A Web-based System for Representing, Retrieving, and Visualizing Analogies

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at George Mason University

By

Harry J. Foxwell
Master of Science in Applied Statistics, Villanova University, 1978
Bachelor of Arts in Mathematics, Franklin & Marshall College, 1973

Director: Daniel A. Menascé, Professor of Computer Science

Spring 2003
George Mason University
Fairfax, Virginia

DEDICATION

To my teachers who inspired and believed in me, and
to my family whose love sustained and supported me.

ACKNOWLEDGEMENTS

I thank my employer, Sun Microsystems, for financial support, and especially my former and current managers, Robert McCartin and Paul Tatum, for encouragement and flexibility in scheduling time for my courses and research. Deborah Lapeyre and Wendell Piez of Mulberry Technologies provided valuable advice in the use of XML and XSLT. Thanks to my doctoral committee, Nada Dabbagh, Peter Denning, Gheorghe Tecuci, and especially to my advisor, Daniel A. Menascé, for his patience and encouragement. Special thanks to Honglei Ruan for her excellent work on the Analogy Expression editor. And ultimately, thanks to my wife Eileen. I could not have completed this work without her love and support.

TABLE OF CONTENTS

LIST OF FIGURES

Abstract

A WEB-BASED SYSTEM FOR REPRESENTING, RETRIEVING, AND VISUALIZING ANALOGIES

Harry J. Foxwell, Ph.D.

George Mason University, Spring 2003

Dissertation Director: Dr. Daniel A. Menascé

Analogies are essential in human cognition, reasoning, learning, communication, and problem solving. They can have a profound and broad effect on how we view and understand our world. In this dissertation we design, implement, and evaluate a Web-based system for representing, retrieving, and visualizing human-conceived analogies that provides a medium and a common language for analogy practitioners to share their analogies. To accomplish this, we review the components of analogies, and develop a general representation of their structure. We then develop a compact XML content model of this representation for use in Web-based environments, and show that the model is capable of representing a wide range of human-conceived analogies. We demonstrate, using XSLT, several example methods for visualizing analogy expressions that use our model. We demonstrate methods for storing and retrieving such expressions, and

develop methods for ranking the retrieved expressions. We designed and implemented the MARVIN (Markup for Analogy Representation and Visualization for the InterNet) system to demonstrate these methods. A formative evaluation of the MARVIN system by analogy authors and end users was conducted; both author evaluators and user evaluators agreed that the MARVIN system analogy visualizations can assist them in their use of analogies, and that the system's ability to retrieve analogies and alternates is also of value.

1. INTRODUCTION

> *How do we ever understand anything? I think, by using one or another kind of analogy - that is, representing each new thing as though it resembles something we already know.*
>
> - Marvin Minsky

When Meriwether Lewis and William Clark returned in 1806 from their historic 18-month exploration of western North America, the most significant item they brought back was not the specimens of unknown plant and animal species, nor the accounts of the native inhabitants of that area. What they brought back, indeed, the primary purpose of their expedition, was a map, a map of hitherto unknown territory, a map for others to follow [VIAs, 1998].

Maps show the way, how to get from where you are to where you want to go. They represent key features and their relative locations, and allow you to orient yourself while traveling through unfamiliar territory. Lewis & Clark's map had a profound influence on the newly formed United States. It guided millions of settlers and explorers from the familiar land of the original Eastern colonies to the new lands of the West.

Maps can be created by explorers for others to follow. But they can also serve as reminders. They can document significant discoveries as well as blind alleys and wrong turns. They can be a valuable record of the exploration process.

Analogies are like maps of unfamiliar knowledge territories, created by the explorers of those territories for others to follow. They show you how to get from what you know to what you want to know. Like geographical maps, they represent key features -- concepts -- and their relationships, and they help you orient yourself while you are learning. And just as Lewis and Clark's map guided travelers from the known to the unknown, analogies have guided humans from existing ideas and assumptions to new knowledge. When Galileo wanted to lead people from the familiar Earth-centered world of Ptolemy to the unfamiliar Sun-centered world of Copernicus, he created a map -- an analogy map -- showing how the solar system of planets was like the Jovian system of satellites that he directly observed through his telescope [Galileo, 1632]. When Darwin tried to lead people from the creation-centered world of static biological species to the dynamic world of biological evolution, he created his famous analogical map showing how to get from the familiar process of animal and plant variation under domestication to the process of natural selection [Darwin, 1859].

This dissertation first presents a map of what might be called Analogy Land; describing the points of interest and the routes you must travel to visit them. We then provide you with mapmaker's tools -- what you need to create and share maps of your own explorations, and to understand the analogy maps of other explorers.

1.1 Why are analogies important?

Analogies pervade all human communication and learning. They occur in an extraordinary scope and variety, ranging from the simplest *ratio* form, such as "hand is to arm as foot is to leg," through extended analogical essays proposing that a computer is like a brain [von Neumann, 1958], the Internet economy is like the England railroad boom of the 1800s [Arthur, 2002], and the mind of an autistic is like a Web browser [Grandin, 2000].

Analogies are widely used when explaining ideas, especially in instructional contexts. Using analogies is "one of the core processes of cognition" [Forbus, 2001], and may be the *primary* process of all cognition and communication [Hofstadter, 2001]. Analogies are a key component of learning-by-example, or case-based reasoning [Schank, 2000], and are quite common in science education [Glynn, 1997], [Paris, 2000]. And, although analogies are generally

imperfect and may lead to incorrect mappings or conclusions, their power and utility come from their ability to quickly convey large amounts of information [Ferguson, 2002].

Journalists, lawyers, and other authors of explanatory and persuasive material use analogies to convey their ideas, to strengthen their arguments (much of legal arguments is by analogy to previous cases [Ashley, 1991]), and to connect to their readers' background knowledge and past experiences.

1.2 What is an analogy?

In practice, the definition of *analogy* varies widely among research disciplines and common usage. We define an analogy as

> a *proposed similarity mapping* between an unknown or unfamiliar set of related concepts – called the *target analog*, and a known set – called the *source analog*, presented for instructional or explanatory purposes.

An analogy describes a *perception* that two different objects or situations are similar at some abstract level. Because they are perceptions, analogies are highly personal and contextual. That is, what one person perceives and proposes as an analogy may or may not be understandable to another person, depending on its context and on the receiver's background knowledge.

