture a natural language character statement in a logical form. We have developed an algorithm that uses the anatomy ontology of a designated model organism, a morphological character matrix and a phylogenetic tree to generate anatomy ontologies for the other taxa in the tree. In this chapter we present our work on this problem.

3.1 UNDERSTANDING CHARACTERS AND CHARACTER STATES

The first step towards solving this problem is based on clearly understanding and unambiguously specifying different kinds and classes of morphological characters and their states. Paul Sereno's work on logical basis for characters [21] and the EQ model of character matrices by Phenoscape group [57] are the two preceding efforts.

Character statement: Maxilla, anterior process, length relative to posterior process: shorter (0); longer (1)

Logical components: L_2 = maxilla, L_1 = anterior process, V = length, q = relative to the posterior process, v_0 = shorter, v_1 = longer

Figure 3.1: An example of a character statement and its logical components.

Sereno described morphological characters as statements with logical patterns [21]. Per Sereno, a character is an independent variable and character states are mutually exclusive conditions of a character. Each character statement is composed of up to four kinds of logical components: *locators* L_n (representing morphological structures), *variable V*, *variable qualifier q*, and *character states* v_n (mutually exclusive values of the variable). An example of a character statement and its logical pattern is shown in Figure 3.1.

Maxilla, anterior process, presence: absent (0); present (1)	**Neomorphic**
Maxilla, maxillary fenestra, shape: circular (0); oval (1)	**Transformational**
Calyx, circumference: 1-2 cm (0); 3-5 cm (1)	**Transformational**

Figure 3.2: Kinds of character statements as identified by Sereno [21].

Sereno also identified two fundamental kinds of characters: *neomorphic* and *transformational*. A neomorphic character is about de novo appearance of a

morphological structure or its loss without trace. A transformational character, on the other hand, is about a transformation of a morphological structure from one state to another comparable state. Examples of each kind of character are given in Figure 3.2.

Maxilla, anterior process, presence: absent (0); present (1)
EQ statement 1: E_1 = maxilla, E_2 = anterior process, Q = absent
EQ statement 2: E_1 = maxilla, E_2 = anterior process, Q = present

Maxilla, maxillary fenestra, shape: circular (0); oval (1)
EQ statement 1: E_1 = maxilla, E_2 = maxillary fenestra, Q = circular
EQ statement 2: E_1 = maxilla, E_2 = maxillary fenestra, Q = oval

Figure 3.3: Examples of EQ statements for some characters.

The EQ model for character matrices [57] describes character states (or *phenotypes*) by identifying the morphological entities (E) and qualities or adjectives (Q) involved in a character statement. Under the EQ model each character state is translated into an EQ statement, composed of Es and Qs. Since a character statement lists multiple possible character states, each character statement produces multiple EQ statements.

Compared to Sereno's logical components, the EQ model has a simpler pattern for character states: only E_n and Q_n. However, the inability to capture the missing components, i.e. the character (or variable) and qualifier, is a shortcoming that makes the EQ statements of more complex characters (see Figure 3.1), incomplete and therefore ambiguous. In other words, the EQ statements produced by two different characters will be the same if they involve the same entities and qualities, irrespective of any qualifier on the variable involved.

A strength of the EQ model is that it supports connections to ontologies to disambiguate the given entities and qualities. The entities (Es) in an EQ statement come from the anatomy ontologies of the relevant taxa. The qualities (Qs) are obtained from a

standard ontology of *phenotypic qualities* (adjectives used in morphological characters and character states) called Phenotypic Quality Ontology (PATO) [58].

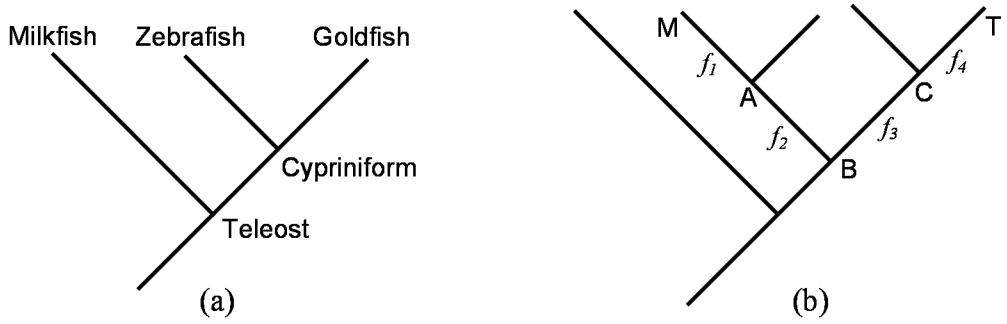


Figure 3.4: Phylogenetic trees depicting changes among taxa: (a) shows a (hypothetical) phylogenetic tree containing zebrafish (an NIH model organism) and other taxa; (b) shows the edges on a phylogeny as transformation functions f_i.

3.2 INFERENCE BASED ON PHYLOGENY TRAVERSAL

The fundamental idea behind our algorithm is as follows:

A *Nomina Anatomica* describes a particular taxon. A phylogenetic tree captures the evolutionary relationships among a group of taxa, and is developed based on the character state assignments for each taxon in the phylogeny. Each edge in the phylogeny represents some evolutionary change between the ancestor and descendent taxa, as evident from the difference between the corresponding state vectors. Starting from the existing anatomy ontology of a particular taxon (perhaps a model organism), applying the changes in state vectors as transformations to the ontology will produce the anatomy ontology for a different taxon.

Hence, starting from the anatomy ontology for some taxon (e.g. zebrafish ontology called ZFA [9]), and a phylogeny that contains that taxon (Figure 3.4a), we can produce anatomy ontologies for all the other terminal or ancestral nodes in the tree (e.g.

goldfish, teleost etc.) by traversing the tree and applying appropriate transforms to the ontology at each step.

Let:

```
OM          = ontology for M
OT          = ontology for T
Path        = A queue of steps between M to T in the tree.
N           = No. of characters; or length of each character state vector.
Characters  = The vector of characters.
States(X)   = The character state vector of taxon X.
```

Algorithm:

```
O := OM

While Path not empty:
  Pop Step <X, Y> from Path

  Comment: O is the ontology for X in this step

  For each n from 1 to N:
    If States(X)[n] <> States(Y)[n]:
      Apply transformation for Character[n] on O

  Comment: O is now the ontology for Y in this step

OT := O
```

Figure 3.5: Algorithm for transformation the ontology for source taxon M to the ontology for target taxon T.

More formally, given the phylogeny shown in Figure 3.4b, if M is the taxon with an existing ontology O(M), and T is the target taxon for the new ontology O(T), we can obtain O(T) in a step-wise fashion by applying transformations f_i at each tree edge as follows:

$$O(A) = f_1\big(O(M)\big)$$

$$O(B) = f_2(O(A)) = f_2\left(f_1(O(M))\right)$$
$$O(C) = f_3(O(B)) = f_3\left(f_2\left(f_1(O(M))\right)\right)$$
$$O(T) = f_4(O(C)) = f_4\left(f_3\left(f_2\left(f_1(O(M))\right)\right)\right)$$

An outline of our algorithm is presented in Figure 3.5.

From Zebrafish to Milkfish Ontology

Consider the example presented in Figure 3.6. A phylogenetic tree contains terminal taxa *zebrafish* and *milkfish*, and an ancestral taxon *node A*. We are also given the anatomy ontology for zebrafish, and a set of character statements. The character state assignments for each terminal and ancestral taxon are also provided as vectors (s_1, s_2) in the tree, where s_1 is the state for character 1 and s_2 is the state for character 2. The goal is to obtain the ontology for milkfish.

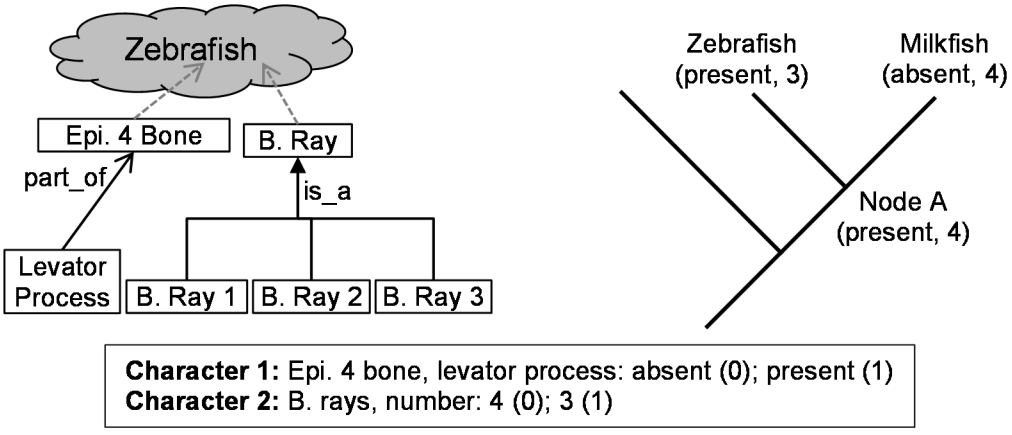


Figure 3.6: Sample input to the ontology generation algorithm: zebrafish anatomy (ZFA) ontology, a set of character statements, and a synthetic phylogenetic tree containing zebrafish and other taxa along with their character state assignments.

Since the path from zebrafish to milkfish in the phylogenetic tree contains two steps (zebrafish to node A, node A to milkfish), two sets of transformations need to be

applied to the zebrafish ontology, based on the differences in the character state vectors of the taxa involved, to obtain the milkfish ontology (Figure 3.7):

- In the first step, the zebrafish ontology is transformed to the node A ontology. According to the character state vectors, the difference between the two taxa is based on the state of character 2 changing from 3 to 4. In other words, the number of *b. rays* has increased from 3 to 4. Adding another b. ray, i.e. *b. ray 4*, to the ontology gives the ontology for node A.
- In the second step, the node A ontology is transformed to the milkfish ontology. Based on the change of state of character 1 from present to absent, the *levator process* in the *epi. 4 bone* is no longer present. Removing the *levator process* and its relationship with *epi. 4 bone* produces the milkfish ontology.

Note that in this example, each step involved changing the character state for only a single character. In reality, successive steps may involve a large number of character state changes. Therefore at each step, a set of transformations may need to take place to get the new ontology, each individual transformation based on understanding the meaning of the corresponding character statement.

3.3 A SOLUTION FOR ONTOLOGY GENERATION

The solution to this problem is based on treating the components of phylogenetic studies, in particular the character statements and phylogenetic trees, as a single ontology based on our meta-model (see Figure 1.3). The following sub-problems were solved in order to develop the algorithm for generating new anatomy ontologies:

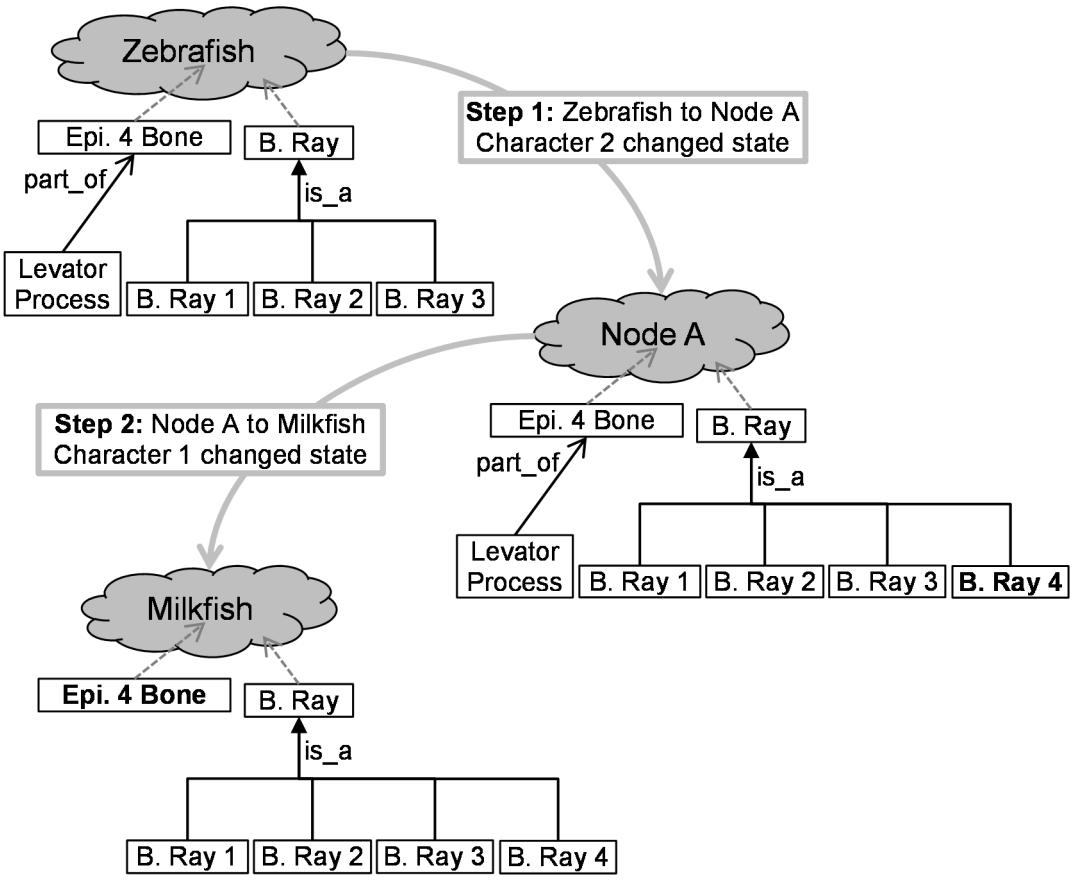


Figure 3.7: Stepwise generation of ontologies to get the target ontology. Transformations, based on character state changes, take place at each step.

1. **Phylogenetic Tree as Ontology:** As described in the earlier example, our algorithm is based on traversing a phylogenetic tree. We represent the tree in an ontological form in order to support connections between the tree and the character data. It also allows us to provide logical definitions of some important concepts in phylogenetics.
2. **Characters in Ontology:** Generating new anatomy ontologies requires a detailed treatment of character statements as ontological concepts. This is necessary in order to represent different kinds of characters in an

unambiguous form. We created a taxonomy of character types for this purpose. Each type of character in the taxonomy has a unique signature that enables its representation in an ontology.

3. **Transformations on Ontology:** A critical aspect of our solution is to perform appropriate transformations on the ontologies based on understanding the character. After distinguishing between different kinds of characters, we identified and implemented transformation rules for each kind. These transformations rules fire on changes in character state assignments. The effects of these rules include adding new concepts, removing existing concepts, or modifying existing concepts and/or their relationships.

In the following sections of this chapter, we provide detailed explanations of our work on these problems.

3.4 PHYLOGENETIC TREE AS ONTOLOGY

A phylogenetic tree represents evolutionary relationships among taxa, i.e., it tells us the ancestor of each taxon in the tree (except for the root). Our meta-model enables us to capture phylogenetic trees as part of the ontology simply by a single association "parent of" that links a taxon to its direct descendants (see Figure 1.3).

Figure 3.8 shows the ontological representation of a phylogenetic tree. Each rectangle in the figure connects to its corresponding matrix cells (and hence character states) and specimens per the meta-model. Our algorithm for automatically creation ontologies works by traversing the "parent of" relationships among the taxa.

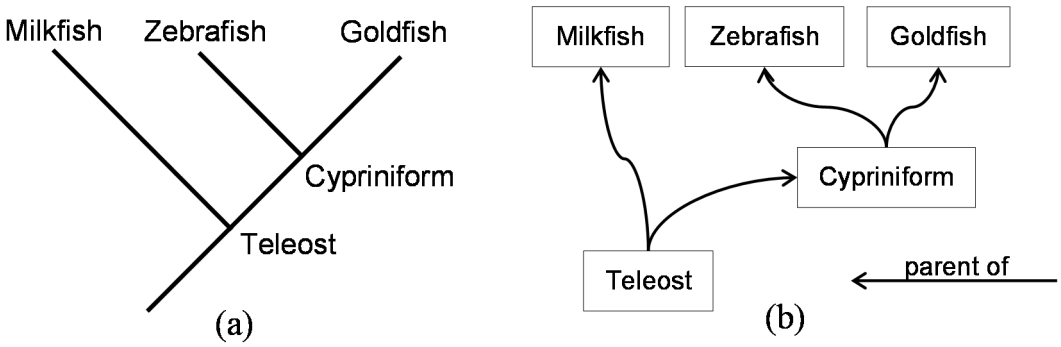


Figure 3.8: Representation of a phylogenetic tree (a) as a part of the ontology (b) where each rectangle represents a taxon, and each edge is "is ancestor of".

In addition, using this representation of the tree, formal definitions of some important concepts in phylogenetics can be specified. Dictionary definitions (taken from a commonly used source cited at the end of the definition), and formal definitions of such concepts are provided below. In these definitions, *T* is the set of all taxa in a phylogenetic tree of a phylogenetic study and *CSt* is the set of all character states in the given character matrix. For first order logic representation, let $IsParentOf(A, B)$ be reflexive and mean that taxon *A* is "parent of" taxon *B*. Also, since each taxon connects to its character states, let $HasState(A, S)$ mean that taxon *A* exhibits character state *S*.

- **Homology:** A character state *cs* in different taxa, say *t* and *u*, is considered a homology if it is similar because it was inherited from a common ancestor that also had that feature [59].

$$Homology(cs, t, u) \equiv cs \in CSt \wedge t, u \in T$$
$$\wedge HasState(t, cs) \wedge HasState(u, cs) \wedge \exists a \in T$$
$$\big(IsParentOf(a, t) \wedge IsParentOf(a, u) \wedge HasState(a, cs)\big)$$

- **Homoplasy:** A character state *cs* in different taxa *t* and *u* that has separate evolutionary origins, but is superficially similar because it evolved to serve the same function is called homoplasy or analogy. These are the result of convergent evolution [59].

$$Homoplasy(cs,t,u) \equiv cs \in CSt \wedge t,u \in T$$
$$\wedge\, HasState(t,cs) \wedge HasState(u,cs) \wedge \neg Homology(cs,t,u)$$

- **Plesiomorphy:** Same as primitive trait, i.e. a feature *cs* that is present in the common ancestor of a group (or *clade*) *g* [60].

$$Plesiomorphy(cs,g) \equiv cs \in CSt \wedge g \subseteq T \wedge HasState(mra(g),cs)$$

- **Symplesiomorphy:** The possession of a character state *cs* that is primitive (plesiomorphic) and shared between two or more taxa [61].

$$Symplesiomorphy(cs,g) \equiv Plesiomorphy(cs,g)$$
$$\wedge\, \exists t,u \in g\big(t \neq u \wedge HasState(t,cs) \wedge HasState(u,cs)\big)$$

- **Apomorphy:** An evolutionary trait *cs* that is unique to a particular taxon *t* and its descendants and which can be used as a defining character for a species or group in phylogenetic terms [62].

$$Apormophy(cs,t) \equiv cs \in CSt \wedge t \in T$$
$$\wedge\, HasState(t,cs) \wedge \forall u \in T(HasState(u,cs) \rightarrow IsParentOf(t,u))$$

- **Synapomorphy:** An apomorphy that is shared by two or more taxa is termed a synapomorphy [62].

$$Synapomorphy(cs,t) \equiv Apomorphy(cs,t) \wedge \exists u,v \in T$$
$$\begin{pmatrix} u \neq v \wedge IsParentOf(t,u) \wedge IsParentOf(t,v) \\ \wedge\, HasState(u,cs) \wedge HasState(v,cs) \end{pmatrix}$$

3.5 CHARACTERS IN ONTOLOGY

Here we provide a detailed treatment of character statements, i.e. characters and character states, as ontological concepts. We have based our work on Paul Sereno's logical basis for characters [21] and the EQ model for character states [57] discussed earlier.

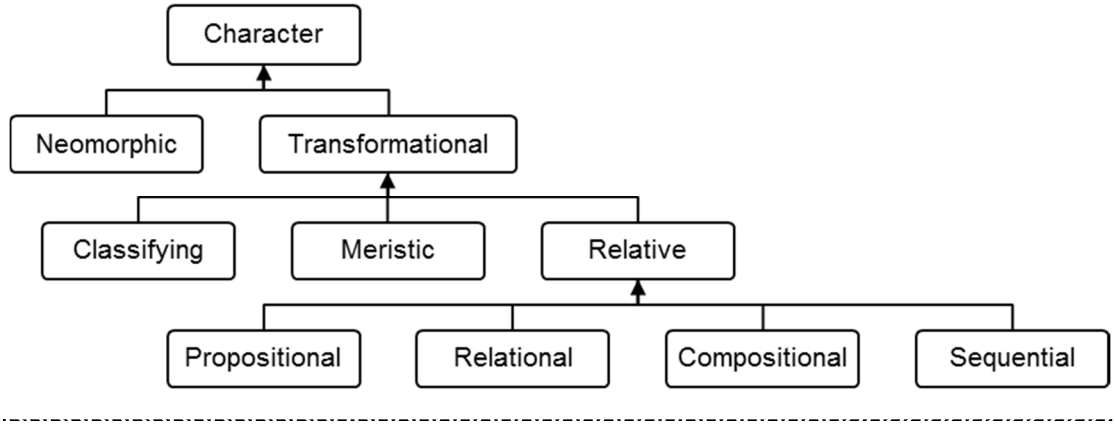


Figure 3.9: A taxonomy of character types.

A Taxonomy of Character Types

There are two fundamental types of characters: *neomorphic* and *transformational*. Neomorphic characters are composed only of locators and refer to their absence or presence. These locators are anatomical entities, and if derived from an anatomy ontology, are the same as an entity (E) in the EQ model. Transformational character statements, on the other hand, may take various forms within the logical pattern described by Sereno (see Figure 3.1), depending upon the meaning of the character statement and the kind of the variable involved. We have identified and named some of these particular forms, and have organized them into a taxonomy of character types (Figure 3.9). This is

not an exhaustive set of character types. We have only considered the types that we have identified as relevant to anatomy ontology transformations.

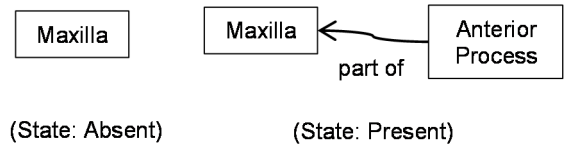
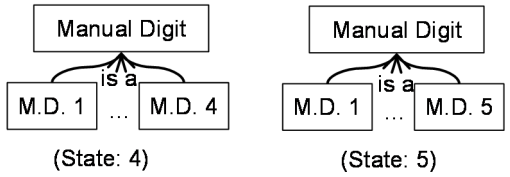
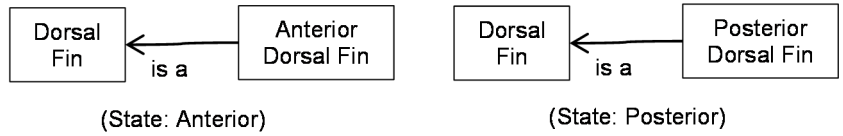
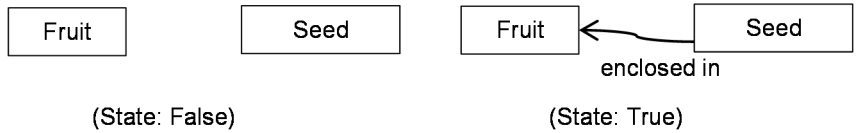
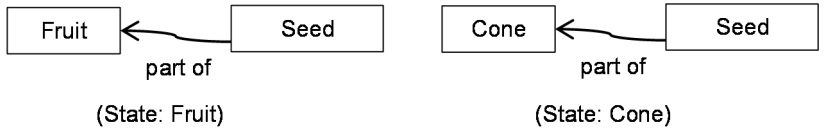
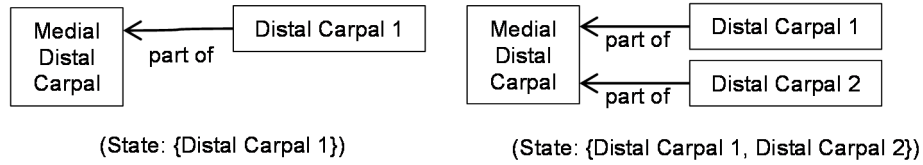
- **Neomorphic Character (NE):** A neomorphic character is about the absence or presence of an entity. An example of a neomorphic character is "Manual digit I: present (0); absent (1)", character statement 16 in [21]. A neomorphic character statement consists of a locator, which is an entity, and specific character states: absent and present.
- **Transformational Character (TR):** As mentioned earlier, a transformational character is about transformations of morphological structures between comparable states. All the remaining types of characters in our taxonomy are transformational.
- **Classifying Character (CL):** We define a classifying character as one that applies simple adjectives (qualities) to an entity. For example, character 51 in [63] is a classifying character: "Lateral eyes: no (0); simple (1); compound (2); stalked-compound (3)". More examples are character statements 7 and 8 in [21].
- **Meristic Character (ME):** A meristic character is about the count of a particular entity. For example, character 52 in [63] is a meristic character: "Median eyes: none (0), four (1), two (2)". Another example is character statement 15 in [21].
- **Relative Character (RV):** Relative character is a subcategory of transformational characters which deals with relationships among multiple entities. The remaining types of characters fall into this category.

- **Propositional Character (PR):** A propositional character concerns the existence of a specified relationship between two entities. There are only two possible character states: true or false, depending on whether the relationship exists or not. An example for this type of character may be: "Seed is enclosed in fruit: true (0), false (1)".
- **Relational Character (RE):** A relational character concerns the existence of a relationship of an entity with multiple alternative entities. The character states for such a character are entities. For example, "Location of seed: fruit (0), cone (1)" is a relational character.
- **Compositional Character (CO):** A compositional character concerns expressing the parts of a particular entity. The character states in this case are sets of entities. An example is given in Table 7 of [21]: "Medial distal carpal, composition: distal carpal 1 (0); distal carpals 1 + 2 (1)".
- **Sequential Character (SE):** A sequential character specifies the order of occurrence of entities, which may be helpful in understanding their organization. The character states for such a character are sequences of entities. For example, character 55 in [63] "Ordering of fate map tissues: anterior – stomodeum – midgut – mesoderm – posterior (0), anterior – midgut – mesoderm – stomodeum – posterior (1)" is a sequence character. In our research we have found that this type of character occurs rarely in phylogenetic studies. While we have included it in the taxonomy of character types, it does not appear further in this work.

Table 3.1: Forms and frames of character types.

Type	Grammar Form	Example	Frame/Signature
Neomorphic (NE)	E_1 [R] E_2, Q: absent (S_0); present (S_1)	Anterior process [part of] maxilla, presence: absent (0); present (1)	E_1 = anterior process R = part of E_2 = maxilla Q = presence S_0 = absent S_1 = present
Meristic (ME)	E Q: X_0 (S_0); X_1 (S_1) ... X_n (S_n)	Manual digits, number: 5 (0); 4 (1)	E = manual digit Q = number/count S_0/X_0 = 5 S_1/X_1 = 4
Classifying (CL)	E Q: Q_0 (S_0); Q_1 (S_1) ... Q_n (S_n)	Dorsal fin, location: anterior (0); posterior (1)	E = dorsal fin Q = location S_0/Q_0 = anterior S_1/Q_1 = posterior
Propositional (PR)	E_1 R E_2: true (S_0); false (S_1)	Seed enclosed in fruit: true (0); false (1)	E_1 = seed R = enclosed in E_2 = fruit S_0 = true S_1 = false
Relational (RE)	E R: E_0 (S_0); E_1 (S_1) ... E_n (S_n)	Seed, part of: fruit (0); cone (1)	E = seed R = part of S_0/E_0 = fruit S_1/E_1 = cone
Compositional (CO)	E Q: {E_1 ... E_i} (S_0); ... {E_j ... E_n} (S_n)	Medial distal carpal composition: distal carpal 1 (0); distal carpals 1 + 2 (1)	E = medial distal carpal Q = composition S_0 = {distal carpal 1} S_1 = {distal carpal 1, distal carpal 2}

Table 3.2: Examples of transformations based on character types.

Type	Rule (fires when state changes)	Illustration (difference between two possible states)
NE	*Absent to present:* add E_1 and its relationship R with E_2 *Present to absent:* delete E_1, and its relationship R with E_2	
ME	*S_0 to S_1:* If $X_0>X_1$, delete X_0-X_1 children of E If $X_1>X_0$, add X_1-X_0 children for E	
CL	*S_0 to S_1:* delete child of E with quality Q_0 and add a new child of E with quality Q_1	
PR	*False to true:* add relationship R between E_1 and E_2 *True to false:* delete relationship R between E_1 and E_2	
RE	*S_0 to S_1:* delete relationship R between E and E_0, and add relationship R between E and E_1	
CO	*S_0 to S_1:* delete all parts of E that are elements of S_0, and add all elements of S_1 are parts of E	

Signatures of Character Types

As mentioned earlier, each character type has its particular logical pattern, signature or *frame*. In order to fully capture a character statement as an ontological concept, it is necessary to identify its type and to populate its frame with the necessary elements.