Analogies assist in acquiring new knowledge by attempting to map the *structure* of existing knowledge to new situations [Gentner, 1983]. For example, in a widely known and often cited analogy, Ernest Rutherford in 1910 proposed that an unfamiliar idea – the structure of the atom, is *like* the structure of a presumably familiar object – the solar system [Bohr, 1922][Gentner, 1983]. In this example the atom is the *target* analog of the analogy and the solar system is the *source* analog. This analogy suggests that we try to *map* what we currently know about the source – the solar system's parts and their interrelations, to the target – the parts of the atom, allowing us to make plausible inferences and predictions and to form hypotheses about the target object. Some of these predictions and inferences may turn out to be wrong, but the analogy provides a useful structure, or *scaffold*, for generating and considering them. [Roblyer and Edwards, 2000][Bruner, 1986].

1.3 Why we want to use the Web to access analogy expressions

The World Wide Web (WWW) is a system for storing, retrieving, and visualizing information authored by people anywhere in the world. It is generally built upon widely accepted technology standards, and consequently its accessibility can extend globally. Emerging technologies such as the Extensible Markup Language (XML) [W3C, 2000] are now used to improve

and restructure Web-based information by allowing the separation of the content from the mechanisms of its presentation, thereby allowing multiple forms of access and display on diverse devices using the same source date. The Web and its technologies constitute an ideal environment for sharing and communicating human-conceived analogies given its ubiquity, global reach, and universal standards.

One of the most important consequences of the Web's growth and scope is the formation of communities of practice centered on personal, social, and especially professional interests. The Web provides the medium for communication, and XML supports the development of a common language for each community. Businesses, government agencies, and academic disciplines are using XML to develop languages for exchanging data, ideas, and documents within their respective communities. For example, mathematicians have agreed upon a common language for representing mathematical expressions on the Web [W3C, 2001], chemists use Chemical Markup Language [Murray-Rust, 1998], and the U.S. Department of Justice is developing Justice XML, a set of projects to enable representation and sharing of data on criminal activities, biometric information, and driving records [USDOJ, 2002].

Web-based communities of practice centered on education topics and issues have become an especially effective aid to teachers and students [Gordin and Gomez, 1996][Wenger, 1998]. Education community professional have also used XML to define a common language for exchanging information and data about curriculum structure and content using the Learning Object Metadata Standard [IEEE, 2003].

We suggest the need for a mechanism to represent, record, share, and retrieve human-conceived analogies in a structured and globally accessible form, for communities of analogy practitioners. Such web-based communities are already forming [Ruhl, 2002]. We therefore require a compact and general representation for analogies that supports their expression and visualization using standard Web-based technologies, and that is relatively easy to understand, author, and extend. Our approach is to provide an augmentation of existing Web documents that contain analogies rather than embedding contextual markup within those documents. This approach recognizes the immense volume of existing HTML-based Web content that will persist for many years, and provides a mechanism for content authors to include easily created structured contextual information about analogies described their Web documents.

1.4 What we want to do with analogies and why

In order to communicate about analogies using the Web, we need to represent them in a standard format, and provide tools to author, display, use, share, and reuse the analogies. Having recorded analogies in a standard format then permits the storage, retrieval, and comparison of analogies in educational and other explanatory contexts. The ability to reference multiple analogies for a given target has been shown to increase students' understanding of the target [Nottis and McFarland, 2001], and it has been recommended that teachers develop a repertoire of analogies [Thiele and Treagust, 1994]. Our system is therefore designed to assist analogy practitioners -- authors and users of analogies – providing a common language for representing, visualizing, and retrieving analogies, providing Web access to multiple alternate analogies, and providing an archive of interesting and useful human-conceived analogies.

Chapter 7 describes MARVIN (Markup for Analogy Representation and Visualization for the InterNet), a system that defines an XML-based representation for analogies and demonstrates how it can be used to represent and visualize analogy expressions, and to store, retrieve, and share such expressions in local and Web-based archives. Figure 1.1 shows the general architecture of the MARVIN system.

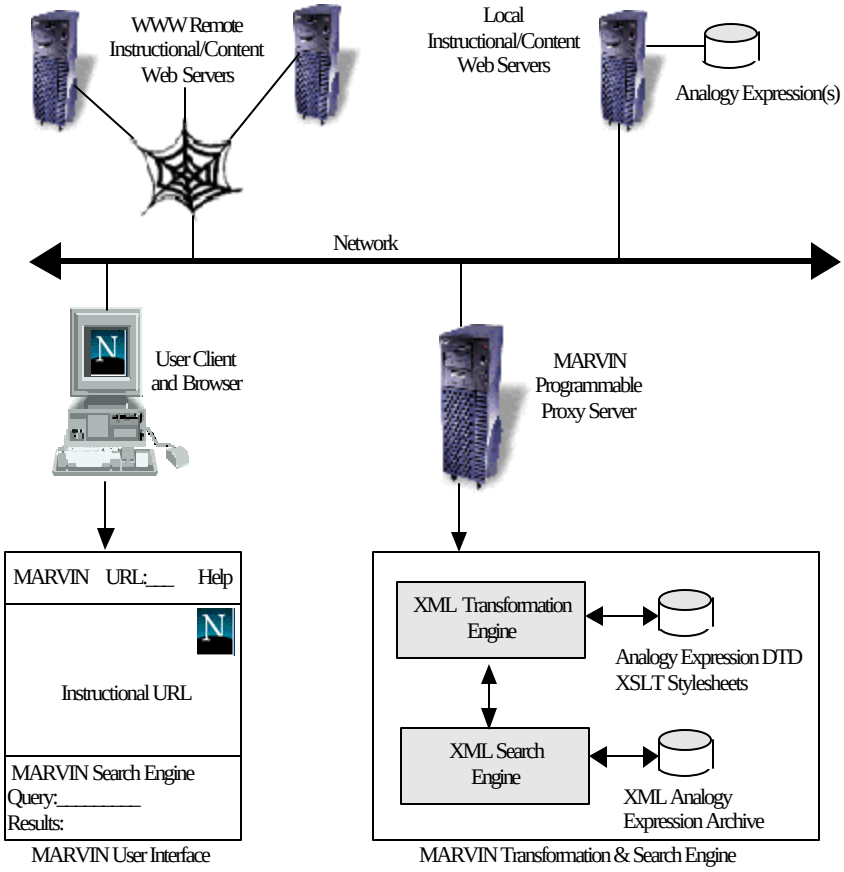


Figure 1.1 The MARVIN System Architecture

1.4.1 Representation

We require a compact and intuitive representation of analogies that is easy to author with generally available tools, and is capable of expressing much of the range of analogies that humans are able to generate. Such an expression must be consistent with current theory and research findings on analogy

components and structure, and must be programmatically useful to permit Web-based sharing, storage, retrieval, and manipulation of the expressions. The MARVIN system defines such an XML content model for the creation of analogy representations; with this model, analogy authors can create analogy expressions using XML editors or the Analogy Expression Editor described in Chapter 7.