In Table 3.1, we list the natural forms, examples that fit the corresponding forms and populated frames of the given character types. Our notation for forms and frames is based on a combination of Sereno's logic and EQ statements. Each E/E_i is an entity from an anatomy ontology, and is the same as a *locator* in Sereno's logic. Each Q/Q_i is a quality or adjective derived from PATO [58]. This includes the *variable* element in Sereno's logic, and the possible character states of some character types. Each R/R_i belongs to the official OBO Relations Ontology (RO) [27]. OBO Relations Ontology contains a set of relationship types for use across biomedical ontologies. We have restricted our scope for relationship types to this ontology.

Now that we have explained how to represent character statements in ontologies, we move on the transformation rules required to generate new ontologies.

3.6 TRANSFORMATIONS ON ONTOLOGY

When going from the *Nomina Anatomica* of one taxon to some other taxon, as explained earlier, we consider the differences between the character states of each taxon. Once a difference is identified, the next step is to make appropriate transformations on the source ontology based on the meaning of that particular change in the character state. The meaning is clear once the corresponding character statement is presented to the system in its frame representation. Based on the type of the character and the fields in the frame, we are able to make the necessary transformations on the source ontology to

produce the target ontology. These transformations are a result of firing appropriate transformation rules.

Table 3.2 shows informal descriptions of the rules for each character type, and also provides illustrations for changes in the ontology when the character state changes from one value to the other (based on the examples provided in Table 3.1). Each rule description in the table starts with an italicized label that states the change in the character state. Some character types have multiple kinds of changes in character states, and hence require multiple transformation rules.

Basic Definitions for Transformations

We have identified the transformation rules based on these six character types. We present these rules using some basic definitions provided in Table 3.3.

Table 3.3: Definitions of predicates and functions involved in formal rules.

Predicate/Function	**Definition**
Predicate	
$CType(c,t)$	Type of character *c* is *t*, where $t \in \{NE, ME, CL, PR, RE, CO\}$
$Change(c,s,t)$	Character *c* changed state from *s* to *t*
Function	
$mra(g)$	Returns the most recent ancestor of the set *g* of taxa

Each transformation rule has two parts, separated by a double headed arrow:

$$< Conditions > \twoheadrightarrow < Actions >$$

i.e., if a set of conditions is satisfied, perform the given set of actions.

A transformation rule can perform two kinds of actions: *ADD* and *RETRACT*. The *ADD* action adds an entity or a relationship between two entities to the knowledgebase, and the *RETRACT* action removes an entity or a relationship from the knowledgebase. *RETRACT* on an entity removes all the relationships between that entity and other

entities. Some notational conventions regarding the use of actions with these objects are presented in Table 3.4.

Table 3.4: Notational conventions regarding the use of actions and objects.

Expression	Explanation
$E \underset{R}{\rightarrow} F$	A relationship of type *R* between entities *E* and *F*.
$ADD(E)$	Add entity *E*.
$ADD\left(E \underset{R}{\rightarrow} F\right)$	Add relationship of type *R* between *E* and *F*.
$RETRACT(n, x)$	If condition *n* holds, *RETRACT x*.
$RETRACT\left(E \underset{R}{\rightarrow} F, G\right)$	Remove *G* if there is a relationship of type *R* between *E* and *F*.

Transformation Rules

Here is the listing of the transformation rules in our system. We discuss the implementation of these rules in the next section. Each rule assumes the existence of a character *c*:

- **Rule NE-1:** This rule is activated when a neomorphic character changes state from *absent* to *present*.

  $CType(c, NE) \wedge Change(c, absent, present) \twoheadrightarrow$
  $ADD(c.E_1)\ , ADD\left(c.E_1 \underset{c.R}{\rightarrow} c.E_2\right)$

- **Rule NE-2:** This rule is activated when a neomorphic character changes state from *present* to *absent*.

  $CType(c, NE) \wedge Change(c, present, absent) \twoheadrightarrow RETRACT(true, c.E_1)$

- **Rule ME-1:** This rule is activated when a meristic character changes state from *X* to *Y*, *0<X<Y*.

$$CType(c, ME) \wedge Change(c, X, Y) \wedge X < Y \twoheadrightarrow$$
$$\forall n, (X < n \leq Y) \left(ADD(e_n) , ADD\left(e_n \underset{is\ a}{\longrightarrow} c.E\right)\right)$$

- **Rule ME-2:** This rule is activated when a meristic character changes state from *X* to *Y*, *0<Y<X*.

$$CType(c, ME) \wedge Change(c, X, Y) \wedge Y < X \twoheadrightarrow$$
$$\forall n, (Y < n \leq X) \left(RETRACT\left(e_n \underset{is\ a}{\longrightarrow} c.E, e_n\right)\right)$$

- **Rule CL-1:** This rule is activated when a classifying character changes state from *V* to *W*.

$$CType(c, CL) \wedge Change(c, V, W) \twoheadrightarrow$$
$$RETRACT\left(e_V \underset{is\ a}{\longrightarrow} c.E, e_V\right), ADD(e_W), ADD\left(e_W \underset{is\ a}{\longrightarrow} c.E\right)$$

- **Rule PR-1:** This rule is activated when a propositional character changes state from *false* to *true*.

$$CType(c, PR) \wedge Change(c, false, true) \twoheadrightarrow ADD\left(c.E_1 \underset{c.R}{\longrightarrow} c.E_2\right)$$

- **Rule PR-2:** This rule is activated when a propositional character changes state from *true* to *false*.

$$CType(c, PR) \wedge Change(c, true, false) \twoheadrightarrow RETRACT\left(true, c.E_1 \underset{c.R}{\longrightarrow} c.E_2\right)$$

- **Rule RE-1:** This rule is activated when a relational character changes state from *G* to *H*.

$$CType(c, RE) \wedge Change(c, G, H) \twoheadrightarrow$$
$$RETRACT\left(true, c.E \underset{c.R}{\longrightarrow} G\right), ADD\left(c.E \underset{c.R}{\longrightarrow} H\right)$$

- **Rule CO-1:** This rule is activated when a compositional character changes state from *Gs* to *Hs*.

$$CType(c, CO) \wedge Change(c, Gs, Hs) \twoheadrightarrow$$
$$\forall G \in Gs \left(RETRACT \left(true, \mathrm{G} \xrightarrow[part\ of]{} c.E \right) \right),$$
$$\forall H \in Hs \left(ADD \left(H \xrightarrow[part\ of]{} c.E \right) \right)$$

In some cases, the naming/identifier conventions for the objects are not very obvious in the formal representation. These can be clarified using the examples given in Table 3.2 or by looking at the implemented source code listing provided as Appendix A.

3.7 IMPLEMENTATION

The transformation rules are implemented in Jess scripting language run by the Jess Rule Engine [64] for the Java Platform. For the transformation rules listed in formal notation earlier, we have a total of 16 rules encoded in Jess. The main body of the algorithm and the data model it operates upon is implemented as a Java program.

```
(defrule rule-03-effect-of-neomorphic-character
   "State change from present to absent in a neomorphic character"
   (Params (target ?tgt) (baseUri ?baseuri))
   (CharNE (id ?chid) (entity ?nee))
   (StateChange (character ?chid) (fromState ?fsid) (toState ?tsid))
   (State (id ?fsid) (name "present"))
   (State (id ?tsid) (name "absent"))
   ?theentity <- (NAEntity (id ?eid))
   (test (eq (str-cat ?baseuri ?tgt "/" ?nee) ?eid))
=>
   (retract ?theentity)
   (printout t "*** NE :: (03) - [E:" ?nee "] ==> " ?chid crlf))
```

Figure 3.10: A transformation rule (for neomorphic character state change) in its Jess script form.

A sample Jess transformation rule is shown in Figure 3.10. Each rule has two parts: the conditions are listed in the first part, and actions in the second. The two parts are separated by the '=>' symbol. As a convention we have implemented the last action for each rule to be an explanation generation instruction for the actions performed by the rule.

Our program outputs all the ontologies generated on the path from the source taxon to the target taxon. These ontologies are exported to our Ontobrowser tool for web based browsing. The target taxon ontology is also generated in OWL format. In addition, the program produces an explanation for each transformation at each step in the algorithm. A sample explanation is shown in Figure 3.11.

```
From N10 to N02

* ME :: (13) + [E:branchiostegal-ray#4]
  [R:branchiostegal-ray#4 is_a branchiostegal-ray] ==> C54-

* NE :: (03) - [E:epibranchial-4-bone-uncinate-process] ==> C36-

* CL :: (06) + [E:straight-epibranchial-bone]
  [R:straight-epibranchial-bone is_a epibranchial-bone] ==> C27-
```

Figure 3.11: A sample explanation output.

Each line of the explanation gives the type of the character involved, a rule number that caused the change, addition or retraction of appropriate entities and/or relationships, and finally the identifier of the character that caused that particular transformation.

We provide the source listing for our Jess rules as Appendix A.

In order to evaluate the results produced by our algorithm, i.e. the new ontologies generated, we have examined the following test cases.

3.8 TEST CASE: FROM ANGIOSPERMS TO OTHER PLANTS

This test case is based on the Plant Structure Ontology (PSO) [65] for flowering plants (*angiosperms*). This is a synthetic test case developed for the specific purpose of testing and demonstrating the algorithm on a small amount of data. We created a phylogenetic tree of 9 (terminal and ancestral) taxa based on the data available on PLANTS Classification Report [66, 67]. The tree is rooted at *embryophyte* (or land plants). An intermediate ancestral taxon in the tree is angiosperm, which serves as the source (or model) taxon for the test case.

We created a character matrix comprising 10 morphological character statements of various kinds. We also labeled the taxa with their corresponding character state vectors. Also, we extracted a portion of PSO (36 entities, 36 relationships) into a small ontology that is suitable for our character statements. We also populated the frames for our character statements with the appropriate elements from PSO and other ontologies, PATO and RO.

Table 3.5: Ontologies generated by the plants test case.
The test case starts from a portion of Plant Structure Ontology for angiosperms containing 36 entities and 36 relationships among entities. The Added/Retracted columns list the number of terms or relationships added or removed from the angiosperm ontology to get the new ontology.

Ontology (Taxon)	**Terms (Entities)**			**Relationships (Triples)**		
	Count	**Added**	**Retracted**	**Count**	**Added**	**Retracted**
Embryophyte	34	2	4	36	2	2
Fern	35	3	4	37	3	2
Spermatophyte	35	2	3	36	2	2
Gymnosperm	36	3	3	37	3	2
Conifer	38	5	3	40	6	2
Cycad	36	3	3	37	3	2
Eudicot	39	4	1	39	4	1
Monocot	35	0	1	35	0	1

We generated ontologies for all the other taxa in the tree. Table 3.5 lists the ontologies created, the number of entities and relationships in the ontology for each taxon, and the number of additions and retractions involved in producing the ontology, starting from the ontology for angiosperms.

The raw input data for this test case is attached as Appendix B.

3.9 TEST CASE: CYPRINIFORMES TREE OF LIFE

This test case is based on the phylogenetic data provided by the biologists working on the Cypriniformes Tree of Life (CTOL) project [68]. This dataset corresponds to a study on the diversity of a particular anatomical region, i.e. gill arch and hyoid arch, across *cypriniformes*. The key taxa to remember for this test case are *teleosts*, *cypriniformes* and zebrafish (*Danio rerio*). *Cypriniformes* form a group of fish species that are studied under the CTOL project. Zebrafish is an NIH designated model organism, and is a species in the *cypriniformes* group. Teleosts are a broader group of fishes that includes *cypriniformes*. The CTOL project does not cover all teleosts.

The data from CTOL consists of two parts. First, we have a morphological character matrix containing 62 character statements and the character state assignments for 65 terminal taxa, including zebrafish. From this matrix, we have identified 39 character statements that fall under our taxonomy of character types, i.e., are relevant to anatomy ontology. We have only used these 39 characters in our tests (see our CTOL character matrix worksheet, Appendix C). Second, based on the character matrix, the biologists have produced a phylogenetic tree with ancestral character state assignments. The tree contains a total of 85 (terminal and ancestral) taxa. The tree contains zebrafish, cypriniformes and an ancestor of cypriniformes (root of the tree, henceforth called CTOL-root taxon), but not teleosts.

In addition to the phylogenetic data (character matrix and phylogenetic tree), we have used two authoritative ontologies in this test case: the Zebrafish Anatomy ontology (ZFA) [9], and the Teleost Anatomy Ontology (TAO) [10]. In addition to the anatomy of teleosts, TAO also provides a list of synonym cross-references between ZFA and TAO. While ZFA and TAO are large ontologies, we have extracted portions from each that correspond to the anatomical region of gill and hyoid arches.

Based on this data, we perform the following tests. First, using the teleost ontology (TAO) as the starting point, we generate a new zebrafish ontology (ZFA*) and compare it with the authoritative zebrafish ontology (ZFA). Second, we produce the ontology for *cypriniformes* gill and hyoid arches anatomy (CGO). We use TAO as the starting point to create CGO, and then use ZFA to create another version of CGO and compare the two versions. We have performed the comparisons between ontologies in a semi-automatic manner: we have used AgreementMaker ontology matching tool [69] to create some initial mappings between the ontologies being compared, and have performed the remaining work manually.

From Teleost to Zebrafish

This test is based on using the Teleost Anatomy Ontology (TAO) as a starting point for building a zebrafish ontology (ZFA*) for the relevant anatomical region. Since the given phylogenetic tree does not include the teleost node, as a first step we use the data in the morphological character matrix to manually modify TAO and create the ontology for the CTOL-root taxon. This modified TAO is used in the automated transformations in our algorithm. It contains 129 entities and 171 relationships among entities.

Since we have identified the types of relevant characters, in the second step we populate the frames for each character statement using the modified TAO, Phenotypic Quality Ontology (PATO) and OBO Relations Ontology (RO), as described earlier.

Once we have the source ontology and the populated character frames, we run the algorithm on the phylogenetic tree, going from CTOL-root to zebrafish, and produce a zebrafish ontology, called ZFA*. This ontology contains 112 entities and 157 relationships among entities. The portion of the original ZFA that we have extracted for comparison contains 103 entities and 142 relationships among entities.

Table 3.6: Synonym cross-references from TAO to ZFA and ZFA*.

Cross-References		**Difference**		
TAO-ZFA*	**TAO-ZFA**	**Overall**	**Design**	**Entities**
107	67	40	27	13

Table 3.7: Summary of newly discovered entities for ZFA.

Reason	**Count**
Difference in cross-referenced entities (from Table 3.6)	13
New entities without sufficient evidence	7
New entities with evidence	6
Other evident new entities not cross-referenced from TAO	4
Total number of evident new entities discovered	10

Synonym cross-references: As mentioned earlier, TAO provides synonym cross-references to the authoritative ZFA. Our system also keeps track of synonyms between source and target ontologies during transformations. We have compared our synonyms to the synonym cross-references provided by the authoritative TAO. This objective of this evaluation is to find any discrepancies between the two synonym lists and to pinpoint the reasons behind them. Table 3.6 shows a summary of our evaluation.

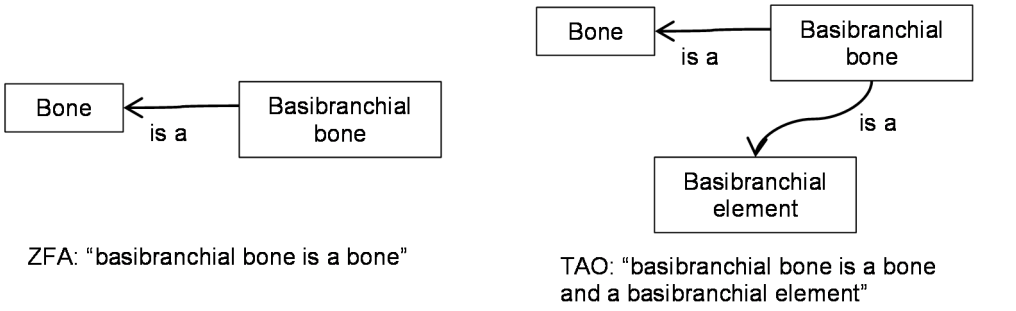


Figure 3.12: Difference in modeling approaches between ZFA and TAO. In terms of biology, there is little difference between the structures since the *basibranchial* part in ZFA already signifies a *basibranchial-element*. TAO has a number of *-element* entities, none of which are modeled in ZFA.

Table 3.8: New entities for ZFA.

No.	Entity (Term)	Evidence (Character)
1.	Hypobranchial 3 element ventral process	Neomorphic (C58A)
2.	Epibranchial 4 bone uncinate process	Neomorphic (C36-)
3.	Epibranchial 4 bone levator process	Neomorphic (C39-)
4.	Urohyal ventral plate	Classifying (C61A)
5.	Fully developed urohyal ventral plate	Classifying (C61B)
6.	Rod-like basibranchial 2 bone	Classifying (C09-)
7.	Spathiform branchiostegal ray	Classifying (C55-)
8.	Straight epibranchial bone	Classifying (C27-)
9.	Hooked-with-wear-surface ceratobranchial 5 tooth	Classifying (C25-)
10.	Tapered-tipped gill raker	Classifying (C47-)

The newly created ZFA* contains 107 cross-references to entities in TAO. Only 67 of those entities are cross-referenced between TAO and ZFA. The additional 40 cross-references between TAO and ZFA* are among the entities that do not exist in ZFA. Our investigation shows that 27 of these entities exist due to a difference of modeling approaches between TAO and ZFA. Since ZFA* is built from TAO, it inherits its

modeling approach and also contains these entities. However, these entities do not represent significant additions to the knowledgebase, as elaborated in Figure 3.12. Of the remaining 13 new cross-references between ZFA* and TAO, 6 are due to the discovery of new candidate entities for ZFA based on the evidence from the given character statements. The other 7 are due to the presence of entities in TAO that are not referenced in the given set of character statements, hence making it impossible for us to decide whether to delete them from ZFA* during the transformation process or to treat them as candidates for ZFA (Table 3.7).

Newly discovered knowledge: We have also examined the transformations at each step during the process of the creation of ZFA*. As the other criteria for the evaluation of our work, we have examined the retracted entities to identify any possible errors made during the creation of ZFA or the given morphological character matrix, and the added entities to identify any candidates for addition into the authoritative ZFA. We have found that all the retractions made by our transformations are consistent with the content of the authoritative ZFA. However, as explained in Table 3.7, we have identified 10 new candidate entities for addition into ZFA. These entities are listed in Table 3.8, along with the identifier for the character statement that serves as the evidence for its existence.

Cypriniformes Gill and Hyoid Arches Anatomy Ontology

Using our algorithm, we have generated the Cypriniformes Gill and Hyoid Arches Anatomy Ontology (CGO). As a new biological knowledgebase, and as the title suggests, this ontology captures the anatomical structure of the gill and hyoid arches of cypriniformes based on the knowledge captured in the CTOL character matrix. The ontology starts from the description of parts and subclasses of *pharyngeal arches*. *Gill*

arch and *hyoid arch* are pharyngeal arches. In addition to capturing the parts of these anatomical regions, the ontology also captures the composition of most anatomical entities involved, which seems to be a recurring feature of interest in the character matrix. Hence, entities are appropriately described as being *bones* or *cartilages*, and in some cases, *teeth*. CGO consists of 139 terms and 2 relationship types (`is_a` and `part_of`). A full listing of this ontology is provided as Appendix D.

Once again, our starting point for the new CGO ontology is the ontology for CTOL-root taxon obtained by manually updating the Teleost Anatomy Ontology using the character matrix. In order to double check the correctness of our ontology, we also started from the Zebrafish Anatomy ontology to produce another version of CGO. The purpose of this exercise is not to establish a formal equality of the two versions, but to examine the structure of the two ontologies to identify any discrepancies.

We manually examined the topology of the two ontologies and found them to be consistent. As an additional exercise, we generated the ontologies for all the taxa in the path from CTOL-root taxon to zebrafish starting from TAO, and the ontologies for the same taxa in the opposite direction, zebrafish to CTOL-root, starting from ZFA. We manually examined the pair of ontologies produced for each taxon in the path and found them to be consistent.

3.10 LESSONS FROM EXISTING ANATOMY ONTOLOGIES

Based on our investigation of many different anatomy ontologies in the course of capturing the background knowledge, developing the algorithm and performing the evaluation, we have encountered some ontology design choices or practices that, in our opinion, warrant discussion. These practices have an impact, positive or negative, on the ability of the ontologies in question to integrate and interoperate with the other ontologies

in the domain. Most existing anatomy ontologies were developed by biologists with little training in ontology engineering practices. Therefore, while some practices may be deliberate, it is possible that others happened simply because they seemed like the easiest way forward to the biologists looking at a single ontology in isolation from the rest of the domain.

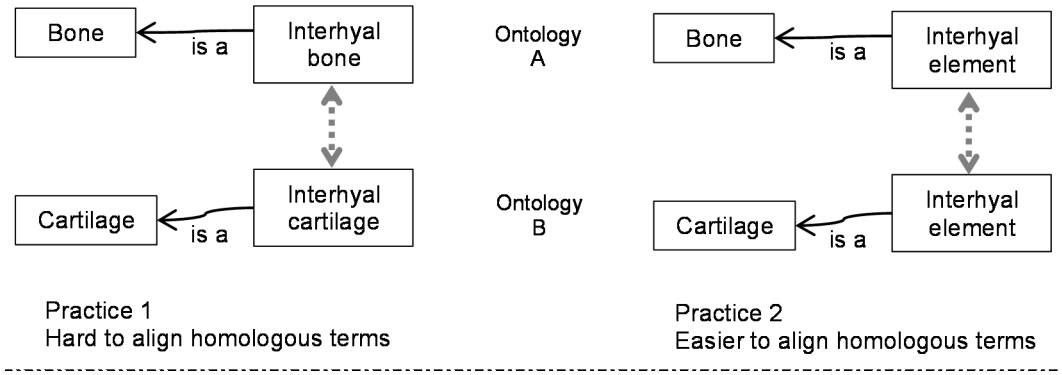


Figure 3.13: A more consistent use of '-element' type entities across single taxon ontology can improve ontology alignment.

- We find the use of '-element' entities in the Teleost Anatomy Ontology (TAO) a significant new practice in anatomy ontologies (Figure 3.12). As mentioned earlier, the Zebrafish Anatomy (ZFA) does not have '-element' entities. One of the reasons for this difference is that TAO is a multi-taxon ontology, and needs to capture the variation across all the taxa in the teleost group. On the other hand, ZFA is a single taxon ontology and does not deal with such variation. For example, TAO contains *interhyal-element* (`TAO:0001892`) as well as both of its possible variations, *interhyal-bone* (`TAO:0000171`) and *interhyal-cartilage* (`TAO:0001511`). ZFA does not need to capture the variation and contains only *interhyal-cartilage* (`ZFA:0001511`).

However, we believe that a better way to model *interhyal-cartilage* in a single taxon ontology such as ZFA is to have an *interhyal-element* and a relationship *interhyal-element is_a cartilage*. This enables easy alignment of multiple single taxon ontologies by making *interhyal-element* an anchor (i.e. a possible homology) among them. This example is depicted in Figure 3.13.

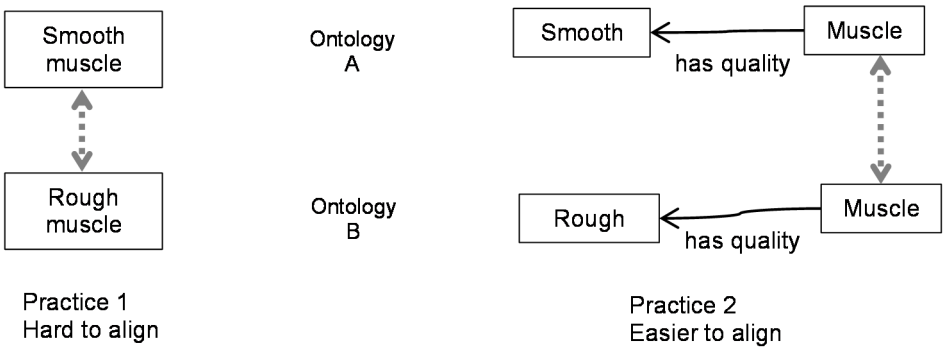


Figure 3.14: Better modeling practices can improve ontology alignment.

- Anatomy ontologies often contain terms that include adjectives in their names or definitions, e.g. the term *smooth-muscle* (`TAO:0005274`) in TAO consists of the adjective *smooth* and an entity *muscle*. Given the existing conventions, our algorithm also produces such terms, in particular when classifying characters are involved (see Table 3.8). As we mentioned earlier Phenotypic Quality Ontology (PATO) provides the commonly used adjectives for this domain. A better way to model such terms may be to model the adjective as a relationship to an appropriate term in PATO, for instance as *muscle has_quality smooth*, where *smooth* (`PATO:0000701`) comes from PATO. Once again, having the term in its primitive form can help ontology alignment algorithms (Figure 3.14).

- Sequences of similar entities are often captured by numbering them. For example, ZFA contains multiple *Weberian-vertebra* (ZFA:0001190) named as *vertebra 1* (ZFA:0001167), *vertebra 2* (ZFA:0001168), *vertebra 3* (ZFA:0001169) etc. The ontology does not capture the relationships among these entities. *Vertebra 2* has the following definition: "vertebra that is posteriorly adjacent to *vertebra 1*". However, there is no formal relationship between the two entities in the ontology, making the definition inaccessible to an inference engine. Given that PATO defines qualities for related entities such as *adjacent to* (PATO:0002259) and *posterior to* (PATO:0001633), ontology developers should strive to capture these relationships as well.

Our algorithm allows automatic generation of single taxon anatomy ontologies based on an existing anatomy ontology and a related phylogenetic study captured in the Morphster meta-model. Our algorithm can be easily extended to produce multi-taxon ontologies as well.

Chapter 4

Knowledge Acquisition using Morphster

Morphster is a domain-specific ontology editor for conducting morphology based phylogenetics research. Unlike general purpose ontology editors such as Protégé [12] and OBO-Edit [11], Morphster builds ontologies that capture knowledge about phylogenetics. The goal of Morphster is to enable biologists to conduct their studies in a way that makes it possible for them to perform knowledge inference across the components of a study. In particular, Morphster is a tool for creating the input knowledge for our inference process for automatically creating new anatomy ontologies. We have defined a meta-model for morphology based phylogenetics that describes domain entities and their relationships, which is the basis for the ontologies created using Morphster.

Images play an increasingly significant role in phylogenetic studies, thus they are at the center of the data model and the user interface of Morphster. Images are commonly used in phylogenetic studies as the definitive basis or exemplars for concepts and are annotated to express hypotheses that may later be validated through the phylogenetic inference methods. Images are used in Morphster as placeholders along a morphologist's normal workflow, capturing initial facts, intermediate hypotheses and final results in the ontology using image annotations. In other words, Morphster is an image-driven ontology editor.

We have implemented the ontology editing actions as side effects of the steps a user takes through the workflow. As a result of this approach, the user interface of Morphster is much more intuitive to morphologists than to knowledge engineers. This is in contrast to conventional ontology building methodology, where a knowledge engineer is often required to assist in the process of encoding the domain knowledge into an ontology. A goal of Morphster is to enable subject matter experts to build their own ontologies, without any assistance from a knowledge engineer.

4.1 ROLE IN PHYLOGENETIC STUDIES

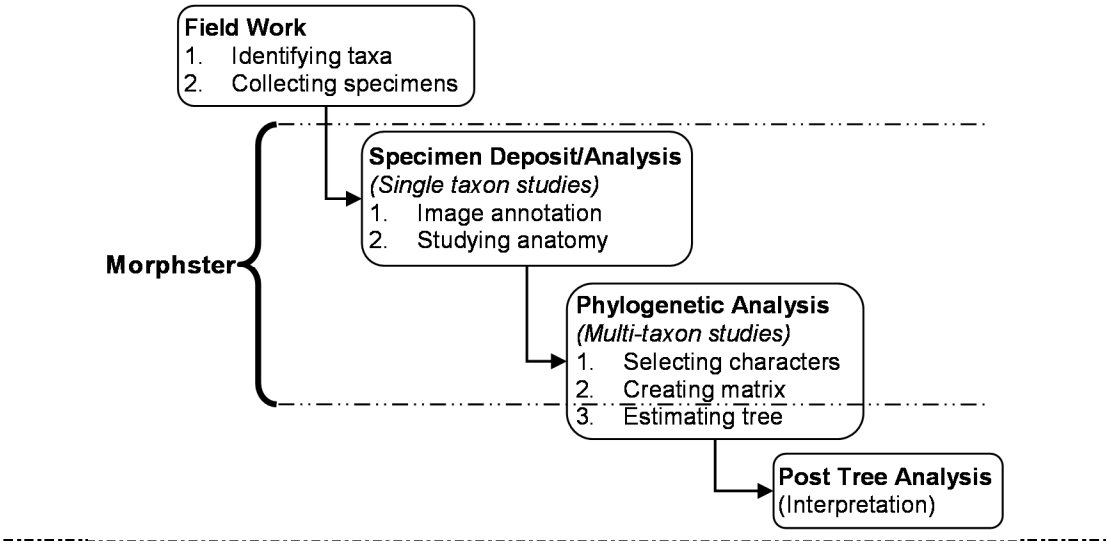


Figure 4.1: Workflow of phylogenetics, and the role of Morphster.

Figure 4.1 presents an outline of steps involved in phylogenetics research and clarifies the role of Morphster in this domain. Being a natural science, research work in systematic biology starts in the field where biologists observe and identify taxa in their natural ecosystems and collect specimens for further study. These studies may be about a

single newly discovered taxon, or multi-taxon studies that involve placing the new taxon in its evolutionary context.

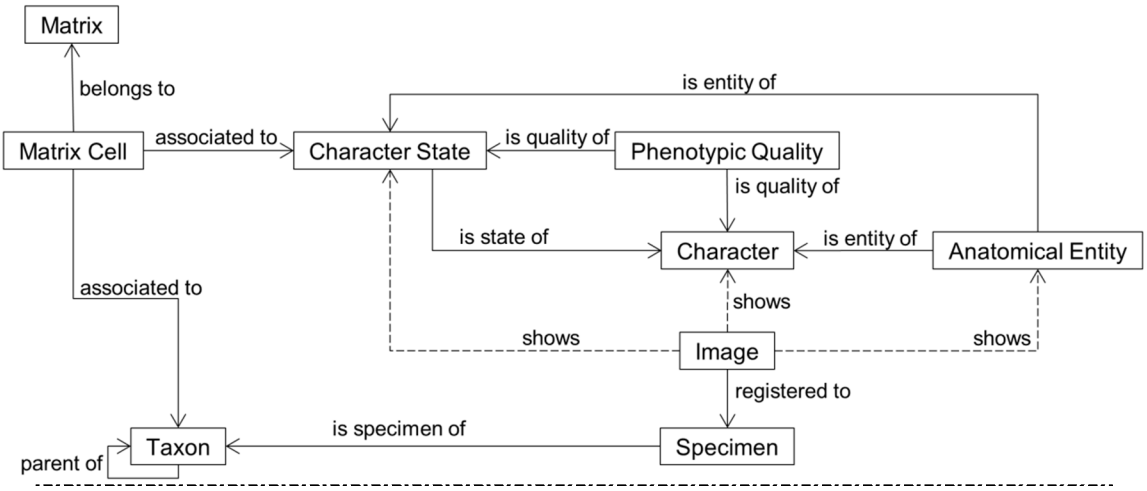


Figure 4.2: Morphster meta-model for phylogenetic studies. The 'shows' associations are shown in dotted lines because they represent a collection of association types called the 'Shows' hierarchy (Chapter 5).

Morphster supports both single and multi-taxon studies. For single taxon studies, Morphster supports preparing annotated image collections from curated specimens, and creating the *Nomina Anatomica*. For multi taxon studies, Morphster supports comparative viewing of imagery from different specimens, as well as anatomy ontologies from multiple taxa, helping scientists identify characters and character states, and building data matrices.

4.2 A META-MODEL FOR PHYLOGENETIC STUDIES

Morphster provides a meta-model for morphology based phylogenetic studies, making it possible to capture the body of scientific knowledge for an entire phylogenetic study as an ontology. We present this meta-model in Figure 4.2 (and its OWL-encoded ontology version as Appendix E).

This meta-model is the foundation of Morphster. It presents the key concepts such as taxon, matrix, character, character state, etc., that form the core of scientific statements developed by morphologists. It also presents other concepts (anatomical entity, phenotypic quality etc.) and relationships (is entity of, is quality of etc.) that are key to properly defining the concepts in the domain.

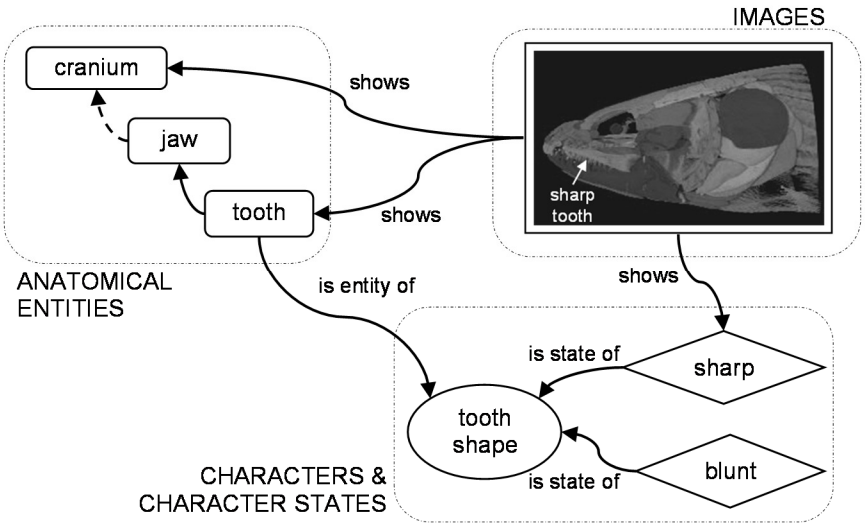


Figure 4.3: An ontology built using Morphster, showing some of the concept types allowed by the meta-model.

The central piece of our meta-model is the image concept and its relationships with other components such as anatomical entities, characters and character states. This model allows morphologists to relate taxa (through specimens) to relevant features (anatomical entities, character states etc.) using images as basis or evidence of the observations.

The case of characters and character states is also of particular significance. A character is a variable often represented as a descriptive statement describing certain qualities of some specific anatomical entities, and character states are the possible values

for that variable or character. As explained in Chapter 3, we treat characters as compound concepts, best captured as a composition of different kinds of terms, such as anatomical entities and qualities or adjectives.

Knowledge from the scientific work done using Morphster is represented in this ontological form (see Figure 4.3 for an example). However, like all other domains, morphologists are trained to recognize and understand certain visualizations of this knowledge, such as in the form of a data matrix. By default, Morphster queries the knowledgebase to produce those specific visualizations of the ontology.

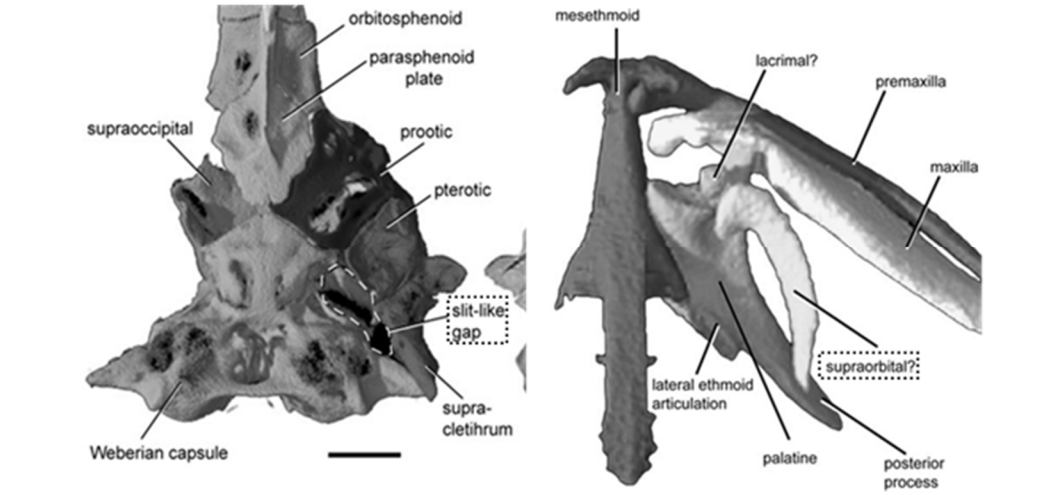


Figure 4.4: Images of *Sarcoglanis simplex* documenting observations. Outlined labels highlight characters or character states, or documented hypotheses.

4.3 KNOWLEDGE ACQUISITION USING MORPHSTER

Morphster's knowledge acquisition process is different from conventional ontology editors such as OBO-Edit [11] and Protégé [12]. Images play a central role in the knowledge acquisition process of Morphster. The process of building an ontology is based on annotating different collections of images at various stages in the workflow.

Image-driven Knowledge Acquisition

We have identified two roles served by images in the morphology workflow. An image may serve as a concept exemplar, or as a record of a scientist's conjecture. For example, Claeson et al. [70] describe a rare catfish called *Sarcoglanis simplex* using images obtained from an adult specimen (see Figure 4.4). These images not only document anatomical entities and possible character states (e.g. slit-like gap) but also new hypotheses (e.g. is the present entity a supraorbital).

In order to support different roles of images we have developed a framework, named the 'Shows' hierarchy, for integrating images with ontologies. In short, the 'shows' relationship for images shown in Figure 4.2 represents a set of relationship types that may be used for image associations depending upon the intended role of an image at a particular point during the study.

Based on the 'Shows' hierarchy, we have implemented image-driven ontology editing actions. The idea is quite straightforward: Image annotations cause updates to the knowledgebase and the progress of the user through the workflow decides the precise actions that are triggered when an annotation takes place.

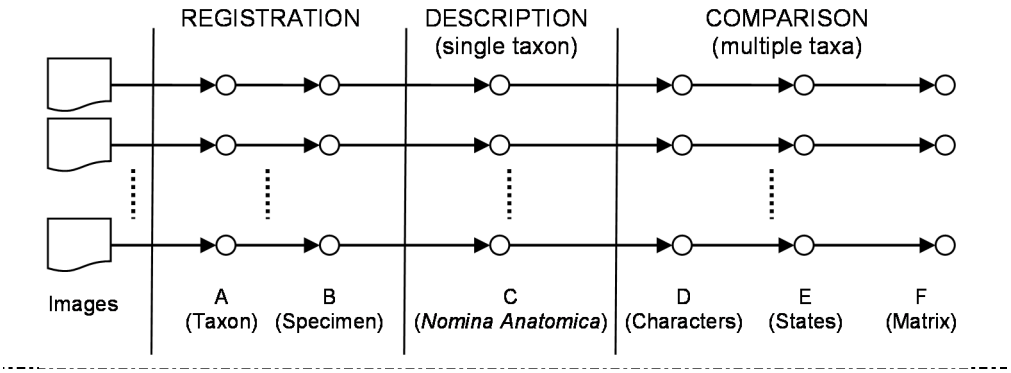


Figure 4.5: Image collections are used as checklists that take a biologist through the workflow, building the knowledgebase in the process.

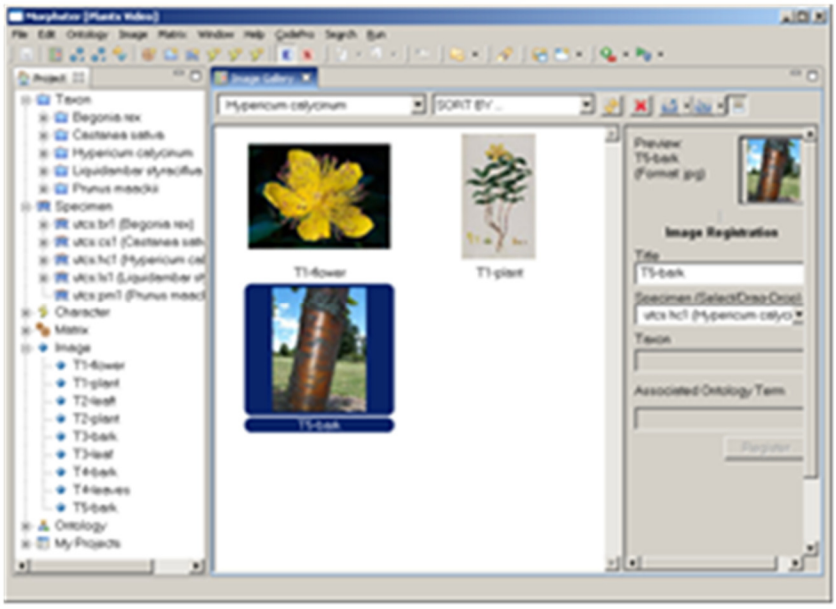


Figure 4.6: Screenshot of Morphster image gallery. Each image goes through its role at a particular step in the workflow and disappears from the gallery. The image collection serves as a check list of to-do items at each step in the workflow.

Iterative Population of the Ontology

Conventionally, knowledge is acquired in a frame-by-frame manner, i.e. a knowledge engineer populates a particular concept before moving on to the next. We refer to this a *depth-first* approach to ontology building. Applied to a domain like systematic biology, that would mean populating the knowledge base with all the information about a particular taxon, from identifying anatomical entities to recognizing and defining characters and states etc., before moving on to another taxon. This process contradicts with the conventions of systematic biologists.

In systematic biology studies, often a goal is to study new taxa by means of comparing them with each other as well as with known and well described model taxa. This entails that knowledge about each taxon is not available in isolation from other taxa.

In other words, the knowledge about all the taxa is discovered in parallel as they are investigated in comparative settings, using similar images from each taxon. These comparative settings are sometimes called *3-taxon statements*, where three taxa are compared to identify which two of the three may be more closely related to each other.

Morphster's knowledge acquisition process has been derived from this methodology. We use images as placeholders for updates to the ontology (see Figure 4.5). The collections (or queues) of images serve as checklists for work to do. Per the stages in the workflow, images may belong to the registration queue, description queue or the comparison queue. The user interface of Morphster provides a metaphor to manage these image collections to keep track of progress (see Figure 4.6).

Images belonging to the registration queue go through a registration process to identify the taxon and specimen it belongs to, as well as the metadata fields. As each image is registered, it moves to the description queue. Images in the description queue are used to identify anatomical entities, resulting in the development of the *Nomina Anatomica* for each taxon. As each of these images is annotated, it updates the relevant *Nomina Anatomica* with frames corresponding to anatomical entities and their slots that represent anatomical relationships. This step corresponds to single taxon studies. Finally, the images go into the comparison queue for multi-taxon analysis. Each image at this stage starts as a placeholder for incomplete definitions of one or more characters. As the image is annotated, the relevant characters get further populated with their character states. Finally, these annotations contribute towards automatically producing the character data matrix. So, in an *iterative* style, the ontology is populated with the domain knowledge.

4.4 IMAGE-DRIVEN PHYLOGENETICS

Single Taxon Study: Describing a Taxon

Morphster supports single taxon studies using digital imagery belonging to a particular taxon. In these studies, the imagery is annotated with corresponding anatomical entities and features that belong to a *Nomina Anatomica*. Figure 4.7 presents a sample screenshot showing imagery being annotated with anatomical entities drawn from a *Nomina Anatomica*.

There are usually two ways in which the digital imagery is annotated. One of the ways is to use an existing *Nomina Anatomica*, which may belong to a similar model organism, and derive terms from that ontology for the annotation of the taxon under study. For example, given a picture of a catfish skull and the ontology for zebrafish, a morphologist may need to use the zebrafish 'skull' ontology term to describe the homologous catfish skull shown in the image.

Another method of image annotation involves not only the association of images with appropriate terms, but also the creation of a new *Nomina Anatomica* ontology in parallel. Our work on associating images with ontology terms is one of the contributions of this dissertation and is explained in detail later (see Chapter 5).

Multi-Taxon Study: Characters and Character States

For multi-taxon studies, Morphster provides ways for morphologists to identify characters and character states, and build matrices.

As shown in Figure 4.7, users can view multiple images side-by-side in order to identify variations of features across taxa. They can create characters and character states by drawing ontology terms of different kinds (anatomical entities, phenotypic qualities etc.), and annotating the images them. A specific user interface titled 'concept

workspace' allows the users to compose the necessary terms as new characters and/or character states.

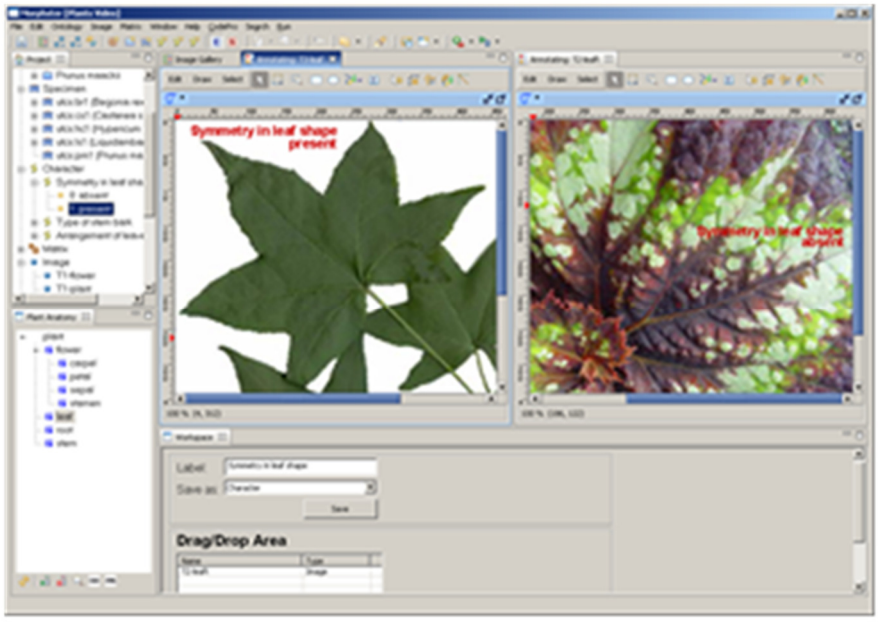


Figure 4.7: Screenshot of comparative use of images and concept workspace (bottom panel). The workspace enables formation of new concepts through the composition of existing ontology terms.

Also, once a complete set of characters and character states has been created, the associations of registered images with character states are used to build a data matrix by inferring the values of the matrix cells automatically.

4.5 DEVELOPMENT CHALLENGES

The development of Morphster faced challenges regarding support for ontology languages as well as storage and querying of large ontologies. During the development of Morphster, we have developed solutions and practical experience for these challenges.

OBO and OWL ontology languages are both used in the biology community. We learned from our early experiences in the Morphster project that it is essential to build a system that is not only capable of serving both ontology languages, but also of integrating ontologies represented across languages.

Our solution to this challenge was to create a round-trip transformation of OBO ontologies to OWL. This is a contribution of this dissertation and is explained in detail in Chapter 6. We developed a methodology for translating OBO ontologies to OWL using the organization of the Semantic Web. The approach enabled us to create quickly create transformation rules and identify potential mismatches. We have collaborated with other OBO community members and our mapping is now an accepted and widely used standard for the biomedical community.

Morphster maintains all the data produced during phylogenetic studies as a part of a single central ontology. The second challenge for Morphster was to identify the best way to manage this ontology.

Most of the meta-model could easily be modeled into a relational database. However, anatomical entities and phenotypic qualities come from arbitrary ontologies that may be imported into Morphster, making it infeasible to use a purely relational structure. On the other hand, storing an arbitrary ontology in a relational DBMS is possible, and a specific solution called a *triple store*, is often used for this purpose. It was possible to store the entire ontology in a triple store. We experimented with the most widely used framework at the time, Jena [71] and its triple store implementation Jena SDB [72]. In this setting, we quickly ran into scalability issues, especially with large ontologies (25k+ concepts and 50k+ relationships).

Not wanting to rely completely on Jena as our generic framework, we made two changes: First, we retracted partially from using only a triple store. Only the imported ontologies are now stored in the triple store, while the rest of the data stored in a relational schema, using URIs to maintain references. Second, instead of using Jena's API to query the data from the triple table, we created a more efficient API to suit our application needs. Compared to Jena's memory intensive approach, our API pushed most of the work to the RDBMS query engine, dramatically improving query performance. Leading with these lessons on the use of triple stores and RDBMS, we have been working on bridging the gap between SQL databases and ontologies. A representative publication of our work in this direction is [73].

4.6 ONTOBROWSER

Ontologies created in Morphster are publishable on the Internet using a companion component called Ontobrowser [74]. Ontobrowser allows users to browse ontologies and images and search for terms in the ontologies. Annotations on imagery created through Morphster are rendered as hot links for efficient browsing. Ontobrowser also serves as the query front-end to the Morphbank database of natural specimen images [75]. Given image annotations, terms are sent to Morphbank and matching images returned. This feature is currently being extended so that if matching images are not initially found, the search terms are automatically generalized (based on the ontology) and as assessed by ontology similarity measures, similar images are retrieved from Morphbank [76].

4.7 IMPLEMENTATION AND USE

Morphster is a Java based application built using the Eclipse SWT API for user interface. At the back-end, we deployed a Microsoft SQL Server database, which is accessed by a JDBC API. For communicating with external services we use an XML-based Web Service infrastructure.

Morphster is a productivity tool for systematic biologists. A full evaluation requires their feedback on the tool and joint assessment of quality of the resulting ontologies. Such activities are just starting, including deployment on the Fishes of Texas project [77]. Even so, we have exploited the system for smaller activities as follows. These studies represent a promising beginning to the application of Morphster in phylogenetics.

Two NSF funded AToL efforts have created ontology representations of their *Nomina Anatomica*: Spider Ontology by Spider AToL [78] and Hymenoptera Anatomy Ontology by the Hymenoptera group HymAToL [79]. Illustrated versions of these ontologies ("Spider with Images" and "Hymenoptera with Images") can be found at the Ontobrowser web site [74]. "Spider with Images" contains over 550 terms illustrated with 139 images. "Hymenoptera with Images" contains over 1000 terms illustrated with 369 images.

Morphster has been used to create a plant ontology using the stack-of-photos use case, building characters and character states as well as a final matrix for 5 plant taxa. We have also created a short *Nomina Anatomica* for *Herrerasaurus*, one of the earliest dinosaurs, using imagery obtained from fossils. These studies appear in unpublished tutorial materials.

Chapter 5

Capturing Biological Hypotheses using Imagery

Digital imagery often serves as a proxy for biological specimens and is the foundation for all the scientific work in such phylogenetic studies. These images (computed tomography (CT) scans, field photographs, sketches etc.) are used to capture the ground facts, and more importantly are the basis for building and recording scientific conjectures that are studied and proved or disproved at the conclusion of analytic work. We have already provided an example of these uses of images in Chapter 4 (see Figure 4.4).

In another case, Ramírez et al [80] provided a protocol for documenting newly discovered spider taxa. They defined approximately 400 standard views on spiders. They proposed that when a spider species is newly discovered a set of standard view images be deposited in an online digital collection at the same time the prototype specimen is deposited into a museum collection. The online collection categorized by taxon (species) and standard view serves as a basis for an ontology of all spiders.

While anatomy ontologies are another means for capturing the ground facts, the lack of means for capturing hypotheses using ontologies has been a major hindrance in building ontology based tools for conducting phylogenetic studies. A practical solution

requires a framework for capturing scientific facts and hypotheses by integration of images with ontologies.

In this context we explore the connection between images and ontology terms. Usually, ontology terms are annotated with textual definitions of the concepts they represent. These definitions may be ambiguous or open to many different interpretations. In biology, illustrations are often employed in place of textual descriptions to unambiguously define concepts. In order to integrate images with ontologies, it is necessary to understand their roles, and to develop a framework that captures these roles.

Our work provides a solution for integrating scientific images with ontologies. We recognize the need for identifying and formally defining different roles of images in ontologies and provide multiple ways of connecting images to classes based on the intended roles of images. This enables us to build richer ontologies capable of inferring more accurate results on common scientific queries than existing ontology based image retrieval systems based on ontologies. It also gives an opportunity for creating scientific use cases based on images that simplify ontology creation and alignment for subject matter experts.

5.1 RELATED WORK

Ontology based image annotation and retrieval is an active area of research. The existing body of work in this area ranges from the simple use of domain ontologies to annotate images to the development of sophisticated mechanisms to describe and infer spatial and temporal relationships of depicted objects.

Schreiber et al. [81] explored the use of a photo annotation ontology and a domain ontology to index and search collections of photographs. In a qualitative comparison between ontology based and keyword based systems they showed that ontology based

retrieval returned more accurate results. Hyvönen et al. [82] also annotated images using domain-specific ontologies.

The PhotoStuff project [83, 84] was motivated by the need for toolkits that would allow annotation of multimedia content on the Web. The PhotoStuff tool provides users the ability to annotate whole or regions of images to identify instances of ontological classes.

Pastra et al. [85] demonstrated the use of ontologies to describe crime scene photos. They used a domain ontology called OntoCrime to translate natural language photo captions that often describe spatial relationships, into relational facts. By applying the same translation on queries to the knowledgebase, they were able to use pattern matching on the facts to retrieve images. In a later work, Pastra used a similar approach to describe 3D indoor scenes depicted in 2D pictures [86].

Petridis et al. [87] and Hudelot et al. [88] worked on bridging the gap between low level descriptive features of images (e.g. dominant color) and content descriptions based on ontologies. MPEG-7 is a standard for creating audiovisual descriptions [89]. Petridis et al presented an approach based on prototype instances for integrating MPEG-7 descriptions expressed in RDF with domain ontologies, using a tool called M-OntoMat-Annotizer. Hudelot et al. used an ontology backed by fuzzy logic to capture spatial relationships among objects depicted in pictures. They demonstrated the use of ontologies and fuzzy logic to perform reasoning on medical images.

Bertini et al. [90] further investigated the problem of obtaining a complete expression of information in digital media, and included support for maintaining spatial as well as temporal relationships among entities. They used domain ontologies to

annotate instances found in the media, and used rule based pattern matching for inference.

The existing work in the area of image-ontology integration clearly demonstrates the value of ontologies in describing the contents and semantics of images and other multimedia. A common feature of these efforts is the use of existing domain ontologies to identify instances of given concepts.

Our approach and solution is based on the use of images in scientific work. Our concern is to capture the roles of images as *exemplars* (to capture scientific facts) and as placeholders for recording conjectures. We aim to integrate images into ontologies by providing formally defined constructs for capturing these roles, and using them as a basis for developing ontologies that capture scientific knowledge. Moreover, while it is typical for images to be used to identify instances of domain concepts, in scientific work images often define the concepts themselves. Specifically, some kinds of images, such as hand-drawn sketches, are often used in scientific work. These images are identified as depictions of concepts that ultimately belong in the ontologies, and are not depictions of any real world instances of those concepts.

In addition, the role of images to record conjectures means that the class for an image may be unknown to the scientist. In this situation, it may be inaccurate and often impossible to associate the image with an instance, since the actual class for the image may not even exist in the knowledgebase. In such cases, the scientist may choose to express similarity with an existing class. A way to implement this scenario in a system that associates images to instances is to introduce an unknown class into the ontology and to mark the image as its instance. However, at the stage of recording conjectures, it is

unknown whether that class should exist or not, and can only be determined once the scientific process has taken its course.

These uses of images are inherent to our work; hence image annotations are made directly with the domain concepts. Our inference rules also work on these direct associations between images and ontologies.

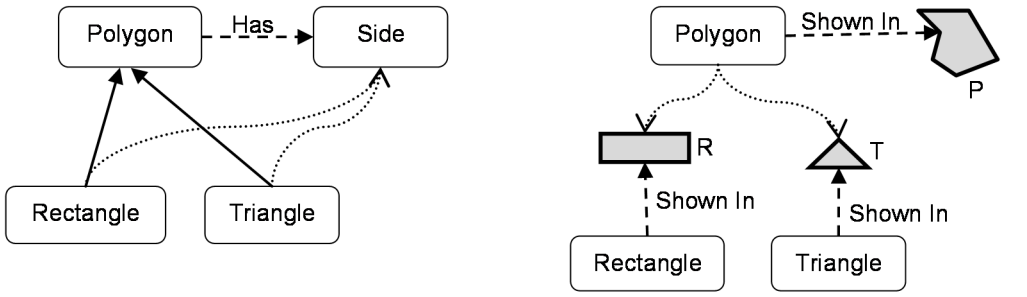


Figure 5.1: Representing class hierarchy and its effect on relationships and image associations using an ontology of common shapes as an example. Arrows with solid lines represent subclass relations, dashed lines show explicitly defined relations, and dotted lines are implicit relations.

5.2 FRAMEWORK FOR INTEGRATION OF IMAGES WITH ONTOLOGIES

We have defined specific roles of images in ontologies and formalized unique semantic properties of image associations to build a framework for image ontology integration. Ontology systems provide standard mechanisms for creating class relationships (triples that connect two classes to each other) and metadata attributes (e.g. textual definitions of classes). A comparison of metadata attributes with image associations is straightforward. Metadata attributes belong to a class, whereas images can exist as independent entities in the ontology. Hence image associations are not similar to

metadata attributes. Compared to class relationships, image associations have different semantics. Here we elaborate on this point.

Unique Semantic Properties of Image Associations

Consider a section of an ontology of common shapes shown in Figure 5.1a. The ontology expresses class relationships such as *SubClassOf(Rectangle,Polygon)*, *SubClassOf(Triangle,Polygon)* and *Has(Polygon,Side)*. Because class relationships follow inheritance rules we can infer *Has(Rectangle,Side)* and *Has(Triangle,Side)*. Figure 5.1b shows the same *Polygon*, *Rectangle* and *Triangle* classes, connected to their respective images using *ShownIn* associations and *ShownIn(Polygon,P)*, *ShownIn(Rectangle,R)* and *ShownIn(Triangle,T)* hold.