1.4.2 Visualization

Visualization is a visual/spatial display in which information is communicated by the spatial arrangement of elements in the representation [Hegarty, 2002]. Such displays are cognitive aids that promote memory and information processing [Tversky, et al., 2002]; visualizations of analogies facilitate learning and enhance the use of analogies in both instruction and problem solving [Craig, et al., 2002], [Paris, 2000]. We require a method for producing multiple visualization forms from our analogy expression that separates the process of producing the visualization from the expression of the content and structure of the analogy, and that allows for display of these visualizations using standard Web browser technologies.

An analogy can be understood when the person hearing or reading it knows something about the source analog and is able observe and map the

components of the source to the target. Understanding of the analogy is significantly improved when the relational structures present in the analogy can be visualized in a tabular or graphical form [Paris, 2000][Matocha, Camp, and Hooper, 1998]. The MARVIN system provides this capability through the use of XSLT stylesheets, which transform analogy expressions into visualizations that can be displayed using standard Web browsers.

1.4.3 Storage/Retrieval

We wish to make interesting and useful human-conceived analogies storable and retrievable in a standard structured format for computation and expression. An archive of such expressions would permit the recording of analogies conceived during the development of problem solutions in order to document, replicate, and share the thought processes of the problem solvers [Dunbar, 2001]..

For those who wish to use analogies for instructional or explanatory purposes, a searchable archive of analogy expressions may be queried to locate an appropriate analogy for the topic under discussion. Or, if a proposed analogy is not understood by the learner or reader, the archive may be queried to locate additional or alternate analogies better suited to the learner's background knowledge. Education researchers have recommended

that teachers develop a repertoire of analogies for their instruction [Thiele and Treagust, 1994]; they have also found that presenting multiple analogies for a given target analog resulted in greater understanding of the target [Nottis and McFarland, 2001].

The ability to retrieve several analogies of possible interest implies the need for analogical ranking methods that may be used to order the results of a query. Such methods, discussed in Chapter 6, permit a learner to select candidate analogies most appropriate to the learning task. The MARVIN system was therefore designed to implement storage, retrieval, and ranking of analogy expressions.

1.5 The Focus of this Research

Analogy research historically has focused on several basic and overlapping areas – understanding the cognitive processes of analogical reasoning [Hofstadter, 1995][Marshall, 1999][Gentner, 1989][Falkenhainer, 1989], computer simulation of human analogical reasoning [French, 1995][Gentner, 1989][Forbus, Gentner, and Law, 1995], and using analogies in educational settings [Glynn, 1997][Schank, 2000][Paris, 2000]. While the computer simulation research has yielded great insight into the cognitive foundation of analogical reasoning, it is often limited to relatively small, well defined, and

easily represented areas of knowledge known as *microdomains*. *CopyCat* [Hofstadter and Mitchell, 1994] and *TableTop* [French, 1995] are examples of this approach. As noted in [Forbus, 2001], such systems represent and test limited forms of analogy-related tasks, and such systems "cannot possibly scale to handle the kinds of cognitive processing that human beings clearly do." The work presented here is focused on helping humans *share* and *use* analogies they have already conceived or discovered, rather than using computers to discover analogies or to perform analogical reasoning.

Human analogical reasoning typically involves the following processes [Holyoak, et al., 2001]:

- recall a source analog
- map the components of the source to the target
- generate plausible inferences about the target
- evaluate the inferences about the target
- accept (learn/remember) new knowledge about the target

The system described in this dissertation focuses on helping humans with the first two steps – retrieving source analogs already perceived and described by others, and visualizing the mappings between the source and target analogs, and the final step – remembering the analogy. Generating and evaluating

inferences suggested by the analogy remains the responsibility of the analogy author and users. That is, we are not concerned here with computer-aided analogy generation.

1.6 Overview of Contributions

The primary contribution of our work is the design and development of MARVIN (**M**arkup for **A**nalogy **R**epresentation and **V**isualization for the **I**nter**N**et), a prototype Web-based system that enables authors and users of instructional content to record, retrieve, visualize, and query human-conceived analogy expressions. We demonstrate the usefulness of the system through a formative evaluation process by content experts and authors, and by end users. Additional contributions associated with the development of the MARVIN system are discussed below.

We developed a compact, general representation of analogy expressions, using an XML content model for universality and Web-based access, and demonstrated through examples the power of this representation to express analogies from varied domains, such as science, history, medicine, religion, and literature.

We developed multiple visualization methods, such as tabular and graphical

displays, which are generated *directly* from the XML analogy representations using the Extensible Stylesheet Language (XSL) [W3C, 1999], and demonstrated a variety of example visualizations.

We developed Web-based methods for storage and retrieval of analogy expressions, and demonstrated examples of retrieving alternate analogies and ranking the retrieved expressions.

The above contributions use our model of the components and structure of analogies. This model is consistent with both common usage and formal characterizations of analogies from cognitive science research, from educational practice and research, and from work on Web-based knowledge representation.

1.7 Dissertation Organization

The remainder of this dissertation is organized as follows:

Chapter 2 discusses the various Web technologies used in the design and implementation of the MARVIN system, and reviews prior research on analogies by computer scientists, cognitive scientists, and education researchers.

Chapter 3 reviews and defines the components of analogies, discusses the basic structure of analogies, and introduces a graphical representation for several key structures commonly found in analogies. This chapter also presents and discusses several example analogies from various domains.

Chapter 4 discusses the need for a general-purpose representation for analogies, discusses design goals for such representations, presents a formal definition for analogy expressions, and implements that definition using an XML content model.

Chapter 5 discusses how analogies that are expressed using our XML content model may be visualized in several forms using Web-based tools such as XSL Transformations (XSLT) [W3C, 1999] and Scalable Vector Graphics (SVG) [W3C, 2001], discusses design goals for such visualizations, and presents several example visualizations.

Chapter 6 discusses the storage and retrieval of analogy expressions, and introduces and demonstrates methods for ranking the results of queries of analogy expression archives.

Chapter 7 discusses the design goals of the MARVIN system for recording,

retrieving, and visualizing analogy expressions that use our XML model and stylesheets, and describes the components and system architecture. Performance characteristics of the MARVIN system are also discussed. The Analogy Expression Editor, used to aid authors in creating XML analogy expressions, is also described.

Chapter 8 presents the results of formative evaluations of the MARVIN system by content authors and individual users.

Chapter 9 provides a summary of the dissertation, and discusses future research suggested by this work in the area of analogy representation, visualization, and retrieval.

2. Background

2.1 Web Technologies Used in Our Research

The work described in this dissertation employs several technologies developed by the World Wide Web Consortium (W3C) that were designed to provide structure and meaning to Web-based content and to provide programming languages and interfaces to access that content. These technologies include XML (Extensible Markup Language), XSLT (Extensible Stylesheet Language for Transformations), and related tools and languages such as Xerces, XPath, Xalan, Apache Tomcat, Jakarta Lucene, Apache Cocoon, and HTML. The following sections provide overviews of these technologies. The section 2.2 of this chapter reviews research concerning analogies.

2.1.1 XML

XML (Extensible Markup Language) is a meta-language designed to provide a "universal format for structured documents and data on the Web" [W3C, 2000]. Virtually all Web pages are currently written and formatted using

HTML (HyperText Markup Language), which was specifically designed for presentation and display of Web content [W3C, 1999a]. But HTML has no capability for attaching meaning or interpretation to the content. Using XML, content authors can create special-purpose descriptive languages that can be used to "tag" content structure and components. Such languages enable communities of practitioners to use a common language through the Web for effective sharing of data and ideas.

XML tags are unique, case-sensitive labels for sections of content, delimited by angle brackets. Tags are used to encapsulate content elements and give them meaning. This enables search engines and other Web tools to reduce ambiguity in the search space. For example, tagging the word "Brown" in a Web document listing personal information, using `<LastName>Brown</LastName>` or `<HairColor>Brown</HairColor>` permits a search engine or other program to distinguish between "Brown" the name and "Brown" the color.

The XML specification describes strict rules for the tagging of content in order to simplify document parsing and to eliminate ambiguity. Content elements have start-tags and end-tags; an XML document is *well formed* if elements delimited by start-tags and end-tags nest properly within each other (that is, `<A><B>text</A></B>` is not permitted, while

`<A><B>text</B></A>` is properly nested). All XML documents must be well formed.

XML tags may be defined within the document content file itself, or may be defined in an associated file using the XML DTD (Document Type Definition) specification [W3C 2000] or the XML Schema specification [W3C, 2001]. A *valid* XML file uses *only* the tags defined in its DTD or Schema file, and must also be well formed.

2.1.2 XML Editors

The MARVIN system requires the creation of XML files to represent analogies. There are numerous commercial and open-source XML editors, such as XML-Spy [XML-Spy, 2002], MS XML NotePad, Morphon [Morphon, 2002], and epcEDIT [epcEDIT, 2002] to name only a few. Such editors enable the creation of valid, well-formed XML files that conform to a DTD or schema. Analogy authors using the MARVIN system can elect to use such editors to create analogy expression files, but using such editors requires knowledge of XML structure and syntax. We therefore designed and implemented an Analogy Expression Editor, written in Java, for use with the MARVIN system, which permits the creation and modification of XML files that conform to our content model, without requiring the author to know any

XML. This editor is described further in Chapter 7.

2.1.3 Java

The Java programming language [Joy and Gosling, 2000] is ideal for Web-based applications because of its rich network APIs and its ability to run on diverse operating systems and architectures. Most of the technologies used in this dissertation use Java directly or indirectly, because they are written as Java applications or as Java *servlets* [Horstmann and Cornell, 2000][Sun, 2003]. Servlets are used to extend the capabilities of Web servers by enabling programmers to produce interactive, dynamic Web content based on user actions and data content. Servlets run within a *container* that manages the servlet's interaction with the server's operating system and Web server. Servlet containers are typical components of commercial and open-source Web and application servers, and can also be implemented as stand-alone Web services.

2.1.4 Jakarta Tomcat

Jakarta Tomcat is an open-source Java servlet container that is the official reference implementation for Java Servlets [Apache, 2002]. It can be integrated with Web servers such as Apache [Apache, 2002], or run as a stand-alone Web service. The MARVIN prototype system described in

Chapter 7 is implemented using Jakarta Tomcat to run the XML transformation and search servlets.

2.1.5 Apache Cocoon

Apache Cocoon [Apache, 2002] is an XML Web publishing framework that runs as a servlet within Apache Tomcat. It enables the development, management, and generation of dynamic Web content from XML source documents. It permits the separation of the *representation* of Web content from the processing necessary to generate multiple forms of display. The MARVIN system uses Cocoon to transform XML representations of analogies into a variety of visualization forms, and to interface with the retrieval and text search servlets.

2.1.6 XML Parsers

XML documents must be read and interpreted according to the XML specification. A program that performs this task is called an XML *parser*. A parser that enforces a document's compliance with a DTD or schema is called a *validating* parser. There are several commercial and open-source validating XML parsers available. For the work described in this dissertation, we use the *Xerces* Java Parser [Apache, 2001], which supports the XML 1.0 recommendation [W3C, 2000]; the Apache Cocoon servlet used

in the MARVIN system uses *Xerces* to parse the XML analogy expressions.

2.1.7 XSLT

XSLT (Extensible Stylesheet Language for Transformations) is a language for transforming XML documents into other forms, including PostScript, Adobe PDF, HTML, Java, and alternate XML representations [W3C, 2001]. XSLT is also a specification for such transformations. It is partially implemented in some browsers such as [Mozilla, 2002], but these implementations are still immature and buggy. A more mature and complete XSLT processor is Xalan [Apache, 2002]. The Xalan processor operates on a parsed XML document and transforms it according to instructions contained in a *stylesheet* file. These instructions, called *templates*, specify transformations to be applied to selected node elements of the parsed XML document [Kay, 2001].

The XSLT programming model is not procedural, driven by the program code. Rather, it is event-oriented, driven by the data, in this case by the XML document. When the XSLT processor reads the parsed XML document, it detects document node match events and processes the node data according to the template defined for that node.

XSLT templates may be thought of as independently selectable processing instructions that “bind” to a node in the XML document when a match is encountered, analogous to the way messenger RNA binds to a segment of DNA during cell reproduction [Piez, 2002], [Foxwell, 2002]. Like the messenger RNA provides instructions for constructing a specific protein, the template provides instructions for constructing a specific component of an output document. The Apache Cocoon servlet used in the MARVIN system uses the *Xalan* XSLT processor.