If we apply inheritance, we get *ShownIn(Rectangle,P)* and *ShownIn(Triangle,P)*, which are obviously incorrect statements. However, there is a different possibility: *ShownIn(Rectangle,R)* and *ShownIn(Triangle,T)* imply *ShownIn(Polygon,R)* and *ShownIn(Polygon,T)*. In other words, *an image that shows a rectangle also shows a polygon* as well as *an image that shows a triangle also shows a polygon*. Accordingly, image associations of children may imply image associations of the parent class which is in contrast to inheritance where a relationship of a parent class implies a relationship for its children. Clearly, image associations have different semantics than class relationships. We refer to this property of image associations as the **contra-inheritance** property.

It is important to note that this property is novel only in the context of our approach and within the given domain. In the systems that operate on instance associations, the effect of collecting images from the instances of child classes is similar to the contra-inheritance property.

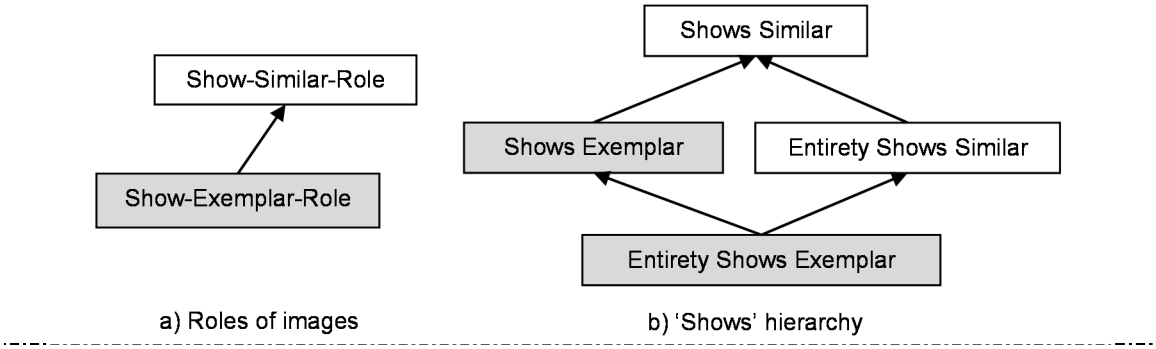


Figure 5.2: Hierarchy of image roles in ontologies, and inheritance hierarchy of image association types, called 'Shows' hierarchy. Show-Similar-Role is captured by association types in white background, and Show-Exemplar-Role is captured by association types in grey background.

Specification of Image Roles

Images can play two key roles (Figure 5.2a): an image is used as a definitive example (or *exemplar*) of an entity (`Show-Exemplar-Role`), or an image is used to hypothesize similarity between two entities (`Show-Similar-Role`). While it is easy to understand `Show-Exemplar-Role`, the use of similarity in `Show-Similar-Role` requires further explanation.

In the context of systematic biology, we can define similarity in terms of shared evolutionary origin of two entities (*homology*) or convergent evolutionary origin (*homoplasy*). In other words, an assertion that 'two entities are similar' is a hypothesis that may lead to establishing a homology or homoplasy between the entities in question, with continued investigation.

'Shows' Hierarchy

After organizing the roles of images in an inheritance hierarchy (Figure 5.2a) we proceed to define types of associations that will enable us to capture image roles in ontologies.

Table 5.1: Basic predicates and associations needed for defining image associations.

Predicate	Definition
$Cls(x)$	True when *x* is a class, otherwise false.
$Img(x)$	True when *x* is an image, otherwise false.
$SubClass(x, y)$	*x* is a subclass of *y*. It is transitive, reflexive and anti-symmetric.
$PartOf(x, y)$	*x* is a part of *y*. It is transitive, reflexive and anti-symmetric.
$InstanceOf(x, y)$	*x* is an instance of *y*.
$SimilarTo(x, y)$	*x* is similar to *y*. It is transitive, reflexive and symmetric.
$ExemplarOf(x, y)$	*x* is an exemplar of *y*. It is reflexive and anti-symmetric. Also, by definition: $ExemplarOf(x, y) \rightarrow SimilarTo(x, y)$
$Plays(x, y)$	*x* plays role *y*, as defined by Fan et al. [91]. A similar property exists in a later work by Mizoguchi et al. [92].

We have identified four types of associations between images and ontological concepts, henceforth called *image association types*. We have organized them into an inheritance hierarchy, called the 'Shows' hierarchy (Figure 5.2b). Image association types towards the top of the hierarchy are more general than the ones further down the hierarchy. Hence, *Shows Similar* is the most generic type, and *Entirety Shows Exemplar* is the most specific. Here we list and define these associations:

- ***Shows Similar (m, x):*** some part of image *m* plays `Show-Similar-Role` for class *x*. This is the most general association type used to record conjectures.
- ***Entirety Shows Similar (m, x):*** the entire image *m* plays `Show-Similar-Role` for class *x*. This association type is also used to record conjectures. Notice that the entire image is considered a part of itself, thus making this association type more specific than the previous one.

- ***Shows Exemplar (m, x):*** some part of image *m* plays `Show-Exemplar-Role` for class *x*. This association type is used to identify exemplars. It is a specialization of *Shows Similar* considering that an exemplar association depicts a hypothesis that has been proved through a scientific process.
- ***Entirety Shows Exemplar (m, x):*** the entire image *m* plays `Show-Exemplar-Role` for class *x*.

Table 5.2: Formal rules for image associations introduced by the 'Shows' hierarchy.

Associations & Definitions
$ShowsSimilar(m,x) \rightarrow Plays(m, Show-Similar-Role)$
$EntiretyShowsSimilar(m,x) \rightarrow$ $ShowsSimilar(m,x) \wedge EntiretyShowsExemplar(m,y) \wedge SimilarTo(y,x)$
$ShowsExemplar(m,x) \rightarrow$ $Plays(m, Show-Exemplar-Role) \wedge ShowsSimilar(m,x)$
$EntiretyShowsExemplar(m,x) \rightarrow$ $ShowsExemplar(m,x) \wedge EntiretyShowsExemplar(m,x) \wedge$ $\forall n(PartOf(n,m) \wedge EntiretyShowsExemplar(n,x) \rightarrow n = m)$

Formal Semantics of Image Association Types

In Table 5.1 we present a listing of the basic predicates and their definitions that will be used in formalizing semantics for image association types introduced above. Some concepts, like `is-a` (*SubClass*) and `part-of` (*PartOf*), are commonly used in biomedical ontologies and are therefore standardized by the biomedical community [27]. In Table 5.2 we further define the formal rules that govern the use of image associations. These rules include a formal specification of the inheritance relationships shown in Figure 5.2b.

In addition, we define the following key rules that make common scientific queries possible:

- **Equality of Images:** Trivially, two images may be considered equal if they have the same identifier or have the same bits in the image file. Here we define a non-trivial equality. Two images *x* and *y* are equal, or *Equal(x,y)*, if they serve as exemplars for exactly the same set of ontological classes.

$$Img(x) \wedge Img(y) \wedge Cls(w) \wedge$$
$$ShowsExemplar(x, w) \equiv ShowsExemplar(y, w) \rightarrow Equal(x, y)$$

  The notion of equality in this rule does not refer to the two images being the same; it rather presents a way of recognizing content-wise equality of a pair of images.

- **Similarity of Images:** Two images *x* and *y* that capture similarity hypotheses for exactly the same set of ontological classes, are similar.

$$Img(x) \wedge Img(y) \wedge Cls(w) \wedge$$
$$ShowsSimilar(x, w) \equiv ShowsSimilar(y, w) \rightarrow SimilarTo(x, y)$$

  This is a very basic way of defining similarity between images used in morphology. Based on the needs of a particular domain, different kinds of similarity rules may be defined. For example, it may be possible to introduce a similarity index in the rule to quantify the similarity between the images.

- **Sub-Image:** If an image *y* serves as exemplar for all the classes exemplified by image *x*, image *x* is considered a sub-image of image *y*, or *SubImage(x,y)*.

This property is very similar to the subclass relationship among classes, or the sub-property relationship among properties.

$$Img(x) \wedge Img(y) \wedge Cls(w) \wedge$$
$$\big(ShowsExemplar(x,w) \rightarrow ShowsExemplar(y,w)\big) \rightarrow SubImage(x,y)$$

- **Contra-Inheritance:** If *x* is a subclass of *y*, and an image *w* shows *x*, then image *w* also shows *y*. This is the scenario that was discussed earlier as well (see Figure 5.1).

$$Cls(x) \wedge Cls(y) \wedge SubClass(x,y) \wedge Img(w) \wedge$$
$$Shows^*(w,x) \rightarrow Shows^*(w,y)$$

Notice the use of predicate *Shows* in this rule. It represents any of the four association types present in the 'Shows' hierarchy. In essence, this rule represents four independent rules, one for each image association type.

Querying Ontologies using Image Association Semantics

Given the formally defined association types and inference rules, we present an example fish ontology (Figure 5.3) and present a list of possible scientific queries that are better handled by our framework compared to the previous work (Table 5.3). For each listed query we indicate relevant association types and rules involved in providing the answer.

Because image association types were defined based on the roles of images, it is necessary to present queries to the system in a way that clarifies the role of concerned images. To keep the query statements simple, `Show-Exemplar-Role` is assumed by default unless `Show-Similar-Role` is explicitly mentioned in the query (e.g. Q6).

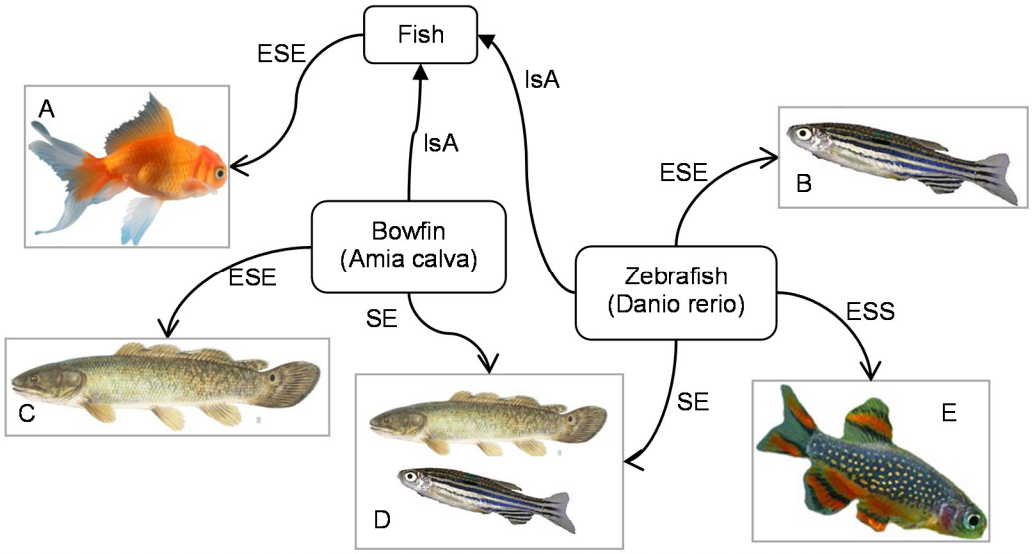


Figure 5.3: A small fish ontology. Images are associated to the concepts in multiple ways, unlike the other ontology based image retrieval systems. Abbreviations: ESE=EntiretyShowsExemplar, SE=ShowsExemplar, ESS=EntiretyShowsSimilar, IsA=Subclass relationship.

Table 5.3: Selected queries and results comparing our work (Shows) to the other systems.

Query	**Shows**	**Other**
Q1. What are the pictures of Zebrafish?	{B, D}	{B, D, E}
Q2. Find model pictures (exemplars) of Bowfin.	{C, D}	{ }
Q3. What are the pictures of fish?	{A, B, C, D}	{A}
Q4. Find some possible exemplar images of Fish.	{A, B, C, D}	{ }
Q5. Show sub-images of picture D.	{B, C, D}	{B, C, D}
Q6. Show some pictures that are similar to the picture A.	{A, B, C, D, E}	{ }

Q1 (‘what are the pictures of Zebrafish?’) will give us images B and D in our system, given the use of an exemplar association type for the images. Other IR systems, where there is no distinction between association types, will additionally return image E, even though it is not an exemplar for class Zebrafish.

Q2 (‘find model pictures (exemplars) of Bowfin’) will find images C and D. It will require the system to find each image m that plays exemplar role, i.e. satisfies *Plays(m,Show-Exemplar-Role)*, for class Bowfin.

Q3 (‘what are the pictures of Fish?’) and Q4 (‘find some possible exemplar images of Fish’) are identical queries for our system, given the exemplar role assumption. Inference for these queries makes use of the contra-inheritance rule and marches down the hierarchy to collect exemplar images. For other IR systems, only Q3 is answerable (image A), since Q4 explicitly requires use of exemplar role which is undefined.

Q5 (‘show sub-images of picture D’) requires the use of sub-image rule to find images that serve as exemplars of only the things exemplified by image D. This is an example of querying-by-example where an image itself serves as a part of the query. This particular query demonstrates the use of an image as a universal set of concepts. All the images retrieved should be annotated only with the concepts in the given universal set. These kinds of queries are supported in most systems that perform inference on the image annotations.

Q6 (‘show some pictures that are similar to the picture A’) is another image based query. It addresses the similarity role of images. This particular query makes use of the rules for similarity of images and contra-inheritance to go down the hierarchy starting from *Fish* and produce the set of all images {A, B, C, D, E}. If we modify the query to finding pictures similar to image B instead, we will get the same result even though using contra-inheritance from image B (or *Zebrafish*) does not include going to parent class *Fish* and hence finding image A, for instance. However, symmetric and transitive properties of similarity enable us to finding the complete set in this case as well. In other words, since Q6 tells us that *SimilarTo(B,A)*, it implies *SimilarTo(A,B)*. By transitivity,

for any image *x* such that *SimilarTo(x,A)* holds, *SimilarTo(x,B)* holds as well. Hence the modified query can be readily transformed back to finding images similar to image A, i.e. Q6.

5.3 MORPHSTER ONTOLOGY DEVELOPMENT USE CASES

Our modeling of the image-ontology integration framework is driven by the needs of phylogenetics studies, and hence Morphster workflows. Here we present three use cases that are made possible by our framework. The first one, called *stack-of-photos*, is a trivial use case that is of great value to biologists as a means for quickly building and publishing their ontologies. For the sake of presenting a comprehensive description of our work, we present that use case first and then move on to the other use cases.

Algorithm:

1. Compile a collection of images
2. Initialize a new ontology by providing a starting point (a root class)
3. Import the images into the system
4. Associate and annotate each image
   a. Attach image to a class in the ontology
   b. Label contents on the image
   c. Mark image as complete
5. Publish a finished ontology

Figure 5.4: Outline of the algorithm for the stack-of-photos use case.

Building Ontology with Stack-of-Photos

The *stack-of-photos* use case demonstrates the utility of our work to simplify anatomy ontology building for a biologist. Compared to a typical scenario where a knowledge engineer would be required to assist in encoding the ontology, our work

translates typical image annotation actions of a biologist into updates to the ontology. Given a collection of images that are exemplars of anatomical entities of a taxon, creating its ontology is straightforward (see Figure 5.4).

After initializing an ontology with a root term like *whole organism*, start with an image that can serve as an exemplar of the root, and create an *Entirety Shows Exemplar* association between the image and the root. For each new term *T* labeled by the user on the exemplar image, create a *Shows Exemplar* association between T and the image. In the workflow, it also implies a `part-of` relationship between *T* and *whole organism*. Iteratively, each of the new terms can be attached to an image in the collection using *Entirety Shows Exemplar* association, and any new classes on those images will create further parts of these classes. When this process is completed for the entire collection of images, the result will be a hierarchical ontology consisting of all labeled classes based on `part-of` relationships.

Aligning or Matching Ontologies

Determining correspondences between classes in multiple ontologies is a significant semantic integration problem [93, 46] and is significant to systematic biology. Bodenreider et al. [94] and Mork et al. [95] provided solutions for creating alignments between mouse and human anatomies. A common feature of their approaches was to first create a lexical alignment by identifying shared reference concepts or *anchors* in the ontologies using a dictionary. These anchors, which are either proven homologies or simply hypothesis, are then used for further alignment of ontologies.

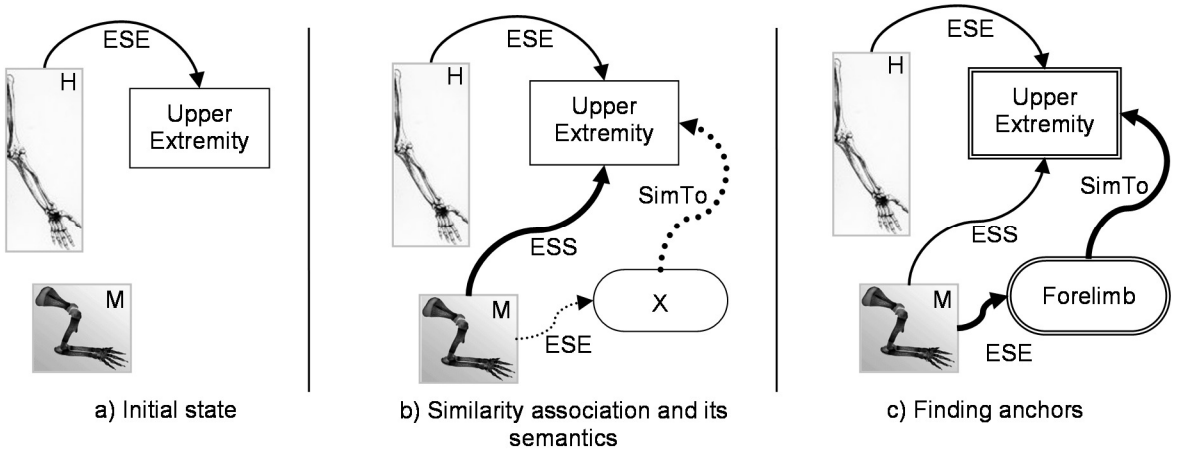


Figure 5.5: Image associations simplify creating anchors for ontology alignment. Terms in rectangles belong to human anatomy, and the ones in rounded boxes are from mouse anatomy. Abbreviations: ESE = EntiretyShowsExemplar, ESS = EntiretyShowsSimilar, SimTo = SimilarTo. Solid lines: explicit relationships; dotted lines: implied relationships resulting from application of formal semantics. Double lined box: anchor.

In Figure 5.5, we demonstrate the use of image associations and their semantics to create anchors between human and mouse anatomies. Starting from the human anatomy ontology, we use the similarity associations to connect shared concepts. Figure 5.5a shows the initial state of the ontology, where M is a new image that has not been connected to any classes so far. If a new association *Entirety Shows Similar* is created between image M and class *Upper Extremity*, the semantics imply that the class *Upper Extremity* is similar to an unknown class *X* (see Figure 5.5b). Finally, if image M is explicitly connected as an exemplar of a class, say *Forelimb*, in mouse ontology. This may result in identifying the unknown class *X* as *Forelimb*, and a similarity is established between *Forelimb* and *Upper Extremity*, marking them as anchors. In other words, *Forelimb* and *Upper Extremity* are marked as possibly homologous entities.

Extracting New Ontologies from Model Ontologies

While building new ontologies of anatomies is very common [16], such efforts often build around existing ontologies of model organisms. For example, Dahdul et al. [10] developed a multi-species Teleost Anatomy Ontology (TAO) using the Zebrafish Anatomical Ontology (ZFA) as a reference. Starting as a clone of ZFA, TAO now contains over 400 new terms that describe teleost fishes. The Amphibanat project [41] is another example of building a new ontology, an ontology of amphibian anatomy, from a model organism ontology. They extracted relevant portions of the class hierarchy and other relationships from ZFA to form the initial framework for their ontology.

We believe that the similarity capturing role (`Show-Similar-Role`) of images in our work can simplify the task of using existing ontologies to extract relevant classes and relationships for new ontologies. It also promises to maintain cross-references between the two ontologies seamlessly by automatically using image associations to maintain anchors.

A common feature of the use cases mentioned above is that it becomes simpler for a subject matter expert to develop ontologies without requiring assistance from a knowledge engineer. Our specification of roles of images and novel image association types allow users to develop knowledge bases while following the customary method of using images to observe and document knowledge.

5.4 IMPLEMENTATION FOR THE SEMANTIC WEB

In order to demonstrate our work on the Semantic Web, we provide an implementation of the 'Shows' hierarchy and relevant constructs (see Appendix F for complete implementation) in RDF and OWL. In this section, we discuss some parts of this implementation. Since the existing relationship types are not suitable for image

associations, we provide a new kind of association to support relationships between images and classes. We have called this association `ShowsProperty`. Image associations in the 'Shows' hierarchy are instances of `ShowsProperty`.

```
<rdfs:Class rdf:ID="ShowsProperty">
    <rdfs:label>ShowsProperty</rdfs:label>
    <rdfs:subClassOf rdf:resource="&rdf;Property"/>
</rdfs:Class>
<ido:ShowsProperty      rdf:ID="showsSimilar" />
<ido:ShowsProperty      rdf:ID="entiretyShowsSimilar" />
<ido:ShowsProperty      rdf:ID="showsExemplar" />
<ido:ShowsProperty      rdf:ID="entiretyShowsExemplar" />
```

In order to ensure that the associations in the 'Shows' hierarchy always associate an image to a concept in the ontology, we have associated a domain with the *Shows Similar* property.

```
<ido:ShowsProperty rdf:about="#showsSimilar">
    <rdfs:label>Shows Similar</rdfs:label>
    <rdfs:domain rdf:resource="&ido;Image" />
</ido:ShowsProperty>
```

The rest of the image associations inherit from the *Shows Similar* association, and thus also have instances of images their domain.

5.6 CONCLUSION

We provide a novel way of associating images with ontologies by providing formally defined image association types. As a result, these ontologies provide interesting inference and content based querying opportunities on images. Existing ontology based

image retrieval systems do not provide this kind of formalism and are therefore limited in terms of querying and inference.

In natural sciences, especially systematic biology, images often serve as an authoritative basis for definition of concepts. Our approach towards the integration of images and ontologies allows us to use images as building blocks for ontologies in such domains. This allows for easy development of ontologies by domain experts, reducing the overhead of involving knowledge engineers in the process. Our *stack-of-photos* use case provides a simple example. While it is a straightforward procedure, the absence of the ability to build even primitive ontologies using other image-ontology integration systems demonstrates the novelty of our approach.

Even though extending existing systems to create anatomy ontologies may be a simple matter, this alone does not serve the purpose of conducting science using images as a fundamental source of knowledge. Identification of the possible roles for images is critical. While in existing systems images are used to tag instances of ontology concepts, we have identified two key roles that are implicit in scientific approach. When images serve as exemplars of concepts, they help in building a domain knowledge base. On the other hand, when an image serves as a means to document scientific hypothesis, it not only supports scientific workflow, but also building a knowledge base for a broader domain or bridging the gaps between ontologies by matching concepts in different but scientifically related domains.

Chapter 6

Mapping between OBO and OWL

Ontology systems Open Biomedical Ontologies (OBO) [16] and the Semantic Web [17], each provide different ontology languages and tools and are widely used by biologists. However, the absence of a bridge between the two systems is cause for lack of interoperability between biomedical ontologies developed by different systems. This also causes problems for new biology projects that have to pick a technology with a view that choosing one system over the other will restrict them to the tools and existing ontology content provided by that particular system only.

Given the volume and growth of OBO content, OBO Foundry may rightly be called the backbone for biomedical ontology content. Semantic Web, on the other hand, is intended to facilitate search and information integration by providing formally defined semantics, global identifiers and expressive languages for querying ontologies and reasoning on them to infer new knowledge. Integrating the features promised by the Semantic Web with OBO content would provide significant benefit to the biomedical community. One way to provide those features is to create a system that allows back and forth translation of OBO ontologies between the two systems. We have developed such a round-trip between OBO format and the Semantic Web's Web Ontology Language (OWL).

We provide a methodology for organizing a mapping between two systems such as OBO and OWL, and a lossless round-trip mapping between OBO and OWL for ontologies originally developed in OBO. Through collaboration with other people in the OBO community, our work has grown into a community standard, and is now the official mapping supported by the Gene Ontology project and OBO Foundry [23]. Our source code for transformation software is also a part of the Gene Ontology source repository on SourceForge [96].

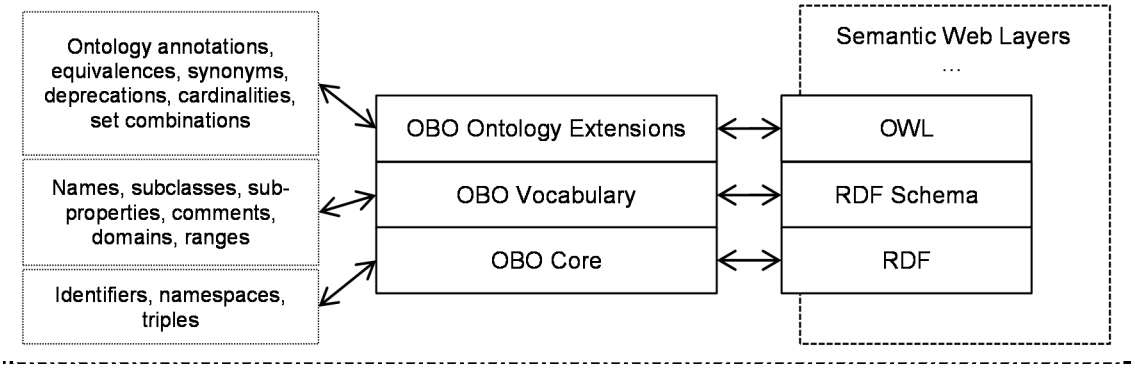


Figure 6.1: A layer cake for OBO (layers for OBO Core, OBO Vocabulary and OBO Ontology Extensions), with some examples and the corresponding layers in the Semantic Web layer cake.

6.1 SYSTEM DESCRIPTION

OBO and Semantic Web Layers

The Semantic Web was envisioned as an expressive hierarchy that is often illustrated as a layer cake [97] (see Figure 2.2). At the beginning of this research it was our conjecture that the precise organization of the hierarchy transcends the Semantic Web and could be used, retroactively, to formalize the structure of other data and concept modeling systems. Thus, as a first step towards the creation of a transformation mechanism between OBO and OWL, we created a layer cake for OBO whose structure

mirrored that of the Semantic Web layer cake. This allowed us to identify straightforward mappings between OBO and OWL as well as the cases that do not match very well. We term this the 'two layer cakes' methodology. This methodology has also been successfully applied towards the transformation of SQL databases into OWL ontologies [73].

OBO Layer Cake

We methodically examined each of the constructs of OBO. We find that most of the OBO format can be decomposed into layers with direct correspondence to the Semantic Web. We call these layers OBO Core, OBO Vocabulary, and OBO Ontology Extensions (see Figure 6.1).

1. **OBO Core:** In OBO, a concept can either be a term (class) or a typedef (relationship type). OBO Core deals with assigning IDs and ID spaces to concepts, and representing relationships as triples.
2. **OBO Vocabulary:** OBO Vocabulary allows annotating concepts with metadata such as names and comments. It also supports describing subclass and sub-property relationship types, as well as the domains and ranges for typedefs.
3. **OBO Ontology Extensions:** In addition to concept-level tags, OBO Ontology Extensions (OBO-OE) layer defines tags for expressing metadata on the entire ontology as well. It also allows defining synonyms and equivalences and supports deprecation of concepts. OBO-OE layer can also express specific properties of OBO terms (e.g. set combinations, disjoints etc.), and typedefs (e.g. transitivity, uniqueness, symmetry, cardinalities).

Table 6.1: Layer cake assignments for OBO constructs.

Layer	List of constructs
OBO Core	`id, idspace, relationship`
OBO Vocabulary	`name, definition, comment, is_a, domain, range`
OBO Ontology Extensions	`format-version, version, date, saved-by, auto-generated-by, namespace, default-namespace, subsetdef, alt_id, relationship, subset, synonym, is_obsolete, is_cyclic, is_transitive, is_symmetric, import, synonymtypedef, intersection_of, union_of, disjoint_from, replaced_by, consider, inverse_of, transitive_over`

Table 6.1 provides assignments of OBO constructs to appropriate layers in the OBO layer cake. Since we mostly have an exact mapping of layers between the two languages (Figure 6.1), deciding which constructs to use for each kind of transformation is simplified. OBO Core tags can be transformed using RDF. OBO Vocabulary tags require using RDF Schema constructs. OBO Ontology Extensions tags require constructs defined in OWL.

Incompatibilities between OBO and OWL

We classify incompatibilities between the two languages into one of the two categories. First, in certain cases, the semantic equivalent of a construct in one language is missing from the other language. Second, sometimes the semantics of constructs in OBO are not sufficiently well-defined to map to a formally defined OWL construct, which forces us to define new vocabulary in OWL in order to allow the lossless transformation.

1. Entities in OWL have globally unique identifiers (URIs). On the other hand, OBO allows local identifiers. Transforming OBO into OWL requires transforming the local identifiers in an OBO ontology into URIs. Also, in

order to make the round-trip possible, it is necessary to extract the local identifier back from the URI.