2.1.8 XPath

XPath is an XML language specification for referencing nodes of a parsed XML document [W3C, 1999b]. Its syntax is similar to that for computer file system directories, and permits reference to a document's element and attribute nodes, and to their parent and child nodes. XSLT templates contain XPath references to document nodes, and the instructions for processing the nodes. XSLT processors such as Xalan make use of XPath when referencing XML document nodes for the templates to process.

2.1.9 SVG

SVG (Scalable Vector Graphics) is a language for describing two-dimensional graphics in XML [W3C, 2002], allowing the generation of lines, curves, images, and text from an XML document specification. SVG provides a

compact and portable means for generating resizable and searchable Web based images [Eisenberg, 2002]. Browsers such as MS IE and Mozilla are beginning to support the display of SVG graphics directly, but as with XSLT, these are still early implementations and do not completely support the full SVG specification. However, there are tools for converting SVG graphics to browser-displayable GIF or JPG format graphics. The Cocoon servlet used in the MARVIN system includes an SVG-to-JPG transformer; the SVG analogy visualizations produced by the MARVIN system can thus be converted to JPG format and displayed on any graphics-capable Web browser.

2.1.10 Querying XML Documents

There are various tools under development for searching the contents of XML documents. XQuery [W3C, 2002] is an XML query language currently under development by W3C, but the specification for this language is still in Working Draft status, and at this time there are few complete implementations. On the other hand, there are text search engines that can make use of XML markup in documents, providing the ability to perform a structured search for text within XML tagged fields. One such "structure-aware" search engine is Jakarta Lucene [Apache, 2002], written in Java, and implemented as a Java servlet. Lucene allows queries based on XML tag content; the MARVIN system uses Lucene as its search and retrieval

component.

2.2 Related Research

Research concerning analogies occurs in many disciplines. Computer scientists in the fields of Artificial Intelligence and Machine Learning build systems that attempt to model analogical reasoning; cognitive scientists also build such systems to investigate the underlying mental processes involved in memory, analogy perception, and analogical reasoning; and educators study the use, and abuse, of analogies in teaching. Because the computer is often used as a surrogate for studying and modeling the human mind, there is significant overlap between cognitive science research and computer science research into the workings of analogies.

Additionally, historians analyze and debate the usefulness of historical analogies [Neustadt, 1988][Rourke and Taylor, 1995], and legal scholars and practitioners make extensive use of analogies in legal arguments [Ashley, 1991]. Analogies are also quite common in journalism and editorial writing, although their overuse and oversimplification of important ideas has been criticized [Clark, 2002].

In short, analogies are found in nearly all areas of human communication

and learning, and are widely studied in the scientific and social disciplines. Communities of analogy practitioners and researchers use the Web extensively to share examples, ideas, and research results. The following sections review analogy research in Computer Science, Cognitive Science, and Education.

2.2.1 Analogy Research in Computer Science

Among the prominent early researchers in computer learning and reasoning by analogy is Patrick Winston of the Artificial Intelligence Laboratory at MIT [Winston, 1980]. He designed a LISP-based Frame Representation Language (FRL) system derived from Minsky's knowledge representation *frames* [Minsky, 1985]. FRL was applied to finding analogous story plots in literature samples that were already expressed in concept/relation structures. It used weighted matching criteria to compare concepts and relations, giving high weights to those that were of particular importance to plot structure. While Winston was able to find sub-plot analogies between sections of Shakespeare's *Hamlet* and *Macbeth*, for example, this research focused on finding analogies that were already known and in limited knowledge domains.

Much analogy research explores relatively small, well-defined, and easily represented areas of knowledge known as *microdomains.* The ANALOGY program [Evans, 1968], for example, examined spatial analogies among geometric shapes. Decades later, microdomains such as *Copycat* [Mitchell, 1999], *Tabletop* [French, 1995], and *IDA* [Wolverton, 1994] continue to yield insight into the cognitive processes that cause the perception of analogies. *Copycat*, for example, examines analogies between character strings, and *Tabletop* generates action analogies using common objects on a kitchen table. Wolverton's *IDA* sought to generate engineering design analogies, and focused primarily on finding an efficient algorithm for searching a predefined problem space ontology.

Some attempts at computer modeling of analogy retrieval use concept indexing, spreading activation, and network-matching approaches [Collins and Loftus, 1975]. These techniques were useful in finding *semantically close* analogies (target and source analogs taken from the same specialized knowledge domain), but they were not scalable to finding semantically distant analogies across multiple large knowledge domains (the type humans are good at creating). [Wolverton, 1994] proposed a refinement of spreading activation, called Knowledge-Directed Spreading Activation, consisting of multiple network-matching search agents that would persist when concept

map fragments were matched and would reduce or stop activation when no fragment matches were found. Hofstadter and Mitchell's *Copycat* program [Mitchell, 1999] used a similar agent-based approach, with software agents performing multiple, random searches through the source knowledge archive; each agent would gain or lose resources for continued searching according to its success in finding candidate concept matches.

Because analogies are perceptions that need to be tested against reality, human interpretation and expertise should be used to validate analogies proposed by a computer. The Disciple system [Tecuci 1998], for example, suggests problem solutions derived using analogy to a human expert, who accepts or rejects them and provides an explanation for the decision. The method used in Disciple illustrates an important idea in searching for analogies: replacing a concept in a knowledge expression with a *generalization* of the concept. Hofstadter calls this *variablization*, and Mitchell calls it *conceptual slippage* [Hofstadter, 1995]. A key characteristic of this process is that the *relations* among the concepts in the knowledge expression are preserved. [Gentner, 1983] calls this preservation of relational structure the *systematicity principle*.

Another research effort in the modeling of human reasoning is the Cyc Project [Lenat and Guha, 1990]. Cyc is a large, general purpose knowledge

base containing concepts and “common sense” assertions that can be queried using a natural language interface. Cyc attempts to provide enough background knowledge to a computer so that it can reason about the world, learn (including by analogy), and ultimately model human intelligence. The project has been commercialized, but there is also an open source version available [OpenCyc, 2002].

XML is increasingly being used as the core language for knowledge representation in Artificial Intelligence (AI) research, because of its flexible representational capabilities and the wide availability of XML-aware programming tools [Popov, 2000].