2. OBO language has the 'subset' construct, which does not have an equivalent construct in OWL. An OBO subset is a collection of terms, and is defined as a part of an ontology. An ontology can contain multiple subsets and each term can be a part of multiple subsets. In order to make the transformation possible, we need to define an OWL construct equivalent to OBO subset, and some relationship concepts to represent terms being in a subset, and a subset being a part of an ontology.
3. There are multiple kinds of synonym tags in OBO, e.g. related, narrow, broad, exact etc. The differences between these constructs are not formally documented. This requires defining new concepts in OWL, which can perhaps be mapped to new or already existing constructs in OWL.

Elements of OBO "missing" in Semantic Web are few, and can still be constructed in OWL. Thus, OBO ontologies may be translated to Semantic Web. However, in order to make the round-trip possible, we find it important to store some ancillary in-formation about the OBO ontology in the OWL file, e.g. a base URI etc., so it can be translated back without any loss of knowledge. It is important to note that even changing a local identifier within the whole knowledgebase is counted as loss of knowledge from the original source, even if the overall structure of the ontology remains intact. The presence of such incompatibilities requires us to make some complex mapping choices explained later.

OBO and Sublanguages of OWL

OWL has three increasingly expressive sublanguages; OWL Lite, OWL DL and OWL Full. Each of these sublanguages extends its simpler predecessor with richer constructs that affect the computational completeness and decidability of the ontology.

Our investigation shows that a major portion of OBO Ontology Extensions maps to OWL Lite and provides similar level of expressiveness. Overall, OBO features are a strict subset of OWL DL. In OBO, the definition of a term or a typedef is rigid and not as expressive as OWL Full. OWL Full allows restrictions to be applied on the language elements themselves [33, 34]. In other words, an OWL Full Class can also be an OWL Full Property and an Instance and vice versa. Such features are not supported in OBO.

Recall, the primary concern is the use of the Semantic Web technology and tools for OBO ontologies. Thus, that OBO is less expressive than OWL is the convenient direction of containment. It does mean that round-trips cannot be supported unless the editing of any OBO ontology while in OWL representation is restricted. We talk about the editing of transformed ontologies while in OWL language in a later section as well.

While transforming OBO ontologies into OWL, we must ensure producing a representation that can be used by description logic based inference engines. One of the intended goals of our transformation is to produce OWL DL, and not OWL Full.

6.3 TRANSFORMATION METADATA AND RULES

In this section, we present some of the rules for the transformation of OBO ontologies into OWL. For more complex transformations we describe the transformations and explain our approach.

In order to facilitate the transformation, we have defined a set of OWL meta-classes that correspond to the vocabulary of OBO tags. Complete listing of mappings between OBO and OWL are available in a Google Spreadsheet [98].

Table 6.2: Some OBO elements (taken from ZFA) and their mappings in OWL.

OBO	**OWL**

`[Typedef]`
`id: part_of`
`name: part of`
`is_transitive: true`
	`<owl:TransitiveProperty`
	`  rdf:about="…#part_of">`
	`  <rdfs:label>part of</rdfs:label>`
	`</owl:TransitiveProperty>`

Example A Simple transformations: name, transitivity	

`[Term]`
`id: ZFA:0000434`
`name: skeletal system`
`is_a: ZFA:0001439`
	`<owl:Class rdf:about="...#ZFA_0000434">`
	`  <rdfs:label>skeletal system</rdfs:label>`
	`  <rdfs:subClassOf`
	`  rdf:resource="...#ZFA_0001439"/>`
	`</owl:Class>`

Example B Transformation of 'is-a'	

`[Term]`
`id:  ZFA:0001439`
`name: anatomical system`
`relationship: part_of ZFA:0001094`
	`<owl:Class rdf:about="…#ZFA_0001439">`
	`  <rdfs:label>anatomical system</rdfs:label>`
	`  <rdfs:subClassOf><owl:Restriction>`
	`  <owl:onProperty`
	`  rdf:resource="…#part_of" />`
	`  <owl:someValuesFrom`
	`  rdf:resource="…#ZFA_0001094" />`
	`  </owl:Restriction></rdfs:subClassOf>`
	`</owl:Class>`

Example C Transformation of a relationship	

`[Term]`
`id: ZFA:0000437`
`name: stomach`
`is_obsolete: true`
	`<owl:Class`
	`  rdf:about="&oboInOwl;ObsoleteClass"/>`
	`  <owl:Class rdf:about="...#ZFA_0000437">`
	`  <rdfs:label>stomach</rdfs:label>`
	`  <rdfs:subClassOf`
	`rdf:resource="&oboInOwl;ObsoleteClass"/>`
	`</owl:Class>`

Example D Transformation of obsolete term	

Simple Transformation Rules

Most of the transformations follow simple rules. For most header and term/typedef tags, there is a one-to-one correspondence between OBO tags and OWL elements, either pre-existing or newly defined. In this section, we list the elements with this kind of simple transformation. Table 6.2 Example A provides some examples.

Header: The set of tag-value pairs at the start of an OBO file, before the definition of the first term or typedef, is the header of the ontology. When translated into OWL language, each of the OBO header tags gets translated into the corresponding OWL markup element. The whole ontology header is contained in the `owl:Ontology` element in the new OWL file, and can appear anywhere within the file, as opposed to the start of file in OBO language.

Terms: A term in OBO is a class in OWL. So a term declaration is translated into an `owl:Class` element and the tags associated with a term are contained within this element. Some tags that have straightforward transformations to OWL elements are:

1. The elements for `name` and `comment` about a term fall into the OBO Vocabulary layer, and are translated into `rdfs:label` and `rdfs:comment` respectively. A `definition` tag is translated into `hasDefinition` annotation property, and is therefore placed in the OBO Ontology Extensions layer.
2. The `is_a` tag in OBO specifies a subclass relationship, and is placed in the OBO Vocabulary layer. It is translated into an `rdfs:subClassOf` element (Table 6.2 Example B).

Typedefs: A typedef in OBO is an object property in OWL. A typedef stanza in an OBO file is translated into an `owl:ObjectProperty` element in OWL. The other information associated with the typedef is expressed as elements nested within this element. Some simple transformations are:

1. OBO typedefs can have associated domains and ranges. These are expressed by `domain` and `range` tags, and are in the OBO Vocabulary layer. These tags are translated into RDF Schema elements `rdfs:domain` and `rdfs:range` respectively.
2. Just like subclasses for terms, a property can be a sub-property to another property. A sub-property relationship is expressed using the `is_a` tag, from OBO Vocabulary layer, in a typedef stanza. This tag is translated into an `rdfs:subPropertyOf` element defined in RDF Schema.
3. Typedefs may be cyclic (`is_cyclic` tag), transitive (`is_transitive` tag) or symmetric (`is_symmetric` tag). These tags fall into the OBO Ontology Extensions layer. The corresponding elements in OWL are annotation property `isCyclic`, and property types `owl:TransitiveProperty` and `owl:SymmetricProperty` respectively. The `isCyclic` property specifies a Boolean value.

Identifiers and ID Spaces

OBO has a local identifier scheme. As OBO evolves, ID spaces have been introduced to allow specifying global identifiers. OBO identifiers have no defined syntax, but they are recommended to be of the form:

"<IDSPACE>:<LOCALID>"

However, existing OBO ontologies may contain flat identifiers, ones that do not mention the ID space. OBO identifiers must be converted to URIs for use in OWL. The rules for converting OBO identifiers to URIs in the current mapping are as follows:

If the OBO header declares an ID space of the form: "`idspace: GO http://www.go.org/owl#`", all OBO identifiers with the prefix `GO:` will be mapped to the provided URI, e.g. "`http://www.go.org/owl#GO_0000001`".

If an OBO ID space prefix does not have a declaration in the header, all identifiers that mention that prefix will be transformed using a default base URI, for example an identifier of the form "`SO:0000001`" will become "`<default-base-uri>SO_0000001`". In case the OBO identifier is flat, e.g. `foo`, the transformation again uses the default base URI to create "`<default-base-uri>UNDEFINED_foo`". Notice that the URI contains "`UNDEFINED_`", which clarifies that the URI should be translated into a flat identifier when translating the OWL version back to OBO.

Recall that OBO Relations Ontology [27] standardizes certain typedefs for use across OBO ontologies. Such typedefs have OBO identifiers prefixed with ID space `OBO_REL`. OBO ontology assumes the presence of this ID space with URI "`http://www.obofoundry.org/ro/ro.owl`" even if it is not explicitly stated. When translated into OWL, an XML namespace `xmlns:oboRel` with the same URI is added to the ontology, and the newly created object property is assigned that namespace. As a result, we ensure that all Relations Ontology constructs are mapped to the same URIs across ontologies.

Relationships

Relationships between OBO terms can be defined using the `relationship` tag. A defined relationship is like a binary predicate and consists of a subject (the term being described in the stanza), a relationship type and an object.

There are multiple kinds of restrictions on relationships that can be expressed using OWL. OBO specifications do not specify any formal semantics of the

`relationship` tag that match a specific relationship type restriction defined in OWL. Therefore, based on the use of relationships in existing ontologies, we selected the appropriate element, the `owl:someValuesFrom` restriction in our mappings. This restriction is similar to the existential quantifier of predicate logic [33, 34]. An example of relationship transformation is shown in Table 6.2 Example C.

Subsets

Terms in an OBO ontology can be organized into subsets. A term can belong to multiple subsets. In order to declare a subset, a value for the tag `subsetdef` is specified in the OBO ontology header. This value consists of a subset ID and a quoted description about the subset. A term can be assigned to a defined subset using the `subset` tag. Multiple `subset` tags are used to assign the term to multiple subsets of the ontology.

When the ontology is translated into OWL, the mapping of subsets is one of the more complex processes. This is due to the fact that subsets do not have a semantic equivalent in OWL. Therefore, we use some OWL features to construct elements that serve as subsets. Subsets fall in the OBO Ontology Extensions in the OBO layer cake.

The local ID assigned to the subset becomes the OWL ID of a subset resource. A subset resource is declared using an `oboInOwl:Subset` element. The `inSubset` annotation is used to assign terms to a subset, and it is expressed within the `owl:Class` element.

Obsolete Content

OBO format supports obsolete content. A term or typedef can be marked as obsolete using the `is_obsolete` tag with a `true` Boolean value. The `is_obsolete` tag is in the OBO Ontology Extensions. Obsolete terms and typedefs are not allowed to have

any relationships with other terms or typedefs, including the subclass and sub-property relationships.

When translated into OWL, an obsolete term becomes a subclass of `oboInOwl:ObsoleteClass` (Table 6.2 Example D). Similarly, an obsolete typedef becomes a sub-property of `oboInOwl:ObsoleteProperty`.

Table 6.3: Results from evaluation of our round-trip transformation on some ontologies.[1, 2]

Ontology	Original OBO	OWL Translation	Round-trip OBO
ZFA	Terms: 2219 Typedefs: 4	Classes: 2219 Object Properties: 4	Terms: 2219 Typedefs: 4
MA	Terms: 2882 Typedefs: 1	Classes: 2882 Object Properties: 1	Terms: 2882 Typedefs: 1
SPD	Terms: 494 Typedefs: 1	Classes: 494 Object Properties: 1	Terms: 494 Typedefs: 1
GO	Terms: 28667 Typedefs: 5	Classes: 28667 Object Properties: 5	Terms: 28667 Typedefs: 5

6.4 IMPLEMENTATION AND EVALUATION

Based on the mapping rules, we have implemented a Java implementation of the transformation. Our implementation is part of the official Gene Ontology project source [96]. Gene Ontology project is an open source project on Sourceforge.net, and is home to the OBO ontology editor OBO-Edit. Our implementation is part of the OBO API that provides data structures for storing OBO ontologies, as well as read and write capabilities

[1] ZFA = Zebrafish Anatomy ontology, MA = Adult Mouse Gross Anatomy ontology, SPD = Spider Ontology, and GO = Gene Ontology.
[2] Class counts do not include obsolete classes, or ancillary information required for round-trips.

for OBO and OWL, among other operations. The source code for our transformation tool is available at [99]. Our mapping tool is also used in Morphster.

Finally, we have deployed our transformation as a web service for general use:

`http://www.cs.utexas.edu/~hamid/oboowl.html`

In the OBO API, we have created `NCBOOboInOWLMetadataMapping` class in the package `org.obo.owl.datamodel.impl`. This class implements the round-trip mapping between OBO and OWL. In order to provide console-based use of the transformation tool, we have created `Obo2Owl` and `Owl2Obo` classes in `org.obo.owl.test` package.

In order to evaluate the OWL output of our implementation, we have tested our tool on Gene Ontology, Zebrafish Anatomical Ontology, Spider Ontology and Adult Mouse Gross Anatomy, obtained from NCBO BioPortal. After transformation of these ontologies into OWL, we have successfully loaded the OWL files into Protégé [12], an ontology development tool for the Semantic Web. Using the 'summary' feature of Protégé, we have compared the overall class and object property count with the term and typedef count obtained for the original OBO file, using OBO-Edit's 'extended information' feature The results of the comparison (Table 6.3) show equal values for both versions of the ontologies. Similarly, for testing the round-trip, we compared the original OBO file with the round-trip version, again using OBO-Edit's feature. Our evaluation showed that the two OBO ontologies had the same term and typedef counts (Table 6.3).

6.5 IMPLICATIONS OF TRANSFORMATION

OBO Semantics by Transformation

The transformation system has the additional effect of formalizing the semantics of the OBO language. The semantics of OBO are operationally defined by means of GO and the software systems that support GO. The semantics of OWL have been formally

defined using model theory [32, 100]. Though we have not written it out, a formal document specifying OBO semantics can be created, mechanically, from the contents of this paper and the OWL semantics documents. The contents of that document would comprise an enumeration of the pairwise mapping of constructs between the two languages, restating, in each mapping, the semantics stated for the involved OWL construct.

Table 6.4: Identifying the semantics for OBO constructs using OWL mappings.[3]

Description	**OBO**	**OWL**	**Semantics**
x is a subclass of y	`is_a`	`rdfs:subClassOf`	$CEXT(x) \subseteq CEXT(y)$
x is a sub-property of y	`is_a`	`rdfs:subPropertyOf`	$EXT(x) \subseteq EXT(y)$
x is domain of property y	`domain`	`rdfs:domain`	$< z, w > \in EXT(y)$ $\Rightarrow z \in CEXT(x)$
x is disjoint from y	`disjoint_ from`	`owl:disjointWith`	$CEXT(x) \cap CEXT(y) = \{\}$
p is a transitive property	`is_ transitive`	`owl:Transitive Property`	$< x, y >, < y, z > \in EXT(p)$ $\Rightarrow < x, z > \in EXT(p)$

In Table 6.4, we present a few examples where our transformation mapping could provide formal semantics for OBO constructs, taken directly from OWL semantics specifications.

While the identification is straightforward in these cases, in some other situations it is not very clear. Finding the semantics of relationships in OBO is one such case. As mentioned earlier, OBO specifications do not provide the semantics of the construct used to specify relationships between two terms using a typedef. Therefore, it is hard to decide

[3] CEXT(c): the set of instances of class c; EXT(p): the set of pairs <x,y> related by property p

which of the available relationship constraints in OWL (`owl:allValuesFrom`, `owl:someValuesFrom`) to use, the former being similar to a universal quantifier, and the latter to an existential quantifier. In our transformations, we use `owl:someValuesFrom`, since already built ontologies show examples of use of OBO relationship construct in a compatible way. We recommend that the semantics of relationships should always be defined to match the `owl:someValuesFrom` restriction.

Other OBO tags that do not clearly match with OWL elements, such as synonyms and subsets, as well as the semantics for the `is_obsolete` tag also present a more significant challenge in the identification of semantics.

Updating OBO Ontologies in OWL

The set of constructs for ontology representation provided by OWL is considerably larger than the set of constructs provided by OBO. Therefore, in order to allow round-trip transformations on OBO ontologies, it is important to restrict the editing of such ontologies per some guidelines while they are being represented in OWL.

Our transformation mappings essentially provide a subset of OWL elements that may be used for adding or updating contents of the ontology. We refer to this subset of OWL as *OWL-Bio*, for biomedical ontologies hosted by OBO. Since our mapping produces OWL DL, OWL-Bio is a subset of OWL DL by definition.

Compared to the general use of OWL, there are two key points to keep in mind:

1. To create relationships, use `owl:someValuesFrom` relations. Since OBO does not have a corresponding relationship mechanism for `owl:allValuesFrom`, it is not a part of OWL-Bio.

2. Obsolescence of terms in the ontology should be done using the obsolete elements `oboInOwl:ObsoleteClass` and `oboInOwl:ObsoleteProperty` instead of built in deprecation elements in OWL.

6.6 STANDARDIZATION OF MAPPINGS AND RELATED WORK

We have collaborated with Stuart Aitken, Chris Mungall, Dilvan Moreira and Nigam Shah to produce a standardized mapping. Each of our collaborators, as well as Mikel Egana, Erick Antezana, and LexBio group at Mayo Clinic, contributed unpublished independent effort at creating a transformation system. The results of these efforts are documented in our spreadsheet. No single effort survived in its entirety in the common mapping [98]. Our methodology and mapping choices, however, were fully adopted. The difference between our original work and the standardized mapping is mainly that of different strings (names) for mapping annotations.

Another independent and important effort was that of Golbreich et al. [101, 102] (hereafter Golbreich). Note that this group did not participate in the community effort to standardize the mapping. Golbreich developed a BNF grammar for OBO syntax, as well as a mapping between OBO and OWL 1.1 (now known as OWL 2). The differences between the Golbreich work and the common mapping effort presented in this paper comprise a difference of methodology and practical focus. Golbreich's work laid out valuable syntactic groundwork to formalize the semantics of a large subset of OBO. Much like most of the other first efforts, a complete transformation system was not specified. This particular effort deferred resolving OBO annotations, synonyms, subsets, and deprecation tags. Golbreich's work also did not address the mapping of local identifiers in OBO into global identifiers. The transformations that are specified by Golbreich are largely consistent with the common mappings.

6.7 CONCLUSION

Building ontologies is not a new idea for the biology community, and precedes the development of the Semantic Web. While ontologies are a central part of the architecture of the Semantic Web, the Semantic Web vision includes a broad range of technologies from the Artificial Intelligence field, such as inference and querying mechanisms, as well as anticipating additional elements of distributed computation, such as global identifiers and the use of XML and HTTP as middleware. OBO, on the other hand, has appropriate tool support for building ontologies and hosts a number of important biomedical ontologies. Hence the OBO community has the biggest and most immediate need for the features being developed by the Semantic Web community.

We have standardized the mapping between the two systems to allow the OBO community to utilize the tool base developed for the Semantic Web world, and will also standardize the transformation across OBO tools. We have indirectly formalized the semantics of OBO by creating a round-trip transformation between OBO and OWL. We have also implemented our transformation tool in Java and it is available as a part of open source Gene Ontology project, and also as a web service. We believe our work is an important step towards building interoperable knowledge bases be-tween OBO and the Semantic Web communities.

The implications of our work in providing semantics to OBO as well as in defining a "biomedical flavor" for OWL strongly suggest the use of this mapping as a potential bridge between the OBO and the Semantic Web worlds. Our ability to make round-trips between OBO and OWL-Bio could enable fluid interconnections between the two worlds. While OWL-Bio could serve as a common ground for the two languages, our round-trip tool could be used as a validator for ontologies updated in OWL.

A key difference between the OBO community and the Semantic Web is the methodology for content development across ontologies. The Semantic Web has adapted a completely distributed development mechanism for ontologies that may be integrated using URIs. On the other hand, the OBO community uses a hybrid of centralized and distributed development. While the users of OBO develop ontologies independently, the OBO foundry has the goal of creating a suite of orthogonal interoperable reference ontologies, such as the Relations Ontology, in the biomedical domain. Our transformation system enriches the Semantic Web by providing this this addition-al structured ontology content and the access to the wealth of data annotated using it.

Chapter 7

Conclusions and Future Work

Ontologies have been used to model biological knowledge for centuries under other names, going at least as far back as the Linnaean taxonomy and Charles Darwin's sketches of evolutionary trees. More recently, the development of ontology systems such as the Semantic Web and the Open Biomedical Ontology (OBO) Foundry have made ontologies the tool of choice for capturing and publishing knowledge in systematic biology.

At the highest level, this dissertation represents an attempt to demonstrate the strengths and benefits of ontologies in biology. As the use of ontologies grows in the community, it is crucial to understand the utility of ontologies as a tool for capturing domain knowledge. Most biologists understand only the *knowledge representation* aspect of ontologies and hence see them as another way of publishing their data and results to the world. Our primary goal is to demonstrate the value of the other aspects of ontologies, *knowledge inference* and *knowledge integration*, in the context of scientific research in systematic biology. Each of the problems we have addressed in this dissertation lead us towards this fundamental goal.

Our work represents a beginning, and there is significant progress to be made to help biologists truly utilize ontologies.

- *Capturing knowledge in ontological form*

  A significant part of our work in this dissertation as well as on the Morphster project as a whole focuses on enabling our users, biologists, to capture their scientific data, hypotheses and results in ontological form. While the Morphster meta-model and use cases such as *stack-of-photos* represent progress in this area, there is need for better domain-specific tools that integrate ontology building at each stage in the workflow of a phylogenetic study. A particular case in this regard is that of character statements – capturing biological facts and scientific observations as ontological concepts. Our taxonomy for types of characters represents a major step forward in explaining and capturing character statements. However, it is focused towards building anatomy ontologies. A short-term research direction is to expand the taxonomy to a broader set of character types and investigate its applications to a broader set of biological inference problems.

- *Bringing legacy knowledge from the literature into the knowledgebase*

  In addition to developing tool support for new knowledge acquisition in biology, there is a great need for importing the existing literature and legacy data such as character matrices and trees into the ontology domain. Scientists conducting new studies are always interested in finding legacy work on similar anatomical regions or organisms, and as the ontological content grows, there is need to connect the legacy data to it. One of the ways to achieve this goal is to use natural language processing techniques to identify character statements and matrix data in the literature, and extract the frames for appropriate concepts based on our meta-model. This data is typically

represented in a stylized natural language. Starting from the frames for the character types in our taxonomy, more character types and their frames can be identified to capture a broader set of characters in this stylized language.

- *Capturing and integrating molecular biology with morphology*

  Connecting results in molecular biology with morphology is a significant problem in evolutionary biology. An increasing number of molecular biologists are working with ontologies. Investigations into the use of ontologies to connect the results from molecular biology with morphology may provide interesting results.

- *Integrating biological databases through ontologies*

  There are a large number of independent relational databases managed by biologists for recording their data. Schemas of these databases vary greatly. Ontologies, or federated schemas, have been used in other domains for schema matching. Community developed ontologies may be needed to provide a solution to interconnect these biology databases.

Appendix A

Source Code of Jess Rules

```
(import edu.utexas.cs.morphology.model.*)

(deftemplate Tree (declare (from-class Tree) (include-variables TRUE)))
(deftemplate Taxon (declare (from-class Taxon) (include-variables TRUE)))

(deftemplate NA (declare (from-class NA) (include-variables TRUE)))
(deftemplate Rel (declare (from-class Rel) (include-variables TRUE)))
(deftemplate NAEntity (declare (from-class NAEntity) (include-variables TRUE)))
(deftemplate NARel (declare (from-class NARel) (include-variables TRUE)))
(deftemplate Quality (declare (from-class Quality) (include-variables TRUE)))

(deftemplate Char (declare (from-class Char) (include-variables TRUE)))
(deftemplate State (declare (from-class State) (include-variables TRUE)))

(deftemplate CharNE extends Char (declare (from-class CharNE) (include-
variables TRUE)))
(deftemplate CharTR extends Char (declare (from-class CharTR) (include-
variables TRUE)))

(deftemplate CharCL extends CharTR (declare (from-class CharCL) (include-
variables TRUE)))
(deftemplate CharME extends CharTR (declare (from-class CharME) (include-
variables TRUE)))
(deftemplate CharRE extends CharTR (declare (from-class CharRE) (include-
variables TRUE)))

(deftemplate CharBR extends CharRE (declare (from-class CharBR) (include-
variables TRUE)))
(deftemplate CharRO extends CharRE (declare (from-class CharRO) (include-
variables TRUE)))
(deftemplate CharCO extends CharRE (declare (from-class CharCO) (include-
variables TRUE)))
(deftemplate CharSE extends CharRE (declare (from-class CharSE) (include-
variables TRUE)))

(deftemplate StateEL extends State (declare (from-class StateEL) (include-
variables TRUE)))
(deftemplate StateELElement (slot sid) (slot entity))

(deftemplate StateNU extends State (declare (from-class StateNU) (include-
variables TRUE)))
```

```
(deftemplate StateE extends State (declare (from-class StateE) (include-
variables TRUE)))

(deftemplate Cell (declare (from-class Cell) (include-variables TRUE)))

(deftemplate StateChange (slot character) (slot fromState) (slot toState))
(deftemplate Params (slot model) (slot target) (slot direction) (slot baseUri))

(defrule rule-01-effect-of-neomorphic-character
    "State change from absent to present in a neomorphic character when the
parent is present"
    (Params (target ?tgt) (baseUri ?baseuri))
    (Taxon (id ?tgt))
    (NA (taxon ?tgt) (id ?naid))
    (CharNE (id ?chid) (entity ?nee) (vicinity ?nev) (relation ?ner))
    (StateChange (character ?chid) (fromState ?fsid) (toState ?tsid))
    (State (id ?fsid) (name "absent"))
    (State (id ?tsid) (name "present"))
    (NAEntity (id ?vid) (na ?naid))
    (test (eq (str-cat ?baseuri ?tgt "/" ?nev) ?vid))
    =>
    (bind ?newid (str-cat ?baseuri ?tgt "/" ?nee))
    (bind ?eobj (new edu.utexas.cs.morphology.model.NAEntity  ?newid ?nee
?naid))
    (add ?eobj)
    (bind ?robj (new edu.utexas.cs.morphology.model.NARel ?vid ?newid ?ner
?naid))
    (add ?robj)
    ;(assert (NAEntity (id ?newid) (name ?nee) (na ?naid))
    ;    (NARel (na ?naid) (child ?newid) (relation "part_of") (parent ?vid)))
    (printout t "*** NE :: (01) + [E:" ?nee "] [R:" ?nee " " ?ner " " ?nev "]
==> " ?chid  crlf))

(defrule rule-02-effect-of-neomorphic-character
    "State change from absent to present in a neomorphic character when the
parent is missing"
    (Params (target ?tgt) (baseUri ?baseuri))
    (Taxon (id ?tgt))
    (NA (taxon ?tgt) (id ?naid))
    (CharNE (id ?chid) (entity ?nee) (vicinity ?nev))
    (StateChange (character ?chid) (fromState ?fsid) (toState ?tsid))
    (State (id ?fsid) (name "absent"))
    (State (id ?tsid) (name "present"))
    (forall (NAEntity (na ?naid) (id ?vid))
        (test (neq (str-cat ?baseuri ?tgt "/" ?nev) ?vid)))
    =>
    (bind ?newid (str-cat ?baseuri ?tgt "/" ?nee))
    (bind ?wholeid (str-cat ?baseuri ?tgt "/whole"))
    (bind ?eobj (new edu.utexas.cs.morphology.model.NAEntity  ?newid ?nee
?naid))
    (add ?eobj)
    (bind ?robj (new edu.utexas.cs.morphology.model.NARel ?wholeid ?newid
"part_of" ?naid))
    (add ?robj)
    ;(assert (NAEntity (id ?newid) (name ?nee) (na ?naid))
    ;    (NARel (na ?naid) (child ?newid) (relation "part_of") (parent
?wholeid)))
```

```
    (printout t "*** NE :: (02) + [E:" ?nee "] [R:" ?nee " part_of whole] ==> "
?chid  crlf))

(defrule rule-03-effect-of-neomorphic-character
    "State change from present to absent in a neomorphic character"
    (Params (target ?tgt) (baseUri ?baseuri))
    (CharNE (id ?chid) (entity ?nee))
    (StateChange (character ?chid) (fromState ?fsid) (toState ?tsid))
    (State (id ?fsid) (name "present"))
    (State (id ?tsid) (name "absent"))
    ?theentity <- (NAEntity (id ?eid))
    (test (eq (str-cat ?baseuri ?tgt "/" ?nee) ?eid))
    =>
    (retract ?theentity)
    (printout t "*** NE :: (03) - [E:" ?nee "] ==> " ?chid crlf))

(defrule rule-04-effect-of-binaryrelationship-character
    "State change from yes to no in a binaryrelationship character"
    (Params (target ?tgt) (baseUri ?baseuri))
    (CharBR (id ?chid) (child ?ce) (relation ?rid) (parent ?pe))
    (StateChange (character ?chid) (fromState ?fsid) (toState ?tsid))
    (State (id ?fsid) (name "yes"))
    (State (id ?tsid) (name "no"))
    ?therelation <- (NARel (child ?cid) (relation ?rid) (parent ?pid))
    (test (eq (str-cat ?baseuri ?tgt "/" ?ce) ?cid))
    (test (eq (str-cat ?baseuri ?tgt "/" ?pe) ?pid))
    =>
    (retract ?therelation)
    (printout t "*** BR :: (04) - [R:" ?ce " " ?rid " " ?pe "] ==> " ?chid
crlf))

(defrule rule-05-effect-of-binaryrelationship-character
    "State change from no to yes in a binaryrelationship character"
    (Params (target ?tgt) (baseUri ?baseuri))
    (Taxon (id ?tgt))
    (NA (taxon ?tgt) (id ?naid))
    (CharBR (id ?chid) (child ?ce) (relation ?rid) (parent ?pe))
    (StateChange (character ?chid) (fromState ?fsid) (toState ?tsid))
    (State (id ?fsid) (name "no"))
    (State (id ?tsid) (name "yes"))
    (forall (NARel (na ?naid) (child ?cid) (relation ?rid) (parent ?pid))
        (or (test (neq (str-cat ?baseuri ?tgt "/" ?ce) ?cid))
            (test (neq (str-cat ?baseuri ?tgt "/" ?pe) ?pid))))
    =>
    (bind ?newcid (str-cat ?baseuri ?tgt "/" ?ce))
    (bind ?newpid (str-cat ?baseuri ?tgt "/" ?pe))
    (bind ?robj (new edu.utexas.cs.morphology.model.NARel ?newpid ?newcid ?rid
?naid))
    (add ?robj)
    ;(assert (NARel (na ?naid) (child ?newcid) (relation ?rid) (parent
?newpid)))
    (printout t "*** BR :: (05) + [R:" ?ce " " ?rid " " ?pe "] ==> " ?chid
crlf))

(defrule rule-06-effect-of-classifying-character
    "State change from ? to some other quality in a classifying character"
    (Params (target ?tgt) (baseUri ?baseuri))
    (Taxon (id ?tgt))
```

```
    (NA (taxon ?tgt) (id ?naid))
    (CharCL (id ?chid) (entity ?cle))
    (StateChange (character ?chid) (fromState "?") (toState ?tsid))
    (State (id ?tsid) (name ?tsname))
    =>
    (bind ?parenteid (str-cat ?baseuri ?tgt "/" ?cle))
    (bind ?neweid (str-cat ?baseuri ?tgt "/" ?tsname "-" ?cle))
    (bind ?newename (str-cat ?tsname "-" ?cle))
    (bind ?eobj (new edu.utexas.cs.morphology.model.NAEntity  ?neweid ?newename
?naid))
    (add ?eobj)
    (bind ?robj (new edu.utexas.cs.morphology.model.NARel ?parenteid ?neweid
"is_a" ?naid))
    (add ?robj)
    ;(assert (NAEntity (id ?neweid) (name ?newename) (na ?naid))
    ;    (NARel (na ?naid) (child ?neweid) (relation "is_a") (parent
?parenteid)))
    (printout t "*** CL :: (06) + [E:" ?tsname "-" ?cle "] [R:" ?tsname "-"
?cle " is_a " ?cle "] ==> " ?chid crlf))

(defrule rule-07-effect-of-classifying-character
    "State change from a quality to ? in a classifying character"
    (Params (target ?tgt) (baseUri ?baseuri))
    (Taxon (id ?tgt))
    (NA (taxon ?tgt) (id ?naid))
    (CharCL (id ?chid) (entity ?cle))
    (StateChange (character ?chid) (fromState ?fsid) (toState "?"))
    (State (id ?fsid) (name ?fsname))
    ?theentity <- (NAEntity (id ?neweid) (name ?newename) (na ?naid))
    ?therelation <- (NARel (na ?naid) (child ?neweid) (relation "is_a") (parent
?parenteid))
    (test (eq (str-cat ?baseuri ?tgt "/" ?cle) ?parenteid))
    (test (eq (str-cat ?baseuri ?tgt "/" ?fsname "-" ?cle) ?neweid))
    (test (eq (str-cat ?fsname "-" ?cle) ?newename))
    =>
    (retract ?therelation
        ?theentity)
    (printout t "*** CL :: (07) - [E:" ?newename "] [R:" ?newename " is_a "
?cle "] ==> " ?chid crlf))

(defrule rule-08-effect-of-classifying-character
    "State change from a quality to another quality in a classifying character,
neither state is ?"
    (Params (target ?tgt) (baseUri ?baseuri))
    (Taxon (id ?tgt))
    (NA (taxon ?tgt) (id ?naid))
    (CharCL (id ?chid) (entity ?cle))
    (StateChange (character ?chid) (fromState ?fsid) (toState ?tsid))
    (not (or (test (eq ?fsid "?")) (test (eq ?tsid "?"))))
    (State (id ?tsid) (name ?tsname))
    (State (id ?fsid) (name ?fsname))
    ?oldentity <- (NAEntity (id ?oldeid) (name ?oldename) (na ?naid))
    ?oldrelation <- (NARel (na ?naid) (child ?oldeid) (relation "is_a") (parent
?parenteid))
    (test (eq (str-cat ?baseuri ?tgt "/" ?cle) ?parenteid))
    (test (eq (str-cat ?baseuri ?tgt "/" ?fsname "-" ?cle) ?oldeid))
    (test (eq (str-cat ?fsname "-" ?cle) ?oldename))
    =>
```

```
    (bind ?neweid (str-cat ?baseuri ?tgt "/" ?tsname "-" ?cle))
    (bind ?newename (str-cat ?tsname "-" ?cle))
    (bind ?eobj (new edu.utexas.cs.morphology.model.NAEntity  ?neweid ?newename
?naid))
    (add ?eobj)
    (bind ?robj (new edu.utexas.cs.morphology.model.NARel ?parenteid ?neweid
"is_a" ?naid))
    (add ?robj)
    ;(assert (NAEntity (id ?neweid) (name ?newename) (na ?naid))
    ;    (NARel (na ?naid) (child ?neweid) (relation "is_a") (parent
?parenteid)))
    (retract ?oldrelation ?oldentity)
    (printout t "*** CL :: (08) - [E:" ?oldename "] [R:" ?oldename " is_a "
?cle "] ==> " ?chid crlf
        "               + [E:" ?newename "] [R:" ?newename " is_a " ?cle "] ==>
" ?chid crlf))

(defrule rule-09-effect-of-compositional-character
    "Found an entity that should not be a part of the parent entity of a
compositional character"
    (Params (target ?tgt) (baseUri ?baseuri))
    (Taxon (id ?tgt))
    (NA (taxon ?tgt) (id ?naid))
    (CharCO (id ?chid) (entity ?coe))
    (NAEntity (id ?coeid) (name ?coe) (na ?naid))
    (StateChange (character ?chid) (toState ?tsid))
    (StateEL (id ?tsid))
    ?therelation <- (NARel (na ?naid) (child ?cheid) (relation "part_of")
(parent ?coeid))
    (NAEntity (id ?cheid) (name ?chename))
    (not (StateELElement (sid ?tsid) (entity ?chename)))
    =>
    (retract ?therelation)
    (printout t "*** CO :: (09) - [R:" ?chename " part_of " ?coe "] ==> " ?chid
crlf))

(defrule rule-10-effect-of-compositional-character
    "Found a new entity that should be added as a part of the parent entity of
a compositional character"
    (Params (target ?tgt) (baseUri ?baseuri))
    (Taxon (id ?tgt))
    (NA (taxon ?tgt) (id ?naid))
    (CharCO (id ?chid) (entity ?coe))
    (NAEntity (id ?coeid) (name ?coe) (na ?naid))
    (StateChange (character ?chid) (toState ?tsid))
    (StateEL (id ?tsid))
    (StateELElement (sid ?tsid) (entity ?chename))
    (not (NAEntity (name ?chename) (na ?naid)))
    =>
    (bind ?cheid (str-cat ?baseuri ?tgt "/" ?chename))
    (bind ?eobj (new edu.utexas.cs.morphology.model.NAEntity  ?cheid ?chename
?naid))
    (add ?eobj)
    (bind ?robj (new edu.utexas.cs.morphology.model.NARel ?coeid ?cheid
"part_of" ?naid))
    (add ?robj)
    ;(assert (NAEntity (na ?naid) (id ?cheid) (name ?chename))
```

```
    ;    (NARel (na ?naid) (child ?cheid) (relation "part_of") (parent
?coeid)))
    (printout t "*** CO :: (10) + [E:" ?chename "] [R:" ?chename " part_of "
?coe "] ==> " ?chid crlf))

(defrule rule-11-effect-of-compositional-character
    "Found an existing entity that should be added as a part of the parent
entity of a compositional character"
    (Params (target ?tgt) (baseUri ?baseuri))
    (Taxon (id ?tgt))
    (NA (taxon ?tgt) (id ?naid))
    (CharCO (id ?chid) (entity ?coe))
    (NAEntity (id ?coeid) (name ?coe) (na ?naid))
    (StateChange (character ?chid) (toState ?tsid))
    (StateEL (id ?tsid))
    (StateELElement (sid ?tsid) (entity ?chename))
    (NAEntity (name ?chename) (na ?naid) (id ?cheid))
    (not (NARel (na ?naid) (child ?cheid) (relation "part_of") (parent
?coeid)))
    =>
    (bind ?robj (new edu.utexas.cs.morphology.model.NARel ?coeid ?cheid
"part_of" ?naid))
    (add ?robj)
    ;(assert (NARel (na ?naid) (child ?cheid) (relation "part_of") (parent
?coeid)))
    (printout t "*** CO :: (11) + [R:" ?chename " part_of " ?coe "] ==> " ?chid
crlf))

(defrule rule-12-effect-of-meristic-character
    "Effect of meristic character when state changes to 1 and there are
entities to be removed. State cannot be zero for this character."
    (Params (target ?tgt) (baseUri ?baseuri))
    (Taxon (id ?tgt))
    (NA (taxon ?tgt) (id ?naid))
    (CharME (id ?chid) (entity ?mee))
    (NAEntity (id ?meeid) (name ?mee) (na ?naid))
    (StateChange (character ?chid) (toState ?tsid))
    (StateNU (id ?tsid) (value 1))
    ?extrarel <- (NARel (na ?naid) (parent ?meeid) (relation "is_a") (child
?meechid))
    (test (?meechid startsWith (str-cat ?meeid "#")))
    ?extraentity <- (NAEntity (na ?naid) (id ?meechid) (name ?meech))
    =>
    (retract ?extraentity ?extrarel)
    (printout t "*** ME :: (12) - [E:" ?meech "] [R:" ?meech " is_a " ?mee "]
==> " ?chid crlf))

(defrule rule-13-effect-of-meristic-character
    "Effect of meristic character when state changes from low to high value."
    (Params (target ?tgt) (baseUri ?baseuri))
    (Taxon (id ?tgt))
    (NA (taxon ?tgt) (id ?naid))
    (CharME (id ?chid) (entity ?mee))
    (NAEntity (id ?meeid) (name ?mee) (na ?naid))
    (StateChange (character ?chid) (toState ?tsid) (fromState ?fsid))
    (StateNU (id ?tsid) (value ?tval))
    (StateNU (id ?fsid) (value ?fval))
    (test (> ?tval ?fval))
```

```
    (test (>= ?fval 1))
    =>
    (printout t "*** ME :: (13) ")
    (bind ?white "")
    (if (= ?fval 1) then
        (bind ?no1id (str-cat ?meeid "#1"))
        (bind ?no1name (str-cat ?mee "#1"))
       (bind ?eobj (new edu.utexas.cs.morphology.model.NAEntity  ?no1id ?no1name
?naid))
       (add ?eobj)
       (bind ?robj (new edu.utexas.cs.morphology.model.NARel ?meeid ?no1id
"is_a" ?naid))
       (add ?robj)
        ;(assert (NAEntity (na ?naid) (id ?no1id) (name ?no1name))
        ;    (NARel (na ?naid) (child ?no1id) (relation "is_a") (parent
?meeid)))
        (printout t "+ [E:" ?no1name "] [R:" ?no1name " is_a " ?mee "] ==> "
?chid crlf)
        (bind ?white "               "))
    (for (bind ?i (+ 1 ?fval)) (<= ?i ?tval) (++ ?i)
        (bind ?nextid (str-cat ?meeid "#" ?i))
        (bind ?nextname (str-cat ?mee "#" ?i))
       (bind ?eobj (new edu.utexas.cs.morphology.model.NAEntity  ?nextid
?nextname ?naid))
       (add ?eobj)
       (bind ?robj (new edu.utexas.cs.morphology.model.NARel ?meeid ?nextid
"is_a" ?naid))
       (add ?robj)
        ;(assert (NAEntity (na ?naid) (id ?nextid) (name ?nextname))
        ;    (NARel (na ?naid) (child ?nextid) (relation "is_a") (parent
?meeid)))
        (printout t ?white "+ [E:" ?nextname "] [R:" ?nextname " is_a " ?mee "]
==> " ?chid crlf)
        (bind ?white "               "))
    )

(deffunction to-integer (?str)
    (bind ?hashindex (?str lastIndexOf "#"))
    (bind ?piece (?str substring (++ ?hashindex)))
    (bind ?val (call java.lang.Integer parseInt ?piece))
    (return ?val))

(defrule rule-14-effect-of-meristic-character
    "Effect of meristic character when state changes from high to low value
greater than 1."
    (Params (target ?tgt) (baseUri ?baseuri))
    (Taxon (id ?tgt))
    (NA (taxon ?tgt) (id ?naid))
    (CharME (id ?chid) (entity ?mee))
    (NAEntity (id ?meeid) (name ?mee) (na ?naid))
    (StateChange (character ?chid) (toState ?tsid) (fromState ?fsid))
    (StateNU (id ?tsid) (value ?tval))
    (StateNU (id ?fsid) (value ?fval))
    (test (< ?tval ?fval))
    (test (> ?tval 1))
    ?extrarel <- (NARel (na ?naid) (parent ?meeid) (relation "is_a") (child
?meechid))
    (test (?meechid startsWith (str-cat ?meeid "#")))
```

```
    ?extraentity <- (NAEntity (na ?naid) (id ?meechid) (name ?meech))
    (test (> (to-integer ?meechid) ?tval))
    =>
    (retract ?extraentity ?extrarel)
    (printout t "*** ME :: (14) - [E:" ?meech "] [R:" ?meech " is_a " ?mee "]
==> " ?chid crlf))

(defrule rule-15-effect-of-relationshipoptions-character
    "Effect of a relationshipoptions character"
    (Params (target ?tgt) (baseUri ?baseuri))
    (Taxon (id ?tgt))
    (NA (taxon ?tgt) (id ?naid))
    (CharRO (id ?chid) (child ?ce) (relation ?rid))
    (NAEntity (na ?naid) (name ?ce) (id ?ceid))
    (StateChange (character ?chid) (toState ?tsid))
    (StateE (id ?tsid) (entity ?tse))
    (NAEntity (na ?naid) (name ?tse) (id ?tseid))
    =>
    (bind ?robj (new edu.utexas.cs.morphology.model.NARel ?tseid ?ceid ?rid
?naid))
    (add ?robj)
    ;(assert (NARel (na ?naid) (parent ?tseid) (relation ?rid) (child ?ceid)))
    (printout t "*** RO :: (15) + [R:" ?ce " " ?rid " " ?tse "] ==> " ?chid
crlf))

(defrule rule-16-effect-of-relationshipoptions-character
    "Effect of a relationshipoptions character"
    (Params (target ?tgt) (baseUri ?baseuri))
    (Taxon (id ?tgt))
    (NA (taxon ?tgt) (id ?naid))
    (CharRO (id ?chid) (child ?ce) (relation ?rid))
    (NAEntity (na ?naid) (name ?ce) (id ?ceid))
    (StateChange (character ?chid) (fromState ?fsid))
    (StateE (id ?fsid) (entity ?fse))
    (NAEntity (na ?naid) (name ?fse) (id ?fseid))
    ?extrarel <- (NARel (na ?naid) (parent ?fseid) (relation ?rid) (child
?ceid))
    =>
    (retract ?extrarel)
    (printout t "*** RO :: (16) - [R:" ?ce " " ?rid " " ?fse "] ==> " ?chid
crlf))
```

Appendix B

Input Data Files for Plants Test Case

Plants tree file

```
embryophyte land plant
fern fern
spermatophyte seed plant or phanerogam
gymnosperm naked-seed plant
angiosperm flowering plant
conifer cone bearing plant
cycad cycad
eudicot eudicot
monocot monocot
#
embryophyte fern spermatophyte
spermatophyte angiosperm gymnosperm
gymnosperm conifer cycad
angiosperm eudicot monocot
```

Plants ontology file

```
whole
seed
leaf
simple-leaf
stem
petal
flower
inflorescence
cotyledon
fruit
root
root-cap
root-cortex
exodermis
root-endodermis
shoot
```

```
bud
floral-bract
androecium
stamen
gynoecium
carpel
perianth
calyx
sepal
corolla
nectary
phyllome
compound-leaf
leaflet
shoot-apex
shoot-internode
shoot-node
tuber
embryo
seedling
#
root part_of whole
root-cap part_of root
root-cortex part_of root
exodermis part_of root-cortex
root-endodermis part_of root-cortex
shoot part_of whole
bud part_of shoot
inflorescence part_of shoot
floral-bract part_of inflorescence
flower part_of inflorescence
androecium part_of flower
stamen part_of androecium
gynoecium part_of flower
carpel part_of gynoecium
perianth part_of flower
calyx part_of perianth
sepal part_of calyx
corolla part_of perianth
petal part_of corolla
nectary part_of shoot
phyllome part_of shoot
leaf is_a phyllome
simple-leaf is_a leaf
compound-leaf is_a leaf
leaflet part_of compound-leaf
shoot-apex part_of shoot
shoot-internode part_of shoot
shoot-node part_of shoot
stem part_of shoot
tuber part_of shoot
seed part_of whole
embryo part_of seed
seedling part_of embryo
cotyledon part_of seedling
fruit part_of whole
seed contained_in fruit
```

Plants matrix file

```
C1 NE "produces seeds" seed part_of whole
C2 CL "leaf structure" leaf
C3 NE "stem has wood" wood part_of stem
C4 ME "merosity of flower petals" petal
C5 NE "has flowers" flower part_of inflorescence
C6 ME "number of cotyledons" cotyledon
C7 CL "type of wood" wood
C8 NE "produces fruit" fruit part_of whole
C9 RO "seed contained in" seed contained_in
C10 NE "produces cones" cone part_of whole
#
S11 "absent" C1
S12 "present" C1
S21 "simple" C2
S22 "compound" C2
S31 "present" C3
S32 "absent" C3
S41 "3" C4
S42 "4" C4
S51 "absent" C5
S52 "present" C5
S61 "1" C6
S62 "2" C6
S71 "hardwood" C7
S72 "softwood" C7
S81 "present" C8
S82 "absent" C8
S91 "fruit" C9
S92 "cone" C9
S101 "present" C10
S102 "absent" C10
#
angiosperm S12 S21 S32 ? S52 S61 ? S81 S91 S102
embryophyte S11 ? S32 ? S51 S62 ? S82 ? ?
spermatophyte S12 ? S32 ? S51 S62 ? S82 ? ?
gymnosperm S12 ? S31 ? S51 S62 ? S82 ? S102
conifer S12 ? S31 ? S51 S62 S72 S82 S92 S101
fern S11 S22 S32 ? S51 S62 ? S82 ? ?
cycad S12 ? S31 ? S51 S62 ? S82 ? S102
eudicot S12 ? S31 S42 S52 S62 S71 S81 S91 S102
monocot S12 ? S32 S41 S52 S61 ? S81 S91 S102
```

Appendix C

CTOL Matrix Worksheet

List of characters and their signatures: (Each line starts with the ID of the character, followed by its type, text and frame elements. Character states are listed later).

```
C01- NE "Basihyal element presence" basihyal-element part_of ventral-hyoid-arch
C07- NE "Basibranchial 1-3 toothplate presence" basibranchial-1-3-toothplate
  part_of gill-arch-1-5-skeleton
C08A NE "Basibranchial 1 element presence" basibranchial-1-element is_a
  basibranchial-element
C08B RO "Basibranchial 1 element composition" basibranchial-1-element is_a
C09- CL "Basibranchial 2 bone shape" basibranchial-2-bone
C11- RO "Basibranchial 4 element composition" basibranchial-4-element is_a
C12- RO "Basibranchial 5 element composition" basibranchial-5-element is_a
C14- NE "Post-ceratobranchial cartilage presence" post-ceratobranchial-
  cartilage is_a pharyngeal-arch-cartilage
C15- RO "Hypobranchial 1 element composition" hypobranchial-1-element is_a
C16- NE "Hypobranchial 1 bone antero-medial process presence" hypobranchial-1-
  bone-antero-medial-process part_of hypobranchial-1-element
C17- RO "Hypobranchial 3 element composition" hypobranchial-3-element is_a
C18- BR "Hypobranchial 4 cartilage presence" hypobranchial-4-element is_a
  hypobranchial-cartilage
C21- NE "Ceratobranchial 5 tooth presence" ceratobranchial-5-tooth part_of
  ceratobranchial-5-bone
C24- ME "Ceratobranchial 5 tooth number" ceratobranchial-5-tooth
C25- CL "Ceratobranchial 5 tooth shape" ceratobranchial-5-tooth
C27- CL "Epibranchial bone curvature" epibranchial-bone
C29- NE "Epibranchial 1 bone anterior membranous flange presence" epibranchial-
  1-bone-anterior-membranous-flange part_of epibranchial-1-bone
C33- NE "Epibranchial 1 bone uncinate process presence" epibranchial-1-bone-
  uncinate-process part_of epibranchial-1-bone
C34- NE "Epibranchial 2 bone uncinate process presence" epibranchial-2-bone-
  uncinate-process part_of epibranchial-2-bone
C35- NE "Epibranchial 3 bone uncinate process presence" epibranchial-3-bone-
  uncinate-process part_of epibranchial-3-bone
C36- NE "Epibranchial 4 bone uncinate process presence" epibranchial-4-bone-
  uncinate-process part_of epibranchial-4-bone
C38- NE "Epibranchial 4 bone, efferent artery flange presence" epibranchial-4-
  bone-efferent-artery-flange part_of epibranchial-4-bone
```

```
C39- NE "Epibranchial 4 bone, levator process, presence" epibranchial-4-bone-
  levator-process part_of epibranchial-4-bone
C40A NE "Pharyngobranchial 1 element presence" pharyngobranchial-1-element
  part_of gill-arch-1-skeleton
C40B RO "Pharyngobranchial 1 element composition" pharyngobranchial-1-element
  is_a
C41A NE "Pharyngobranchial 4 element presence" pharyngobranchial-4-element
  part_of gill-arch-4-skeleton
C41B RO "Pharyngobranchial 4 element composition" pharyngobranchial-4-element
  is_a
C46- NE "Pharyngobranchial 3 tooth plate presence" pharyngobranchial-3-tooth-
  plate part_of gill-arch-3-skeleton
C47- CL "Gill raker tip shape" gill-raker
C50- NE "Sublingual(s) presence" sublingual is_a endochondral-bone
C52A NE "Interhyal element presence" interhyal-element part_of ventral-hyoid-
  arch
C52B RO "Interhyal element composition" interhyal-element is_a
C54- ME "Branchiostegal rays number" branchiostegal-ray
C55- CL "Branchiostegal rays shape" branchiostegal-ray
C56- NE "Gill filament ossifications presence" gill-filament-ossification
  part_of gill
C58A NE "Hypobranchial 3 element ventral process presence" hypobranchial-3-
  element-ventral-process part_of hypobranchial-3-element
C58B RO "Hypobranchial 3 element ventral process composition" hypobranchial-3-
  element-ventral-process is_a
C61A NE "Urohyal ventral plate presence" urohyal-ventral-plate part_of urohyal
C61B CL "Urohyal ventral plate state" urohyal-ventral-plate
#
```

List of character states: (Each line starts with the ID of the character state, followed by its text and character).

```
C01-v0 "present" C01-
C01-v1 "absent" C01-
C07-v0 "present" C07-
C07-v1 "absent" C07-
C08Av0 "absent" C08A
C08Av1 "present" C08A
C08Bv0 "basibranchial-bone" C08B
C08Bv1 "cartilage" C08B
C09-v0 "hourglass" C09-
C09-v1 "rodlike" C09-
C09-v2 "spathulate" C09-
C09-v3 "anvil" C09-
C09-v4 "hourglass-with-flange" C09-
C09-v5 "rodlike-with-protrusions" C09-
C09-v6 "round" C09-
C11-v0 "cartilage" C11-
C11-v1 "basibranchial-bone" C11-
C12-v0 "cartilage" C12-
C12-v1 "basibranchial-bone" C12-
C14-v0 "absent" C14-
C14-v1 "present" C14-
C15-v0 "hypobranchial-bone" C15-
C15-v1 "hypobranchial-cartilage" C15-
```

```
C16-v0 "absent" C16-
C16-v1 "present" C16-
C17-v0 "hypobranchial-bone" C17-
C17-v1 "hypobranchial-cartilage" C17-
C18-v0 "no" C18-
C18-v1 "yes" C18-
C21-v0 "absent" C21-
C21-v1 "present" C21-
C24-v0 "1" C24-
C24-v1 "16" C24-
C24-v2 "26" C24-
C25-v0 "needle-like" C25-
C25-v1 "bicuspid" C25-
C25-v2 "hooked-with-no-wear-surface" C25-
C25-v3 "hooked-with-wear-surface" C25-
C25-v4 "molariform" C25-
C25-v5 "molariform-with-serrations" C25-
C25-v6 "bicuspid-and-hooked" C25-
C27-v0 "straight" C27-
C27-v1 "dorsally-arched" C27-
C29-v0 "absent" C29-
C29-v1 "present" C29-
C33-v0 "absent" C33-
C33-v1 "present" C33-
C34-v0 "absent" C34-
C34-v1 "present" C34-
C35-v0 "absent" C35-
C35-v1 "present" C35-
C36-v0 "absent" C36-
C36-v1 "present" C36-
C38-v0 "absent" C38-
C38-v1 "present" C38-
C39-v0 "absent" C39-
C39-v1 "present" C39-
C40Av0 "absent" C40A
C40Av1 "present" C40A
C40Bv0 "cartilage" C40B
C40Bv1 "pharyngobranchial-bone" C40B
C41Av0 "absent" C41A
C41Av1 "present" C41A
C41Bv0 "cartilage" C41B
C41Bv1 "pharyngobranchial-bone" C41B
C46-v0 "present" C46-
C46-v1 "absent" C46-
C47-v0 "tapered-tipped" C47-
C47-v1 "expanded-tipped" C47-
C50-v0 "absent" C50-
C50-v1 "present" C50-
C52Av0 "absent" C52A
C52Av1 "present" C52A
C52Bv0 "bone" C52B
C52Bv1 "interhyal-cartilage" C52B
C54-v0 "4" C54-
C54-v1 "3" C54-
C55-v0 "spathiform" C55-
C55-v1 "acinaciform" C55-
C56-v0 "absent" C56-
C56-v1 "present" C56-
```

```
C58Av0 "absent" C58A
C58Av1 "present" C58A
C58Bv0 "bone" C58B
C58Bv1 "cartilage" C58B
C61Av0 "absent" C61A
C61Av1 "present" C61A
C61Bv0 "reduced" C61B
C61Bv1 "fully-developed" C61B
#
```