2.2.2 Analogy Research in Cognitive Science

Analogy research tends to take one of two approaches, either investigating the processes of identifying analogies in *existing* cognitive representations, as in Gentner's structure-mapping theory [Gentner, 1989a], or investigating the cognitive *mechanisms* of analogy *construction*, as in Hofstadter & Mitchell's work on high-level perception [Hofstadter, 1995], [Mitchell, 1999]. In the former, the focus is on programs that “discover” preselected analogies, while the latter focus on constructing new or unknown analogies. Schank's research on case-based reasoning also starts with preselected analogies used

to enhance learning [Watson, 1994], [Schank, 2000].

The Structure Mapping Engine (SME) [Gentner, 1983a] is a widely cited example of an “analogy finder.” It emphasizes the relational structure of analogies, first generating all possible mappings between the concepts of a target analog and a potential source analog, then preferentially selecting mappings based on similar relational constructs in the target and source. SME is primarily used to validate Gentner's Structure Mapping Theory (SMT) of cognition, and it is tested on known analogies such as the Rutherford analogy (atom / solar system), and heat flow / water flow [Gentner, 1983b].

SME is used in the MAC/FAC model [Forbus, 1995] to further improve analogy retrieval. The first step, MAC (Many Are Called) performs a broad search of similarity based matching of a target analog with potential source analogs, followed by FAC (Few Are Chosen) in which the SME is called multiple times to find the best structural match. Like SME, MAC/FAC is used to validate the results of psychological experiments on similarity based and relation based memory retrieval.

Large-scale knowledge bases are under development which use efficient and scalable analogical reasoning tools such as MAC/FAC. These tools and

knowledge bases are focused on creating general-purpose reasoning and problem-solving systems that are equivalent to human reasoning [Forbus, Mostek, and Ferguson, 2002].

2.2.3 Analogy Research in Education

Analogies are used extensively in education, especially in the teaching of Science [Glynn, 1998], [Schank, 2000]. Analogies in instructional settings are powerful tools for connecting the learners background knowledge to the topic to be learned. They provide a structure for thinking about the topic, preparing the learner an *advance organizer* for their prior learning [Ausubel, 1960], and a supporting structure, or *scaffold*, for generating inferences about the topic [Bruner, 1986].

In a study particularly relevant to our research, [Paris, 2000] showed that the explicit visualization of scientific analogies enhanced the *students*' interest and understanding of the subject matter, and improved their inference-making and metacognitive skills. Additionally, the study showed that the creation and use of elaborate analogies (text descriptions accompanied by explicit representations of the similarities between the target and the source) also improved the *teachers'* interest and comprehension of the subject matter, and thereby improved the effectiveness of their teaching.

The presentation and visualization of analogies for instructional purposes have been shown to significantly enhance learning and retention, particularly for novel or complex topics [Matocha, Camp, and Hooper, 1998], but educators recognize the limitations and dangers of misleading analogies. Some have attempted to develop metrics for analogical validity [Nottis and McFarland, 2001], while most have emphasized the need, during the analogical reasoning process, to indicate where analogies break down [Herr, 2001].

2.2.4 The Semantic Web and Knowledge Representation

At present, most of the documents on the World Wide Web are visual and textual, marked with HTML tags for browser display formatting and for ease of navigation among documents. Tim Berners-Lee, credited with inventing the Web, envisions a richer form of content he calls the *Semantic Web* [Berners-Lee, Hendler, and Lassila, 2001], [W3C, 2002]. In the Semantic Web, every object (word or image) in a document is labeled with its meaning and context, and is linked to related objects according to the purpose of the document. Such labeling and linking, using XML metadata, may eventually enable intelligent software agents to access, interpret, and transform all Web content for other software agents as well as for human users.

RDF (Resource Description Framework) is a framework that supports the Semantic Web project for describing and exchanging data about objects represented on the Web [W3C, 2003]. It describes all Web content in terms of resources (object location), properties (author, title, domain), and associations among resources, and is focused on providing computer-searchable metadata for identifying and locating resources. It presumes, however, that eventually all Web content will be re-created or at least annotated using RDF codes; the process and standards for this effort are still under development, and there is concern that authors of Web content will find the required markup to enable the Semantic Web vision too burdensome [Suter, 2003].

Topic Maps [TopicMaps.Org, 2001] are a related Web technology explicitly designed to represent and display associations among terms within a document, permitting a content author to link pairs or groups of concepts and to describe the nature of the relationships among the concepts. Topic maps and Semantic Web technologies both require substantial markup *within* a Web document, however, and the resulting concept link structure is generally *unique* for each document. In this dissertation, we focus on analogies only, and present a general approach to representing the concepts and relations that compose an analogy, providing a compact markup structure for

representing a wide range of analogies, and expressing that structure as a link *attachment* to *existing* Web content. We will show in Chapter 5 that this approach allows for representations of analogical concept relationships that can be transformed into several forms using XSL stylesheets, including tabular structures, graphical visualizations, and other knowledge representations including Topic Maps.

As noted in [Dunn, 2002], the effort to provide useful and general context and meaning markup for Web documents is an enormous task, still too difficult for average users. The result of this difficulty, along with the decentralized nature of the Web, encourages communities of practitioners to take a simpler approach, developing their own ontologies and metadata [Staab, 2002].

2.2.5 The MARVIN System

The MARVIN system described in this dissertation is designed to provide a Web-based, common-language environment for describing and sharing analogies already conceived by teachers, scientists, journalists, doctors, and other analogy practitioners. It is therefore directed at communities of analogy *users*, especially, but not exclusively, educators. And while it is not explicitly designed for interaction with the analogical reasoning engines used

in cognitive science and computer science, the analogies produced by those systems can be represented and stored using the MARVIN system. Thus MARVIN may be useful to researchers in computer science, cognitive science, and education, by providing an environment for capturing and sharing interesting or unusual analogies, whether produced by humans or by machines.

3. The Structure and Components of Analogies

What is considered to be analogy or analogical reasoning varies substantially among practitioners and researchers in various fields, particularly education, cognitive science, and computer science. While most researchers agree that the mapping of *relational* structures is the defining characteristic of analogies, some use the term *analogy* more broadly to include simple mappings of concepts or of similar properties. We include here several examples from the analogy literature to illustrate and develop a terminology for the structure and components analogies. We revisit these examples in Chapters 4, 5, and 6 to illustrate representation, visualization, and retrieval, respectively. Additional examples may be found on the author's website [Foxwell, 2002] and in the Appendix.