Matrix: (Each line starts with the ID of the taxon, followed by its character state vector. Each element is an ID of a character state, or a ? to designate unknown or inapplicable).

```
N02 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v4 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v1 C18-v0 C21-v1 C24-v2 ?      C27-v0 C29-v0 ?      ?      C35-v1 C36-v0 C38-
  v0 C39-v1 C40Av1 ?      C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v0 C55-v0 C56-v0 C58Av0 ?      C61Av1 C61Bv1
N03 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v0 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v1 C18-v0 C21-v1 C24-v2 C25-v0 C27-v0 C29-v0 ?      ?      C35-v1 C36-v0 C38-
  v1 C39-v0 C40Av1 C40Bv1 C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v0 C55-v0 C56-v0 C58Av0 ?      C61Av1 C61Bv1
N04 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v0 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v2 C25-v0 C27-v0 C29-v0 ?      ?      C35-v1 C36-v0 C38-
  v1 C39-v0 C40Av1 C40Bv1 C41Av1 C41Bv0 C46-v0 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v0 C55-v0 C56-v0 C58Av0 ?      C61Av1 C61Bv1
N05 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v5 C11-v0 C12-v0 C14-v1 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v2 C25-v2 C27-v0 C29-v0 C33-v0 C34-v0 C35-v1 C36-v0 C38-
  v1 C39-v0 C40Av1 C40Bv1 C41Av1 C41Bv0 C46-v0 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v0 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N06 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v0 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v2 C25-v0 C27-v0 C29-v0 C33-v1 C34-v1 C35-v1 C36-v0 C38-
  v1 C39-v0 C40Av1 C40Bv1 C41Av1 C41Bv0 C46-v0 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v0 C55-v0 C56-v1 C58Av0 ?      C61Av1 C61Bv1
N07 C01-v1 C07-v1 ?      ?      C09-v0 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v1 C18-v0 C21-v1 C24-v2 C25-v0 C27-v0 C29-v0 C33-v0 C34-v0 C35-v1 C36-v0 C38-
  v1 C39-v0 C40Av1 C40Bv1 C41Av1 C41Bv1 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v0 C55-v1 C56-v0 C58Av0 ?      C61Av1 C61Bv1
N08 C01-v1 C07-v1 C08Av0 ?      C09-v0 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v1 C18-v0 C21-v1 C24-v2 C25-v0 C27-v0 C29-v0 C33-v0 C34-v0 C35-v1 C36-v1 C38-
  v1 C39-v0 C40Av0 ?      C41Av1 C41Bv1 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v0 C55-v1 C56-v1 C58Av0 ?      C61Av1 C61Bv1
N09 C01-v1 C07-v1 C08Av1 C08Bv1 C09-v1 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v1 C18-v0 C21-v1 C24-v2 C25-v0 C27-v0 C29-v0 C33-v0 C34-v0 C35-v1 C36-v0 C38-
  v1 C39-v0 C40Av1 C40Bv1 C41Av1 C41Bv1 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v0 C55-v1 C56-v0 C58Av0 ?      C61Av1 C61Bv0
N10 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v4 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v1 C18-v0 C21-v1 C24-v2 ?      ?      C29-v0 ?      C34-v1 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av1 ?      C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av0 ?      C61Av1 C61Bv1
N11 C01-v0 C07-v1 C08Av1 C08Bv0 ?      C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v3 ?      C29-v0 ?      C34-v1 C35-v1 C36-v1 C38-
```

```
  v0 C39-v1 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N12 C01-v0 C07-v1 C08Av1 C08Bv0 ?       C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v3 C27-v0 C29-v0 C33-v0 C34-v1 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N13 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v0 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v3 C27-v0 C29-v0 C33-v0 C34-v0 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N14 C01-v0 C07-v1 C08Av0 ?       C09-v0 C11-v0 C12-v0 C14-v0 C15-v0 C16-v1 C17-
  v0 C18-v0 C21-v1 C24-v0 ?       C27-v0 C29-v0 C33-v0 C34-v0 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N15 C01-v0 C07-v1 C08Av0 ?       C09-v0 C11-v0 C12-v0 C14-v0 C15-v0 C16-v1 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v0 C27-v0 C29-v0 C33-v0 C34-v0 C35-v1 C36-v1 C38-
  v1 C39-v1 C40Av0 ?       C41Av1 C41Bv1 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v1 C56-v0 C58Av0 ?       C61Av1 C61Bv0
N16 C01-v0 C07-v1 C08Av0 ?       ?       ?       ?       C14-v0 C15-v0 C16-v1 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v0 C27-v0 C29-v0 C33-v0 C34-v0 C35-v1 C36-v1 C38-
  v1 C39-v1 C40Av0 ?       C41Av1 C41Bv1 C46-v1 C47-v0 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v1 C56-v0 C58Av0 ?       C61Av1 C61Bv0
N17 C01-v0 C07-v1 C08Av0 ?       ?       ?       ?       C14-v0 C15-v0 C16-v1 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v0 C27-v0 C29-v0 C33-v0 C34-v0 C35-v0 C36-v0 C38-
  v1 C39-v1 C40Av0 ?       ?       ?       C46-v1 C47-v0 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v1 C56-v0 C58Av0 ?       C61Av1 C61Bv0
N18 C01-v0 C07-v1 C08Av0 ?       C09-v2 C11-v1 ?       C14-v0 C15-v0 C16-v1 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v0 C27-v0 C29-v0 C33-v0 C34-v0 C35-v0 C36-v0 C38-
  v1 C39-v1 C40Av0 ?       ?       ?       C46-v1 C47-v0 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v1 C56-v1 C58Av0 ?       C61Av1 C61Bv0
N19 C01-v0 C07-v1 C08Av0 ?       C09-v2 C11-v1 C12-v1 C14-v1 C15-v0 C16-v1 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v0 C27-v0 C29-v0 C33-v0 C34-v0 C35-v0 C36-v0 C38-
  v1 C39-v1 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v1 C56-v1 C58Av0 ?       C61Av1 C61Bv0
N20 C01-v0 C07-v1 C08Av0 ?       C09-v2 C11-v1 C12-v1 C14-v1 C15-v0 C16-v1 C17-
  v0 C18-v1 C21-v1 C24-v0 C25-v0 C27-v0 C29-v0 C33-v0 C34-v0 C35-v0 C36-v0 C38-
  v1 C39-v1 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v1 C56-v1 C58Av0 ?       C61Av1 C61Bv0
N21 C01-v0 C07-v1 C08Av0 ?       C09-v2 C11-v1 C12-v1 C14-v1 C15-v0 C16-v1 C17-
  v0 C18-v1 C21-v1 C24-v0 C25-v0 C27-v0 C29-v0 C33-v0 C34-v0 C35-v0 C36-v0 C38-
  v1 C39-v1 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v1 C56-v0 C58Av0 ?       C61Av1 C61Bv0
N22 C01-v0 C07-v1 C08Av0 ?       C09-v2 C11-v1 C12-v1 C14-v1 C15-v0 C16-v1 C17-
  v0 C18-v1 C21-v1 C24-v0 C25-v0 C27-v0 C29-v0 C33-v0 C34-v0 C35-v0 C36-v0 C38-
  v1 C39-v1 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v1 C56-v1 C58Av0 ?       C61Av0 ?
N23 C01-v0 C07-v1 C08Av0 ?       C09-v2 C11-v1 C12-v1 C14-v1 C15-v0 C16-v1 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v0 C27-v0 C29-v0 C33-v0 C34-v0 C35-v0 C36-v0 C38-
  v1 C39-v1 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v1 C56-v1 C58Av0 ?       C61Av1 C61Bv0
N24 C01-v0 C07-v1 C08Av0 ?       C09-v2 C11-v1 C12-v1 C14-v1 C15-v0 C16-v1 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v6 C27-v0 C29-v0 C33-v0 C34-v0 C35-v0 C36-v0 C38-
  v1 C39-v0 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v1 C56-v1 C58Av0 ?       C61Av1 C61Bv0
N25 C01-v0 C07-v1 C08Av0 ?       C09-v2 C11-v1 ?       C14-v0 C15-v0 C16-v1 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v0 C27-v0 C29-v0 C33-v0 C34-v0 C35-v0 C36-v0 C38-
  v1 C39-v1 C40Av0 ?       ?       ?       C46-v1 C47-v0 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v1 C56-v1 C58Av0 ?       C61Av1 C61Bv0
```

```
N26 C01-v0 C07-v1 C08Av0 ?       C09-v2 C11-v1 ?       C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v0 C27-v0 C29-v0 C33-v0 C34-v0 C35-v0 C36-v0 C38-
  v1 C39-v1 C40Av0 ?       ?       ?       C46-v1 C47-v0 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v1 C56-v1 C58Av0 ?       C61Av1 C61Bv0
N27 C01-v0 C07-v1 C08Av0 ?       C09-v2 C11-v1 C12-v1 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v0 C27-v0 C29-v0 C33-v0 C34-v0 C35-v0 C36-v0 C38-
  v1 C39-v1 C40Av0 ?       C41Av1 C41Bv1 C46-v1 C47-v0 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v1 C56-v1 C58Av0 ?       C61Av1 C61Bv0
N28 C01-v0 C07-v1 C08Av0 ?       C09-v2 C11-v1 ?       C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v1 C21-v1 C24-v0 C25-v0 C27-v0 C29-v0 C33-v0 C34-v0 C35-v0 C36-v0 C38-
  v1 C39-v1 C40Av0 ?       C41Av0 ?       C46-v1 C47-v0 C50-v1 C52Av0 ?       C54-
  v1 C55-v1 C56-v1 C58Av0 ?       C61Av1 C61Bv0
N29 C01-v0 C07-v1 C08Av0 ?       C09-v2 C11-v1 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v1 C21-v1 C24-v0 C25-v0 C27-v0 C29-v0 C33-v0 C34-v0 C35-v0 C36-v0 C38-
  v1 C39-v1 C40Av0 ?       C41Av0 ?       C46-v1 C47-v0 C50-v1 C52Av0 ?       C54-
  v1 C55-v1 C56-v1 C58Av0 ?       C61Av1 C61Bv0
N30 C01-v0 C07-v1 C08Av0 ?       C09-v2 C11-v1 ?       C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v1 C21-v1 C24-v0 C25-v0 C27-v0 C29-v0 C33-v0 C34-v0 C35-v0 C36-v0 C38-
  v0 C39-v1 C40Av0 ?       C41Av0 ?       C46-v1 C47-v0 C50-v1 C52Av0 ?       C54-
  v1 C55-v0 C56-v1 C58Av1 C58Bv0 C61Av1 C61Bv0
N31 C01-v0 C07-v1 C08Av0 ?       C09-v2 C11-v1 ?       C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v0 C27-v0 C29-v0 C33-v0 C34-v0 C35-v0 C36-v0 C38-
  v1 C39-v1 C40Av0 ?       C41Av0 ?       C46-v1 C47-v0 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v1 C56-v1 C58Av1 C58Bv0 C61Av1 C61Bv0
N32 C01-v0 C07-v1 C08Av0 ?       C09-v2 C11-v1 ?       C14-v0 C15-v0 C16-v1 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v0 C27-v0 C29-v0 C33-v0 C34-v0 C35-v0 C36-v0 C38-
  v1 C39-v1 C40Av0 ?       C41Av1 C41Bv1 C46-v1 C47-v0 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v1 C56-v1 C58Av0 ?       C61Av1 C61Bv0
N33 C01-v0 C07-v1 C08Av0 ?       C09-v0 ?       ?       C14-v0 C15-v0 C16-v1 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v0 C27-v0 C29-v0 C33-v0 C34-v0 C35-v0 C36-v0 C38-
  v1 C39-v1 C40Av0 ?       C41Av0 ?       C46-v1 C47-v0 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v1 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv0
N34 C01-v0 C07-v1 C08Av0 ?       C09-v2 ?       ?       C14-v0 C15-v0 C16-v1 C17-
  v0 C18-v1 C21-v1 C24-v0 C25-v0 C27-v0 C29-v0 C33-v0 C34-v0 C35-v1 C36-v1 C38-
  v1 C39-v1 C40Av0 ?       C41Av1 C41Bv1 C46-v1 C47-v0 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v1 C56-v0 C58Av0 ?       C61Av0 ?
N35 C01-v0 C07-v1 C08Av0 ?       C09-v0 C11-v0 C12-v0 C14-v0 C15-v0 C16-v1 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v0 C27-v0 C29-v0 C33-v0 C34-v0 C35-v1 C36-v1 C38-
  v1 C39-v1 C40Av0 ?       C41Av1 C41Bv1 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v1 C56-v0 C58Av0 ?       C61Av1 C61Bv0
N36 C01-v0 C07-v1 C08Av0 ?       C09-v3 C11-v0 C12-v0 C14-v0 C15-v0 C16-v1 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v4 C27-v1 C29-v1 C33-v0 C34-v0 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N37 C01-v0 C07-v1 C08Av0 ?       C09-v3 C11-v0 C12-v0 C14-v0 C15-v0 C16-v1 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v4 C27-v1 C29-v1 C33-v0 C34-v0 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N38 C01-v0 C07-v1 C08Av0 ?       C09-v3 C11-v0 C12-v0 C14-v0 C15-v0 C16-v1 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v4 C27-v1 C29-v1 C33-v0 C34-v0 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N39 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v0 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v3 C27-v0 C29-v0 C33-v0 C34-v0 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N40 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v0 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v5 C27-v0 C29-v1 C33-v0 C34-v0 C35-v1 C36-v1 C38-
```

```
  v0 C39-v1 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N41 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v0 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v3 C27-v0 C29-v0 C33-v0 C34-v0 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N42 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v1 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v3 C27-v0 C29-v0 C33-v0 C34-v0 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N43 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v1 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v2 C27-v0 C29-v0 C33-v0 C34-v0 C35-v0 C36-v1 C38-
  v0 C39-v1 C40Av0 ?       C41Av0 ?       C46-v1 C47-v0 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N44 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v0 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v4 C27-v0 C29-v0 C33-v0 C34-v0 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N45 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v0 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v4 C27-v0 C29-v0 C33-v0 C34-v0 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N46 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v0 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v3 C27-v0 C29-v0 C33-v0 C34-v0 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N47 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v0 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v4 C27-v0 C29-v0 C33-v0 C34-v0 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N48 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v0 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v4 C27-v1 C29-v1 C33-v0 C34-v0 C35-v0 C36-v0 C38-
  v0 C39-v0 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N49 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v0 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v4 C27-v0 C29-v0 C33-v0 C34-v0 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N50 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v1 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v3 C27-v0 C29-v0 C33-v0 C34-v0 C35-v0 C36-v1 C38-
  v0 C39-v1 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N51 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v0 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v3 C27-v1 C29-v1 C33-v0 C34-v0 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N52 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v1 C11-v0 C12-v0 C14-v0 C15-v1 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v3 C27-v0 C29-v0 C33-v0 C34-v0 C35-v0 C36-v1 C38-
  v0 C39-v1 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv1 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N53 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v0 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v2 C27-v0 C29-v0 C33-v0 C34-v0 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N54 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v0 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v2 C27-v0 C29-v0 C33-v0 C34-v0 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
```

```
N55 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v0 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v3 C27-v0 C29-v0 C33-v0 C34-v0 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?      C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N56 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v1 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v3 C27-v0 C29-v0 C33-v0 C34-v1 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?      C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N57 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v1 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v3 C27-v0 C29-v0 C33-v0 C34-v1 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?      C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N58 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v1 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v3 C27-v0 C29-v0 C33-v1 C34-v1 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?      C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N59 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v1 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v4 C27-v0 C29-v0 C33-v0 C34-v1 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?      C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N60 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v1 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v4 C27-v0 C29-v1 C33-v0 C34-v0 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?      C41Av0 ?      C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N61 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v0 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v4 C27-v1 C29-v0 C33-v0 C34-v1 C35-v1 C36-v1 C38-
  v1 C39-v1 C40Av0 ?      C41Av1 C41Bv0 C46-v1 C47-v1 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N62 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v1 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v5 C27-v0 C29-v0 C33-v0 C34-v0 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?      C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av0 ?      C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N63 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v1 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v2 C27-v0 C29-v0 C33-v0 C34-v0 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?      C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N64 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v1 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v3 C27-v0 C29-v0 C33-v0 C34-v0 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?      C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N65 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v0 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v3 C27-v0 C29-v1 C33-v0 C34-v0 C35-v1 C36-v0 C38-
  v0 C39-v1 C40Av0 ?      C41Av0 ?      C46-v1 C47-v0 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N66 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v0 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v3 C27-v0 C29-v1 C33-v0 C34-v0 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?      C41Av1 C41Bv0 C46-v1 C47-v0 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N67 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v1 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v2 C27-v0 C29-v0 C33-v0 C34-v0 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?      C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N68 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v0 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 ?      C27-v0 C29-v0 C33-v0 C34-v0 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?      C41Av0 ?      C46-v1 C47-v0 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N69 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v1 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v3 C27-v0 C29-v0 C33-v0 C34-v0 C35-v1 C36-v1 C38-
```

```
  v0 C39-v1 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N70 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v1 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v2 C27-v0 C29-v0 C33-v0 C34-v0 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N71 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v1 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v3 C27-v0 C29-v0 C33-v0 C34-v1 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N72 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v0 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v3 C27-v1 C29-v0 C33-v1 C34-v1 C35-v1 C36-v0 C38-
  v0 C39-v1 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv1 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N73 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v4 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v0 C21-v1 C24-v0 C25-v3 C27-v1 C29-v1 C33-v0 C34-v0 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av0 ?       C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N74 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v4 C11-v0 C12-v0 C14-v1 C15-v0 C16-v0 C17-
  v1 C18-v0 C21-v1 C24-v2 C25-v1 C27-v1 C29-v0 C33-v1 C34-v1 C35-v1 C36-v1 C38-
  v0 C39-v0 C40Av1 C40Bv0 C41Av1 C41Bv0 C46-v1 C47-v1 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv1 C61Av1 C61Bv0
N75 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v4 C11-v0 C12-v0 C14-v1 C15-v0 C16-v0 C17-
  v1 C18-v0 C21-v1 C24-v2 C25-v1 C27-v1 C29-v0 C33-v1 C34-v1 C35-v1 C36-v1 C38-
  v0 C39-v0 C40Av1 C40Bv0 C41Av1 C41Bv0 C46-v1 C47-v1 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv0
N76 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v4 C11-v0 C12-v0 C14-v1 C15-v0 C16-v0 C17-
  v1 C18-v0 C21-v1 C24-v2 C25-v1 C27-v1 C29-v0 C33-v1 C34-v1 C35-v1 C36-v1 C38-
  v0 C39-v0 C40Av1 C40Bv0 C41Av1 C41Bv0 C46-v1 C47-v1 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv1 C61Av1 C61Bv0
N77 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v6 C11-v0 C12-v0 C14-v1 C15-v0 C16-v0 C17-
  v1 C18-v1 C21-v1 C24-v2 C25-v1 C27-v1 C29-v0 C33-v1 C34-v1 C35-v1 C36-v1 C38-
  v0 C39-v0 C40Av1 C40Bv0 C41Av1 C41Bv0 C46-v1 C47-v1 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv1 C61Av1 C61Bv0
N78 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v4 C11-v0 C12-v0 C14-v1 C15-v0 C16-v0 C17-
  v1 C18-v0 C21-v1 C24-v2 C25-v1 C27-v1 C29-v0 C33-v1 C34-v1 C35-v1 C36-v1 C38-
  v0 C39-v0 C40Av1 C40Bv0 C41Av1 C41Bv0 C46-v1 C47-v1 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv1 C61Av1 C61Bv0
N79 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v4 C11-v0 C12-v0 C14-v1 C15-v0 C16-v0 C17-
  v1 C18-v0 C21-v1 C24-v2 C25-v1 C27-v1 C29-v0 C33-v1 C34-v1 C35-v1 C36-v1 C38-
  v0 C39-v0 C40Av1 C40Bv0 C41Av1 C41Bv0 C46-v1 C47-v1 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv1 C61Av1 C61Bv0
N80 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v4 C11-v0 C12-v0 C14-v1 C15-v0 C16-v0 C17-
  v1 C18-v0 C21-v1 C24-v2 C25-v1 C27-v1 C29-v0 C33-v1 C34-v1 C35-v1 C36-v1 C38-
  v0 C39-v0 C40Av1 C40Bv0 C41Av1 C41Bv0 C46-v1 C47-v1 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv1 C61Av1 C61Bv0
N81 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v4 C11-v0 C12-v0 C14-v1 C15-v0 C16-v0 C17-
  v1 C18-v0 C21-v1 C24-v2 C25-v1 C27-v1 C29-v0 C33-v1 C34-v1 C35-v1 C36-v1 C38-
  v0 C39-v0 C40Av1 C40Bv0 C41Av1 C41Bv0 C46-v1 C47-v1 C50-v1 C52Av1 C52Bv0 C54-
  v1 C55-v0 C56-v0 C58Av1 C58Bv1 C61Av1 C61Bv0
N82 C01-v0 C07-v1 C08Av0 ?       C09-v4 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v1 C18-v0 C21-v0 C24-v2 ?       C27-v0 C29-v0 C33-v1 C34-v1 C35-v1 C36-v1 C38-
  v0 C39-v1 C40Av1 C40Bv1 C41Av0 ?       C46-v1 C47-v0 C50-v0 C52Av0 ?       C54-
  v1 C55-v0 C56-v0 C58Av0 ?       C61Av1 C61Bv1
N83 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v4 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v1 C18-v0 C21-v0 C24-v2 ?       C27-v0 C29-v0 ?       ?       C35-v1 C36-v0 C38-
  v0 C39-v1 C40Av1 C40Bv0 C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v0 C55-v0 C56-v0 C58Av0 ?       C61Av1 C61Bv1
```

```
N84 C01-v0 C07-v1 C08Av1 C08Bv0 C09-v1 C11-v0 C12-v0 C14-v0 C15-v0 C16-v0 C17-
  v0 C18-v1 C21-v0 C24-v2 ?      C27-v0 C29-v0 C33-v1 C34-v1 C35-v1 C36-v0 C38-
  v0 C39-v1 C40Av1 C40Bv0 C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av0 ?      C54-
  v0 C55-v0 C56-v0 C58Av1 C58Bv0 C61Av1 C61Bv1
N85 C01-v0 C07-v0 C08Av1 C08Bv1 C09-v4 C11-v0 C12-v0 C14-v1 C15-v0 C16-v0 C17-
  v1 C18-v0 C21-v0 C24-v2 ?      C27-v1 C29-v0 C33-v0 C34-v0 C35-v1 C36-v0 C38-
  v0 C39-v1 C40Av1 C40Bv0 C41Av1 C41Bv0 C46-v1 C47-v0 C50-v0 C52Av1 C52Bv0 C54-
  v0 C55-v0 C56-v0 C58Av0 ?      C61Av1 C61Bv0
```

Appendix D

Cypriniformes Gill & Hyoid Arches Anatomy

Each line contains a single term. Each term has an identifier in square brackets to indicate the sequence where it first occurred in the ontology. The relationships between terms are indicated by indentation. Each term (except the root `whole`) is connected to the last term at the previous level by either `is_a` or `part_of` relationship, indicated as (i) and (p) respectively.

```
[1] whole
  (p) [2] bone
    (i) [3] interhyal-element
    (i) [4] replacement-bone
      (i) [5] hypobranchial-bone
        (i) [6] hypobranchial-1-element
      (i) [7] basibranchial-bone
        (i) [8] basibranchial-2-bone
          (i) [9] hourglass-with-flange-basibranchial-2-bone
        (i) [10] basibranchial-1-bone
      (i) [11] endochondral-bone
        (i) [12] ceratobranchial-bone
          (i) [13] ceratobranchial-5-bone
            (p) [14] ceratobranchial-5-tooth
              (i) [15] ceratobranchial-5-tooth#16
              (i) [16] ceratobranchial-5-tooth#22
              (i) [17] ceratobranchial-5-tooth#15
              (i) [18] ceratobranchial-5-tooth#18
              (i) [19] ceratobranchial-5-tooth#17
              (i) [20] ceratobranchial-5-tooth#1
              (i) [21] ceratobranchial-5-tooth#8
              (i) [22] ceratobranchial-5-tooth#10
              (i) [23] ceratobranchial-5-tooth#2
              (i) [24] ceratobranchial-5-tooth#20
              (i) [25] ceratobranchial-5-tooth#12
```

```
            (i) [26] ceratobranchial-5-tooth#23
            (i) [27] ceratobranchial-5-tooth#9
            (i) [28] ceratobranchial-5-tooth#19
            (i) [29] ceratobranchial-5-tooth#25
            (i) [30] ceratobranchial-5-tooth#21
            (i) [31] ceratobranchial-5-tooth#4
            (i) [32] ceratobranchial-5-tooth#5
            (i) [33] ceratobranchial-5-tooth#14
            (i) [34] ceratobranchial-5-tooth#6
            (i) [35] ceratobranchial-5-tooth#13
            (i) [36] ceratobranchial-5-tooth#3
            (i) [37] ceratobranchial-5-tooth#26
            (i) [38] ceratobranchial-5-tooth#11
            (i) [39] ceratobranchial-5-tooth#7
            (i) [40] ceratobranchial-5-tooth#24
      (i) [41] epibranchial-bone
        (i) [42] epibranchial-3-bone
          (p) [43] epibranchial-3-bone-uncinate-process
        (i) [44] epibranchial-4-bone
          (p) [45] epibranchial-4-bone-uncinate-process
          (p) [46] epibranchial-4-bone-levator-process
        (i) [47] epibranchial-2-bone
          (p) [48] epibranchial-2-bone-uncinate-process
        (i) [49] epibranchial-1-bone
          (p) [50] epibranchial-1-bone-uncinate-process
        (p) [51] epibranchial-bone-uncinate-process
  (i) [52] intramembranous-bone
    (i) [53] dermal-bone
      (i) [54] branchiostegal-ray
        (i) [55] branchiostegal-ray#1
        (i) [56] branchiostegal-ray#3
        (i) [57] spathiform-branchiostegal-ray
        (i) [58] branchiostegal-ray#2
(p) [59] skeletal-element
  (i) [60] basibranchial-element
    (i) [61] basibranchial-4-element
      (i) [62] basibranchial-4-cartilage
    (i) [63] basibranchial-5-element
      (i) [64] basibranchial-5-cartilage
    (i) [65] basibranchial-1-element
      (i) [10] basibranchial-1-bone
    (i) [66] basibranchial-3-element
    (i) [67] basibranchial-2-element
      (i) [8] basibranchial-2-bone
  (i) [68] ceratobranchial-element
    (i) [69] ceratobranchial-3-element
    (i) [70] ceratobranchial-1-element
    (i) [71] ceratobranchial-4-element
    (i) [72] ceratobranchial-2-element
    (i) [73] ceratobranchial-5-element
      (i) [13] ceratobranchial-5-bone
  (i) [74] epibranchial-element
```

```
    (i) [75] epibranchial-1-element
      (i) [49] epibranchial-1-bone
    (i) [76] epibranchial-4-element
      (i) [44] epibranchial-4-bone
    (i) [77] epibranchial-2-element
      (i) [47] epibranchial-2-bone
    (i) [78] epibranchial-3-element
      (i) [42] epibranchial-3-bone
  (i) [3] interhyal-element
  (i) [79] pharyngobranchial-element
    (i) [80] pharyngobranchial-4-element
    (i) [81] pharyngobranchial-1-element
  (i) [82] hypobranchial-element
    (i) [83] hypobranchial-3-element
    (i) [6] hypobranchial-1-element
    (i) [84] hypobranchial-4-element
      (i) [85] hypobranchial-4-cartilage
    (i) [86] hypobranchial-2-element
  (i) [87] basihyal-element
(p) [88] pharyngeal-arch
  (p) [89] pharyngeal-musculature
  (p) [90] pharyngeal-vasculature
  (p) [91] splanchnocranium
    (p) [92] suspensorium
    (i) [93] mandibular-arch
    (i) [94] hyoid-arch
      (p) [95] ventral-hyoid-arch
        (p) [3] interhyal-element
        (p) [96] urohyal
          (p) [97] urohyal-ventral-plate
            (i) [98] fully-developed-urohyal-ventral-plate
        (p) [87] basihyal-element
      (p) [99] dorsal-hyoid-arch
    (p) [100] pharyngeal-arch-cartilage
      (i) [101] hypobranchial-cartilage
        (i) [83] hypobranchial-3-element
      (i) [102] epibranchial-cartilage
      (i) [103] ceratobranchial-cartilage
    (i) [104] gill-arch-1-5-skeleton
      (p) [41] epibranchial-bone
      (i) [105] gill-arch-5-skeleton
      (p) [12] ceratobranchial-bone
      (p) [102] epibranchial-cartilage
      (p) [60] basibranchial-element
      (p) [106] gill-raker-row
      (p) [74] epibranchial-element
      (p) [107] copula
      (p) [108] ceratobranchial-series
      (p) [68] ceratobranchial-element
      (p) [109] epibranchial-series
      (i) [110] gill-arch-3-skeleton
        (p) [83] hypobranchial-3-element
```

```
      (i) [111] gill-arch-1-skeleton
        (p) [6] hypobranchial-1-element
        (p) [81] pharyngobranchial-1-element
      (p) [112] pharyngobranchial-cartilage
      (p) [103] ceratobranchial-cartilage
      (p) [113] gill-raker
        (i) [114] tapered-tipped-gill-raker
      (p) [115] basibranchial-4-tooth-plate
      (p) [116] basibranchial-2-tooth-plate
      (p) [7] basibranchial-bone
      (p) [117] pharyngobranchial-bone
      (p) [118] pharyngobranchial-tooth-plate
      (i) [119] gill-arch-2-skeleton
        (p) [86] hypobranchial-2-element
      (p) [5] hypobranchial-bone
      (p) [82] hypobranchial-element
      (p) [101] hypobranchial-cartilage
      (p) [79] pharyngobranchial-element
      (i) [120] gill-arch-4-skeleton
        (p) [80] pharyngobranchial-4-element
        (p) [84] hypobranchial-4-element
  (i) [121] pharyngeal-arch#1
    (p) [93] mandibular-arch
    (p) [122] mandibular-muscle
  (i) [123] gill-arch
    (p) [124] branchial-muscle
    (i) [125] gill-arch#3
    (i) [126] gill-arch#5
    (i) [127] gill-arch#2
    (i) [128] gill-arch#4
    (p) [104] gill-arch-1-5-skeleton
    (i) [129] gill-arch#1
  (p) [130] gill-ray
  (i) [131] pharyngeal-arch#2
    (p) [94] hyoid-arch
    (i) [132] synovial-joint
    (i) [133] fibrous-joint
    (p) [134] hyoid-muscle
  (p) [135] pharyngeal-pouch
(p) [136] gill
  (p) [113] gill-raker
  (p) [123] gill-arch
(p) [137] cartilage
  (i) [63] basibranchial-5-element
  (i) [138] cranial-cartilage
    (i) [100] pharyngeal-arch-cartilage
  (i) [80] pharyngobranchial-4-element
  (i) [61] basibranchial-4-element
(p) [139] tooth
  (i) [14] ceratobranchial-5-tooth
```

Appendix E

Morphster Meta-Model in OWL

```
Prefix: xsd: <http://www.w3.org/2001/XMLSchema#>
Prefix: owl: <http://www.w3.org/2002/07/owl#>
Prefix: xml: <http://www.w3.org/XML/1998/namespace>
Prefix: rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
Prefix: rdfs: <http://www.w3.org/2000/01/rdf-schema#>
Prefix: skos: <http://www.w3.org/2004/02/skos/core#>
Prefix: morphster: <http://www.cs.utexas.edu/~hamid/ontology/morphster#>


Ontology: <http://www.cs.utexas.edu/~hamid/ontology/morphster>


AnnotationProperty: rdfs:label


ObjectProperty: <morphster:isSpecimenOf>

    SubPropertyOf:
        owl:topObjectProperty

    Domain:
        <morphster:Specimen>

    Range:
        <morphster:Taxon>


ObjectProperty: <morphster:registeredTo>

    SubPropertyOf:
        owl:topObjectProperty

    Domain:
        <morphster:Image>

    Range:
        <morphster:Specimen>


ObjectProperty: <morphster:associatedTo>

    SubPropertyOf:
```

```
        owl:topObjectProperty

    Domain:
        <morphster:MatrixCell>


ObjectProperty: owl:topObjectProperty


ObjectProperty: <morphster:belongsTo>

    SubPropertyOf:
        owl:topObjectProperty

    Domain:
        <morphster:MatrixCell>

    Range:
        <morphster:Matrix>


ObjectProperty: <morphster:shows>

    SubPropertyOf:
        owl:topObjectProperty

    Domain:
        <morphster:Image>

    InverseOf:
        <morphster:shownIn>


ObjectProperty: <morphster:shownIn>

    SubPropertyOf:
        owl:topObjectProperty

    InverseOf:
        <morphster:shows>


ObjectProperty: <morphster:isEntityOf>

    SubPropertyOf:
        owl:topObjectProperty

    Domain:
        <morphster:AnatomicalEntity>

    InverseOf:
        <morphster:appearsInEntity>


ObjectProperty: <morphster:appearsInEntity>

    SubPropertyOf:
        owl:topObjectProperty
```

```
    InverseOf:
        <morphster:isEntityOf>


ObjectProperty: <morphster:isStateOf>

    SubPropertyOf:
        owl:topObjectProperty

    Domain:
        <morphster:CharacterState>

    Range:
        <morphster:Character>


ObjectProperty: <morphster:hasQuality>

    SubPropertyOf:
        owl:topObjectProperty

    InverseOf:
        <morphster:isQualityOf>


ObjectProperty: <morphster:isAncestorOf>

    SubPropertyOf:
        owl:topObjectProperty

    Domain:
        <morphster:Taxon>

    Range:
        <morphster:Taxon>


ObjectProperty: <morphster:isQualityOf>

    SubPropertyOf:
        owl:topObjectProperty

    Domain:
        <morphster:PhenotypicQuality>

    InverseOf:
        <morphster:hasQuality>


Class: <morphster:AnatomicalEntity>

    Annotations:
        rdfs:label "Anatomical Entity"@en

    SubClassOf:
        <morphster:shownIn> some <morphster:Image>
```

```
Class: <morphster:Character>

    Annotations:
        rdfs:label "Character"@en

    SubClassOf:
        <morphster:appearsInEntity> some <morphster:AnatomicalEntity>,
        <morphster:hasQuality> some <morphster:PhenotypicQuality>,
        <morphster:shownIn> some <morphster:Image>


Class: <morphster:CharacterState>

    Annotations:
        rdfs:label "Character State"@en

    SubClassOf:
        <morphster:appearsInEntity> some <morphster:AnatomicalEntity>,
        <morphster:hasQuality> some <morphster:PhenotypicQuality>,
        <morphster:isStateOf> some <morphster:Character>,
        <morphster:shownIn> some <morphster:Image>


Class: <morphster:PhenotypicQuality>

    Annotations:
        rdfs:label "Phenotypic Quality"@en


Class: <morphster:Taxon>

    Annotations:
        rdfs:label "Taxon"@en

    SubClassOf:
        <morphster:isAncestorOf> some <morphster:Taxon>


Class: <morphster:Matrix>

    Annotations:
        rdfs:label "Matrix"@en


Class: <morphster:Image>

    Annotations:
        rdfs:label "Image"@en

    SubClassOf:
        <morphster:registeredTo> some <morphster:Specimen>


Class: <morphster:MatrixCell>

    Annotations:
        rdfs:label "Matrix Cell"@en
```

```
    SubClassOf:
        <morphster:associatedTo> some <morphster:CharacterState>,
        <morphster:associatedTo> some <morphster:Taxon>,
        <morphster:belongsTo> some <morphster:Matrix>


Class: <morphster:Specimen>

    Annotations:
        rdfs:label "Specimen"@en

    SubClassOf:
        <morphster:isSpecimenOf> some <morphster:Taxon>


DisjointClasses:

<morphster:AnatomicalEntity>,<morphster:Character>,<morphster:CharacterState>,<
morphster:Image>,<morphster:Matrix>,<morphster:MatrixCell>,<morphster:Phenotypi
cQuality>,<morphster:Specimen>,<morphster:Taxon>
```

Appendix F

Image Driven Ontology Source in RDF/OWL

```
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [
  <!ENTITY rdf  "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
  <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
  <!ENTITY xsd  "http://www.w3.org/2001/XMLSchema#" >
  <!ENTITY owl  "http://www.w3.org/2002/07/owl#" >
  <!ENTITY ido  "http://www.morphster.org/ontology/2008/ido#" >
]>

<rdf:RDF xmlns:rdf="&rdf;" xmlns:rdfs="&rdfs;" xmlns:owl="&owl;"
xmlns:ido="&ido;" xmlns="&ido;"
xml:base="http://www.morphster.org/ontology/2008/ido#">

<owl:Ontology rdf:ID="ido-meta">
</owl:Ontology>

<!-- *** Start of IDO declarations *** -->

<owl:Class          rdf:ID="Image" />

<rdfs:Class  rdf:ID="ShowsProperty">
<rdfs:label>ShowsProperty</rdfs:label>
<rdfs:subClassOf rdf:resource="&rdf;Property"/>
</rdfs:Class>

<ido:ShowsProperty  rdf:ID="showsSimilar" />
<ido:ShowsProperty  rdf:ID="entiretyShowsSimilar" />
<ido:ShowsProperty  rdf:ID="showsExemplar" />
<ido:ShowsProperty  rdf:ID="entiretyShowsExemplar" />

<!-- *** End of IDO declarations *** -->

<!-- *** Full definitions follow *** -->

<owl:Class rdf:about="#Image">
<rdfs:label>Image</rdfs:label>
<rdfs:comment>The image entity. All images are instances of this
class.</rdfs:comment>
</owl:Class>

<ido:ShowsProperty  rdf:about="#showsSimilar">
<rdfs:label>Shows Similar</rdfs:label>
```

```
<rdfs:comment>Shows Similar (m, x): This association asserts that image m
contains a concept similar to class x.</rdfs:comment>
<rdfs:domain rdf:resource="&ido;Image" />
</ido:ShowsProperty>

<ido:ShowsProperty  rdf:about="#entiretyShowsSimilar">
<rdfs:label>Entirety Shows Similar</rdfs:label>
<rdfs:comment>Entirety Shows Similar (m, x): This association asserts that
image m, in its entirety, depicts a concept similar to class x.</rdfs:comment>
<rdfs:subPropertyOf rdf:resource="&ido;showsSimilar" />
</ido:ShowsProperty>

<ido:ShowsProperty  rdf:about="#showsExemplar">
<rdfs:label>Shows Exemplar</rdfs:label>
<rdfs:comment>Shows Exemplar (m, x): This association asserts than image m
contains exemplar of class x.</rdfs:comment>
<rdfs:subPropertyOf rdf:resource="&ido;showsSimilar" />
</ido:ShowsProperty>

<ido:ShowsProperty  rdf:about="#entiretyShowsExemplar">
<rdfs:label>Entirety Shows Exemplar</rdfs:label>
<rdfs:comment>Entirety Shows Exemplar (m, x): This association asserts that
image m, in its entirety, depicts exemplar of class x.</rdfs:comment>
<rdfs:subPropertyOf rdf:resource="&ido;showsExemplar" />
<rdfs:subPropertyOf rdf:resource="&ido;entiretyShowsSimilar" />
</ido:ShowsProperty>

</rdf:RDF>
```

Bibliography

[1] C. Linnæus. *Systema naturæ, sive regna tria naturæ systematice proposita per classes, ordines, genera, & species*. Lugduni Batavorum, (Haak), 1-12, 1735.

[2] C. Linnæus. *Systema naturæ per regna tria naturæ, secundum classes, ordines, genera, species, cum characteribus, differentiis, synonymis, locis.* (Translated: System of nature through the three kingdoms of nature, according to classes, orders, genera and species, with characters, differences, synonyms, places). Tomus I. Editio decima, reformata, pp [1-4], 1-824, Holmiæ, (Salvius), 1758.

[3] C. Darwin. *Transmutation of Species*. Notebook B, 1837-1838.

[4] M. Ashburner, C. A. Ball, J. A. Blake, D. Botstein, H. Butler, J. M. Cherry, A. P. Davis, K. Dolinski, S. S. Dwight, J. T. Eppig, M. A. Harris, D. P. Hill, L. Issel-Tarver, A. Kasarskis, S. Lewis, J. C. Matese, J. E. Richardson, M. Ringwald, G. M. Rubin, G. Sherlock. Gene Ontology: tool for the unification of biology. *Nature Genetics*, 25, 25-29, 2000.

[5] P. Clark, B. W. Porter. Building Concept Representations from Reusable Components. *National Conference on Artificial Intelligence, AAAI*, 369-376, 1997.

[6] K. Barker, B. W. Porter, P. Clark. A Library of Generic Concepts for Composing

Knowledge Bases. In Proceedings of *1st International Conference on Knowledge Capture*, Victoria, B.C., Canada, 2001.

[7] K. Barker, S. Y. Chaw, J. Fan, B. Porter, D. Tecuci, P. Yeh, V. K. Chaudhri, D. Israel, S. Mishra, P. Romero, P. E. Clark. A Question-Answering System for AP Chemistry: Assessing KR&R Technologies. In Proceedings of the *Ninth International Conference on the Principles of Knowledge Representation and Reasoning*, Whistler, 488-497, 2004.

[8] Assembling the Tree of Life (AToL). http://www.phylo.org/atol/

[9] J. Sprague, L. Bayraktaroglu, D. Clements, T. Conlin, D. Fashena, K. Frazer, M. Haendel, D. Howe, P. Mani, S. Ramachandran, K. Schaper, E. Segerdell, P. Song, B. Sprunger, S. Taylor, C. Van Slyke, M. Westerfield. The Zebrafish Information Network: the zebrafish model organism database. *Nucl. Acids Res.* 34, D581-D585, 2006.

[10] W. M. Dahdul, J. G. Lundberg, P. E. Midford, J. P. Balhoff, H. Lapp, T. J. Vision, M. A. Haendel, M. Westerfield, P. M. Mabee. The Teleost Anatomy Ontology: Anatomical Representation for the Genomics Age. *Systematic Biology*, 59 (4), 369-383, 2010.

[11] OBO-Edit: The OBO Ontology Editor. http://oboedit.org/

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

[13] W. P. Maddison, D. R. Maddison. *Mesquite: a modular system for evolutionary analysis*. 2010. http://mesquiteproject.org

[14] T. Warnow. Constructing phylogenetic trees efficiently using compatibility criteria. *New Zealand Journal of Botany*, Vol. 31, pp. 239-248, 1993.

[15] T. Warnow. *Large-scale phylogenetic reconstruction.* Book chapter, in S. Aluru (editor), Handbook of Computational Biology, Chapman & Hall, CRC Computer and Information Science Series, 2005.

[16] B. Smith, M. Ashburner, C. Rosse, J. Bard, W. Bug, W. Ceusters, L. J. Goldberg, K. Eilbeck, A Ireland, C. J. Mungall, OBI Consortium, N. Leontis, P. Rocca-Serra, A. Ruttenberg, S. A. Sansone, R. H. Scheuermann, N. Shah, P. L. Whetzel, S. Lewis. The OBO Foundry: coordinated evolution of ontologies to support biomedical data integration. *Nature Biotechnology*, 25 (11), 1251-1255, 2007.

[17] T. Berners-Lee, J. Hendler, O. Lassila. The Semantic Web. *Scientific American*, 284 (5), 34-43, 2001.

[18] Mouse adult gross anatomy. http://bioportal.bioontology.org/visualize/40819/MA%253A0000001

[19] NCI Thesaurus. http://ncit.nci.nih.gov/

[20] G. D. Bader, M. Cary, C. Sander. BioPAX – Biological Pathway Data Exchange Format. *Encyclopedia of Genomics, Proteomics and Bioinformatics*, New York: John Wiley & Sons, Ltd, 2006.

[21] P. C. Sereno. Logical basis for morphological characters in phylogenetics. *Cladistics* 23 (6), 565-587, 2007.

[22] Morphster project. http://www.cs.utexas.edu/~miranker/studentWeb/MorphsterHomePage.html

[23] S. H. Tirmizi, S. Aitken, D. A. Moreira, C. Mungall, J. Sequeda, N. H. Shah, D.

P. Miranker. OBO & OWL: Roundtrip Ontology Transformations. In Proceedings of the *Workshop of Semantic Web Applications and Tools for Life Sciences (SWAT4LS)*, Amsterdam, The Netherlands, November 20, 2009.

[24] T. R. Gruber. A Translation Approach to Portable Ontology Specification. *Knowledge Acquisition*, 5 (2), 199-220, 1993.

[25] W3C Recommendation. *OWL Web Ontology Language Use Cases and Requirements*. J. Heflin, ed. 2004. http://www.w3.org/TR/2004/REC-webont-req-20040210/

[26] I. Horrocks, P. F. Patel-Schneider, F. van Harmelen. From SHIQ and RDF to OWL: The Making of a Web Ontology Language. *Journal of Web Semantics*, 1 (1), 7-26, 2003.

[27] B. Smith, W. Ceusters, B. Klagges, J. Kohler, A. Kumar, J. Lomax, C. J. Mungall, F. Neuhaus, A. Rector, C. Rosse. Relations in Biomedical Ontologies. *Genome Biology*, 6:R46, 2005.

[28] G. Antoniou, F. van Harmelen. *A Semantic Web Primer.* MIT Press, 2004.

[29] W3C Recommendation. *Extensible Markup Language (XML) 1.1 (Second Edition)*. Bray, T., Paoli, J., Sperberg-McQueen, C. M., Maler, E., Yergeau, F., & Cowan, J., eds. 2006. http://www.w3.org/TR/2006/REC-xml11-20060816/

[30] W3C Recommendation. *Resource Description Framework (RDF): Concepts and Abstract Syntax*. Klyne, G., & Carroll, J. J., eds. 2004. http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/

[31] T. Berners-Lee, R. Fielding, L. Masinter. Uniform Resource Identifiers (URI): Generic Syntax. RFC 2396, Network Working Group, August, 1998.

[32] W3C Recommendation. *RDF Semantics.* P. Hayes, ed. 10 Feb, 2004.

http://www.w3.org/TR/2004/REC-rdf-mt-20040210/

[33] W3C Recommendation. *OWL Web Ontology Language.* D.L. McGuinness and F. van Harmelen, eds. 10 Feb, 2004. http://www.w3.org/TR/2004/REC-owl-features-20040210/

[34] W3C Recommendation. *OWL Web Ontology Language Reference.* M. Dean and G. Schreiber, eds. 10 Feb, 2004. http://www.w3.org/TR/2004/REC-owl-ref-20040210/

[35] W3C Recommendation. SPARQL Query Language for RDF. Prud'hommeaux, E., & Seaborne, A., eds. 2008. http://www.w3.org/TR/2008/REC-rdf-sparql-query-20080115/

[36] W3C Working Group Note. RIF Overview. Kifer, M., & Boley, H., eds. 2010. http://www.w3.org/TR/2010/NOTE-rif-overview-20100622/

[37] W3C Member Submission. SWRL: A Semantic Web Rule Language Combining OWL and RuleML. Horrocks, I., Patel-Schneider, P. F., Boley, H., Tabet, S., Grosof, B., & Dean, M., eds. 2004. http://www.w3.org/Submission/2004/SUBM-SWRL-20040521/

[38] NCBO BioPortal. http://www.bioontology.org/bioportal.html

[39] P. D. Karp, M. Riley, M. Saier, I. T. Paulsen, J. Collado-Vides, S. M. Paley, A. Pellegrini-Toole, C. Bonavides, S. Gama-Castro. The EcoCyc Database. *Nucleic Acids Research*, 30 (1), 56-58, 2002.

[40] I. M. Keseler, C. Bonavides-Martinez, J. Collado-Vides, S. Gama-Castro, R. P. Gunsalus, D. A. Johnson, M. Krummenacker, L. M. Nolan, S. Paley, I. T. Paulsen, M. Peralta-Gil, A. Santos-Zavaleta, A. G. Shearer, P. D. Karp. EcoCyc: A comprehensive view of *Escherichia coli* biology. *Nucleic Acids Research*, vol

37 (1), D464-D470, 2009.

[41] A. M. Maglia, J. L. Leopold, L. A. Pugener, S. Gauch, S. An anatomical ontology for amphibians. *Proc. Pacific Symposium on Biocomputing*, 12, 367-378, 2007.

[42] OBO Flat File Format Specifications. http://www.geneontology.org/GO.format.shtml

[43] Y. Kalfoglou, M. Schorlemmer. Ontology Mapping: The State of the Art. *The Knowledge Engineering Review*, 18 (1), 1-31, 2003.

[44] N. Choi, I. Y. Song, H. Han. A Survey on Ontology Mapping. *SIGMOD Record*, 2006.

[45] T. Catarci, M. Lenzerini. Representing and Using Interschema Knowledge in Cooperative Information Systems. *Journal of Intelligent and Cooperative Information Systems*, 1993.

[46] E. Rahm, P. Bernstein. A Survey of Approaches to Automatic Schema Matching *The VLDB Journal*, 2001.

[47] P. L. Foreym, C. J. Humphries, I. L. Kitching, R. W. Scotland, D. J. Siebert, D. M. Williams. *Tree-building Techniques. Cladistics: A Practical Course in Systematics.* Oxford University Press Inc., New York, 44-71, ISBN 0198557664, 1993.

[48] D. Fernandez-Baca. *The Perfect Phylogeny Problem.* Steiner Trees in Industry. Cheng, X., & Du, D. editors. Springer-Verlag, New York, LLC, ISBN 9781402000997, 2002.

[49] K. L. Alesandrini. Pictures and adult learning. *Instructional Science*, 13 (1), 63-77, 1984.

[50] R. N. Carney, J. R. Levin. Pictorial Illustrations Still Improve Students' Learning From Text. *Educational Psychology Review*, 14 (1), 5-26, 2002.

[51] National Aeronautics and Space Administration, Mars Mission. *NASA Images Suggest Water Still Flows in Brief Spurts on Mars*. 2006. http://www.nasa.gov/mission_pages/mars/news/mgs-20061206.html

[52] T. Rowe. Phylogenetic systematics and the early history of mammals. (F. S. Szalay, M. J. Novacek, & M. C. McKenna, Eds.) *Mammalian Phylogeny*, 129-145, 1993.

[53] T. Macrini. *Monodelphis domestica*. Digital Morphology, 2001. http://digimorph.org/specimens/Monodelphis_domestica/adult/

[54] M. Demarest, N. Simmons. *Cynopterus brachyotis*. Digital Morphology, 2003. http://digimorph.org/specimens/Cynopterus_brachyotis/head/

[55] M. Giger, H. MacMahon. *Image processing and computer-aided diagnosis*. Radiologic Clinics of North America, 34 (3), 565-596, 1996.

[56] C. B. Medeiros, F. Pires. Databases for GIS. *ACM SIGMOD Record*, 23 (1), 107-115, 1994.

[57] EQ for character matrices. http://www.phenoscape.org/wiki/EQ_for_character_matrices

[58] Phenotypic Quality Ontology (PATO). http://obofoundry.org/wiki/index.php/PATO:Main_Page

[59] *Understanding Evolution. Welcome to Evolution 101!* University of California Museum of Paleontology. http://evolution.berkeley.edu/evolibrary/article/evo_01

[60] *Virtual Paleontology Laboratory Glossary*. University of California Museum of Paleontology. http://www.ucmp.berkeley.edu/IB181/VPL/Glossary.html

[61] *A Dictionary of Earth Sciences*, Oxford University Press, 1999.

[62] *A Dictionary of Biology*, Oxford University Press, Market House Books Ltd., 2000.

[63] W. C. Wheeler, P. Cartwright, C. Hayashi. Arthropod phylogeny: A combined approach. *Cladistics* 9 (1), 1-39, 1993.

[64] Jess, the Rule Engine for the Java™ Platform. http://www.jessrules.com/

[65] K. Ilic, E. A. Kellogg, P. Jaiswal, F. Zapata, P. F. Stevens, L. P. Vincent, S. Avraham, L. Reiser, A. Pujar, M. M. Sachs, N. T. Whitman, S. R. McCouch, M. L. Schaeffer, D. H. Ware, L. D. Stein, S. Y. Rhee. The Plant Structure Ontology, a unified vocabulary of anatomy and morphology of a flowering plant. *Plant Physiology*, 143: 587-599, 2007.

[66] Plants Database at United States Department of Agriculture, Natural Resources Conservation Service. http://plants.usda.gov/classification_about.html

[67] P. H. Raven, R. F. Evert, S. E. Eichhorn. *Biology of Plants.* Worth Publishers, Inc., NY, pp. 775, 1986.

[68] Cypriniformes Tree of Life Project (CTOL). http://bio.slu.edu/mayden/cypriniformes/home.html

[69] I. F. Cruz, F. P. Antonelli, C. Stroe. AgreementMaker: Efficient Matching for Large Real-World Schemas and Ontologies. *International Conference on Very Large Databases* (Demo), Lyon, France, September 2009.

[70] K. M. Claeson, J. W. Hagadorn, K. Luckenbill, J. G. Lundberg. Anatomy of the Very Tiny: First Description of the Head Skeleton of the Rare South American Catfish Sarcoglanis simplex (Siluriformes: Trichomycteridae). *Palaeontologia Electronica*, 11 (2), 2008.

[71] Jena Semantic Web Framework. http://jena.sourceforge.net/

[72] SDB – A SPARQL Database for Jena. http://openjena.org/SDB/

[73] S. H. Tirmizi, J. Sequeda, D. P. Miranker. *Translating SQL Applications to the Semantic Web.* In Bhowmick, S.S., Küng, J., Wagner, R. (eds.), DEXA 2008. LNCS, 5181, 450-464. Springer-Verlag, Berlin, Heidelberg, 2008.

[74] Ontobrowser. http://www.cs.utexas.edu/~miranker/studentWeb/OntobrowserHomePage.html

[75] Morphbank. http://www.morphbank.net/

[76] G. Riccardi, A. Mast, D. Miranker, F. Cilloniz. OntoMorphBankSter: Image-driven Ontology and/or Ontology-driven Image Annotation. Proceedings of *TDWG*, 2008.

[77] Fishes of Texas Collection. http://www.fishesoftexas.org/

[78] AToL: Phylogeny of Spiders. http://research.amnh.org/atol/files/

[79] Hymenoptera – Assembling the Tree of Life. http://www.hymatol.org/

[80] M. J. Ramírez, J. A. Coddington, W. P. Maddison, P. E. Midford, L. Prendini, J. Miller, C. E. Griswold, G. Hormiga, P. Sierwald, N. Scharff, S. P. Benjamin, W. C. Wheeler. Linking of Digital Images to Phylogenetic Data Matrices Using a Morphological Ontology. *Systematic Biology*, 56 (2), 283-294, 2007.

[81] A. T. Schreiber, B. Dubbeldam, J. Wielemaker, B. Wielinga. Ontology-based photo annotation. *IEEE Intelligent Systems*, 16 (3), 66-74, 2001.

[82] E. Hyvönen, A. Styrman, S. Saarela. *Ontology-based image retrieval.* Helsinki Institute for Information Technology. Helsinki, Finland: HIIT Publications, 2002.

[83] C. Halaschek-Wiener, A. Schain, J. Golbeck, M. Grove, B. Parsia, J. Hendler. A

Flexible Approach for Managing Digital Images on the Semantic Web. In Proceedings of the *Fifth International Workshop on Knowledge Markup and Semantic Annotation*. Galway, Ireland, 2005.

[84] C. Halaschek-Wiener, J. Golbeck, A. Schain, M. Grove, B. Parsia, J. Hendler. Annotation and provenance tracking in semantic web photo libraries. *International Provenance and Annotation Workshop*. Chicago, IL, USA, 2006.

[85] K. Pastra, H. Saggion, Y. Wilks. Intelligent Indexing of Crime-Scene Photographs. *IEEE Intelligent Systems Special Issue on Advances in Natural Language Processing*, 18 (1), 55-61, 2003.

[86] K. Pastra. Image-Language association: are we looking at the right features? In Proc. of the *OntoImage LREC Workshop*, Genoa, Italy, 2006.

[87] K. Petridis, D. Anastasopoulos, C. Saathoff, N. Timmermann, I. Kompatsiaris, S. Staab. M-OntoMat-Annotizer: Image Annotation. Linking Ontologies and Multimedia Low-Level Features. In *KES 2006*, Part III, LNAI 4253, 633-640, 2006.

[88] C. Hudelot, J. Atif, I. Bloch. Fuzzy spatial relation ontology for image interpretation. *Fuzzy Sets Syst.* 159, 15, 1929-1951, 2008.

[89] J. Martinez. *Overview of the MPEG-7 Standard (version 5.0).* ISO/IEC JTC1/SC29/WG11 N4031, Singapore, 2001.

[90] M. Bertini, A. D. Bimbo, G. Serra, C. Torniai, R. Cucchiara, C. Grana, R. Vezzani. Dynamic Pictorially Enriched Ontologies for Digital Video Libraries. In *IEEE Multimedia*, 42-51, 2009.

[91] J. Fan, K. Barker, B. Porter, P. Clark. Representing roles and purpose. In Proceedings of the *1st International Conference on Knowledge Capture*

(Victoria, British Columbia, Canada, October 22 - 23, 2001), ACM, New York, NY, 38-43, 2001.

[92] R. Mizoguchi, E. Sunagawa, K. Jozaki, Y. Kitamura. The model of roles within an ontology development tool: Hozo. *Applied Ontology*, 1-21, IOS Press, 2007.

[93] N. F. Noy. Semantic integration: a survey of ontology-based approaches. *ACM SIGMOD Record*, 33(4), 65-70, 2004.

[94] O. Bodenreider, T. F. Hayamizu, M. Ringwald, S. D. Coronado, S. Zhang. Of Mice and Men: Aligning Mouse and Human Anatomies. In Proceedings of *AMIA Annual Symposium*, 61-65, 2005.

[95] P. Mork, R. Pottinger, P. A. Bernstein. Challenges in precisely aligning models of human anatomy using generic schema matching. In Proceedings of *MedInfo*, San Francisco, CA, 2004.

[96] Gene Ontology at Sourceforge. https://sourceforge.net/projects/geneontology/

[97] T. Berners-Lee. Semantic Web Status and Direction. Keynote at the *International Semantic Web Conference*, 2003.

[98] OBO2OWL Mappings. http://spreadsheets.google.com/ccc?key=0AqAZUdKw1IVHcFdOXzRzQnJkOWwxVW1uMUxOOFd1UVE&hl=en

[99] OBO-Edit Source Wiki. http://wiki.geneontology.org/index.php/OBO-Edit:_Getting_the_Source_Code

[100] W3C Recommendation. *OWL Web Ontology Language: Semantics and Abstract Syntax.* P.F. Patel-Schneider, P. Hayes and I. Horrocks, eds. 20 Feb, 2004. http://www.w3.org/TR/2004/REC-owl-semantics-20040210/

[101] C. Golbreich, I. Horrocks. The OBO to OWL mapping, GO to OWL 1.1!

Workshop on *OWL: Experiences and Directions*, Innsbruck, Austria, Jun 6-7, 2007.

[102] C. Golbreich, M. Horridge, I. Horrocks, B. Motik, R. Shearer. OBO and OWL: Leveraging Semantic Web Technologies for the Life Sciences. *ISWC/ASWC*, 169-182, 2007.

Vita

Syed Hamid Ali Tirmizi was born in Jeddah, Kingdom of Saudi Arabia, on October 4, 1981, the son of Syed Basit Ali Tirmizi and Syeda Fatima Safia. He received the degree of Bachelor of Science in Computer Science, with highest honor, from National University of Computer and Emerging Sciences (FAST), Islamabad, Pakistan, in 2003. In the following year he developed Radix Academics Suite for National University of Computer and Emerging Sciences as the lead software engineer.

In August 2004, he moved to the United States of America to pursue graduate studies, and began doctoral studies in Computer Science at The University of Texas at Austin. He received the Master of Science degree in 2009 from the Department of Computer Science at The University of Texas at Austin. He conducted his doctoral research under the supervision of Prof. Daniel P. Miranker.

Permanent Address: C-270, Block 6, Federal B. Area, Karachi, Pakistan

This dissertation was typed by the author.