The word “analogy” as used in both general language and in the research literature refers to a *perceived* level of similarity or sameness between the observed properties, concepts, and relations of two knowledge domains, one assumed to be known, the other partially known or unknown.. A somewhat restrictive definition of analogy by Gentner in [Vosniadou and Ortony, 1989]

that emphasizes the importance of relations, states:

> "...an analogy is a mapping of knowledge from one domain (the base) into another (the target), which conveys that a system of relations that holds among the base objects also holds among the target objects...in interpreting an analogy, people seek to put the objects of the base in one-to-one correspondence with the objects in the target..."

The *objects* in the above definition may be words, sounds, images, processes, or other symbols representing perceived concepts, and the relations among them.

In this dissertation, we define an analogy as

> a set of proposed similarity mappings between an unknown set of concepts and relations (the *target* analog) and a known set of concepts and relations (the *source* analog), used for instructional or explanatory purposes.

The definition of *concept* depends on the context for its use. A concept may be

viewed as a set of instances or examples along with categorization rules for determining set membership [Tecuci 1998]. Others, [ISO, 1990], define a concept as "a unit of thought constituted through abstraction on the basis of properties common to a set of objects." For the research presented here, we require an *operational* definition of concept that provides representational and programmatic simplicity for both developers and especially for content authors who are not expert at semantics and word ontologies. We therefore represent a concept as the *word* or *words* a content author chooses to describe the concept. This is consistent with the view that a concept is a mental representation and that a word is an expression of that representation [Murphy, 2002]. The base form of the word can then be used as a key to interface with lexical databases, such as WordNet [WordNet, 2001], or with conceptual graph tools such as WebKB [Martin and Ecklund, 1999], in order to obtain a better understanding of the meaning and context for the concept word. Similarly, we represent a relation as a word chosen by an analogy author to represent an association between two objects.

Analogies are perceived correspondence *maps* between the respective components and structures of a familiar source analog and an unfamiliar target analog. There are three ways in which the perceptions of objects or collections of objects may be placed in correspondence [Collins and Burstein,

1989]. The simplest type of perceived correspondence is that of properties. Two objects may appear to have the same color, shape, or size, for example. Another type of perceived correspondence is that of concepts. Two collections of objects may have the objects paired according to the perceived concepts represented by the objects. For example, when explaining the structure of the atom to a science student using Rutherford's analogy, we map the concept *nucleus* to *sun*. The third type of perceived correspondence is *system correspondence*, also called *systematicity* [Gentner, 1983]. In this type of correspondence, the relational structures among objects in the target analog are mapped to similar or identical relational structures in the source analog. Using the atom example again, the electron is hypothesized to *orbit* the nucleus as a planet *revolves* around the sun. *Orbit* and *revolve* are the respective relations between the components of the atom and the components of the solar system; explaining atomic structure using an analogy to the solar system places these *relations* in correspondence.

We use these correspondence types to define an *analogy expression* as a collection of *correspondence maps*, which associate the concepts, properties, and relations of the target with those of the source. The systematicity principle additionally specifies that analogies must preserve *higher order relations* such as causality and implication when mapping the components of

the source to those of the target. Using the Rutherford analogy, for example, we can then form (and test) a hypothesis that electrical attraction *causes* the electron to orbit the nucleus in the same way that gravitational attraction *causes* the planet to revolve around the sun [Wilson, et al., 2001]. That is, the proposed analogy preserves in the target the higher order *causal* relationship perceived in the source.

3.1 Analogy Examples

In addition to the Rutherford analogy discussed above, we now examine several additional examples to illustrate the types of analogy structures that can occur. Note that our goal is to *describe* the analogy as its author presents it, without attempting to evaluate the correctness or completeness of the proposed comparisons.

The Altoona List of Medical Analogies [Ruhl, 2002] lists a simple analogy comparing the eye to a camera in order to explain certain types of vision problems to patients. It first establishes a mapping of the parts of the eye to those of a camera, and then explains that a cataract in the eye is like a fogged lens in a camera. Continuing with the analogy, it explains that a detached retina is like a camera with wrinkled film. Once this analogy is established, the eye doctor can continue, perhaps with additional analogies, explaining

the necessary procedures for treating the conditions. As presented, this analogy is primarily a mapping of known concepts – camera parts and their presumably understood functions, to unfamiliar concepts – eye anatomy and vision impairments. Figure 3.1 shows a representation of this mapping.

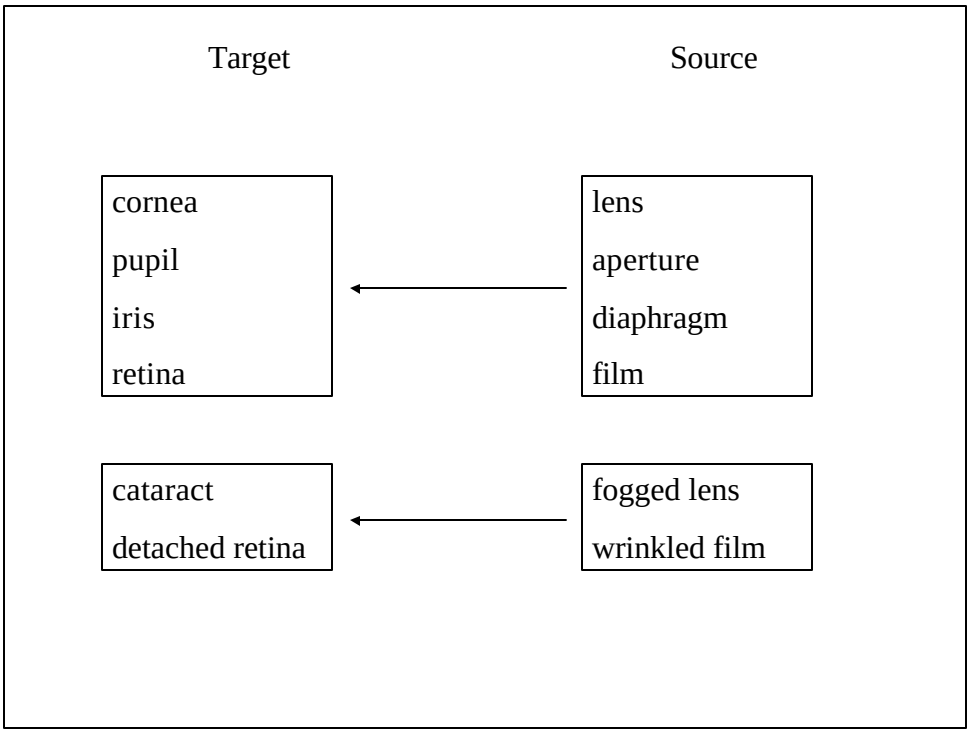


Figure 3.1. The Eye/Camera Analogy

Upon observing the moons of Jupiter, Galileo formed an analogy between the solar system and the Jovian system, proposing that the *relationship* of the planets to the Sun was the same as that of the moons to Jupiter [Galileo,

1632] – that the planets, *including the Earth*, orbit the Sun like the moons orbit Jupiter. This comparison illustrates a fundamental characteristic of analogies – the mapping of *relations*, as shown in Figure 3.2.

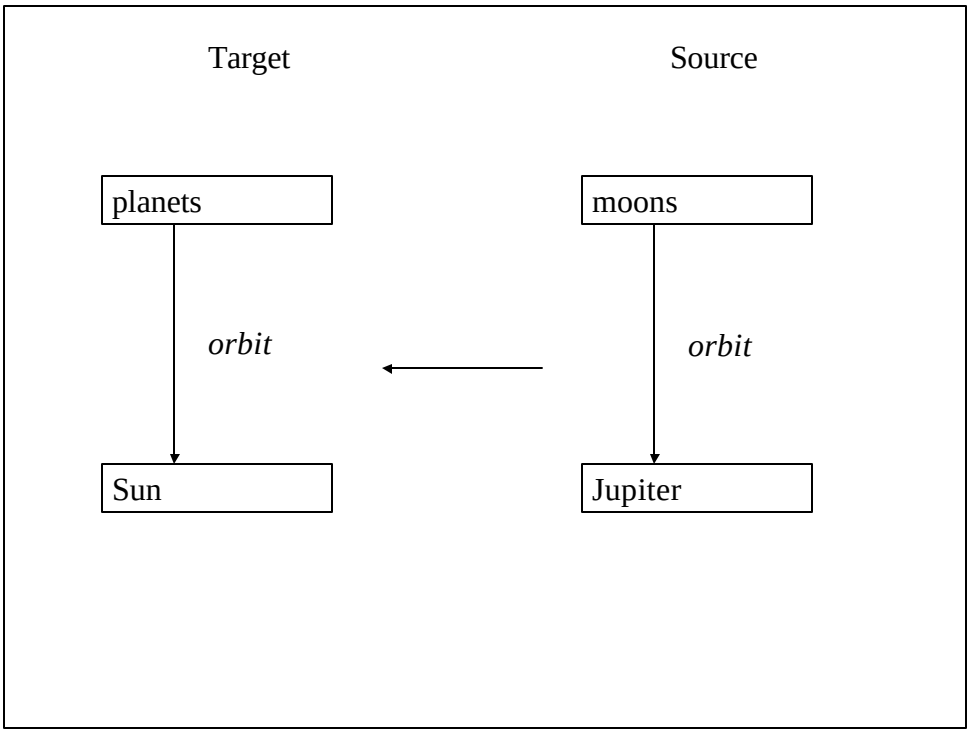


Figure 3.2. Galileo's Solar System Analogy

Analogies that map higher order relations such as causality can provide significant insight into the target analog. In the previously discussed Rutherford analogy, gravitational force as a *cause* of the planets' motion around the Sun is mapped to electrical force as a hypothesized *cause* of electrons “orbiting” the nucleus. In this analogy, a source concept is therefore

associated with a source relation, and a similar relational structure is proposed in the target analog, as shown in Figure 3.3.

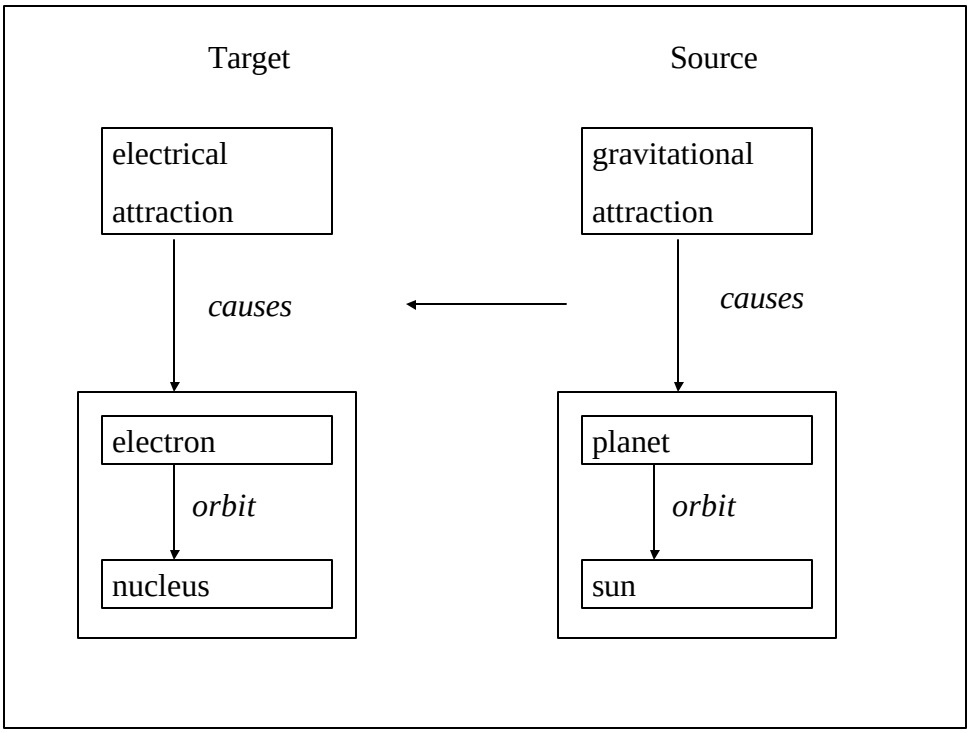


Figure 3.3. The Rutherford Analogy

We see a similar type of structure in another example, the Bohr Liquid Drop model of Nuclear Fission [Koushiappas and Cohen, 1999]. In this analogy, we also see the mapping of a causal relationship – when the Coulomb repulsion between the two halves of a deformed liquid drop is *greater than* the surface tension between the two halves that relationship *causes* the drop to split. The analogy proposes that the same type of causal relationship holds

for the nucleus – that when the Coulomb repulsion between two halves of a deformed nucleus exceeds the binding energy between the halves, this *causes* the nucleus to split. Moreover, the formulae for the calculation of the forces are also proposed to be analogous. In this example, shown in Figure 3.4, we see the structure of a relation (Coulomb repulsion *greater than* surface tension) mapped to a concept (fission). This is similar to the structure we saw in the Rutherford analogy, but with the directionality of the higher order relation reversed.

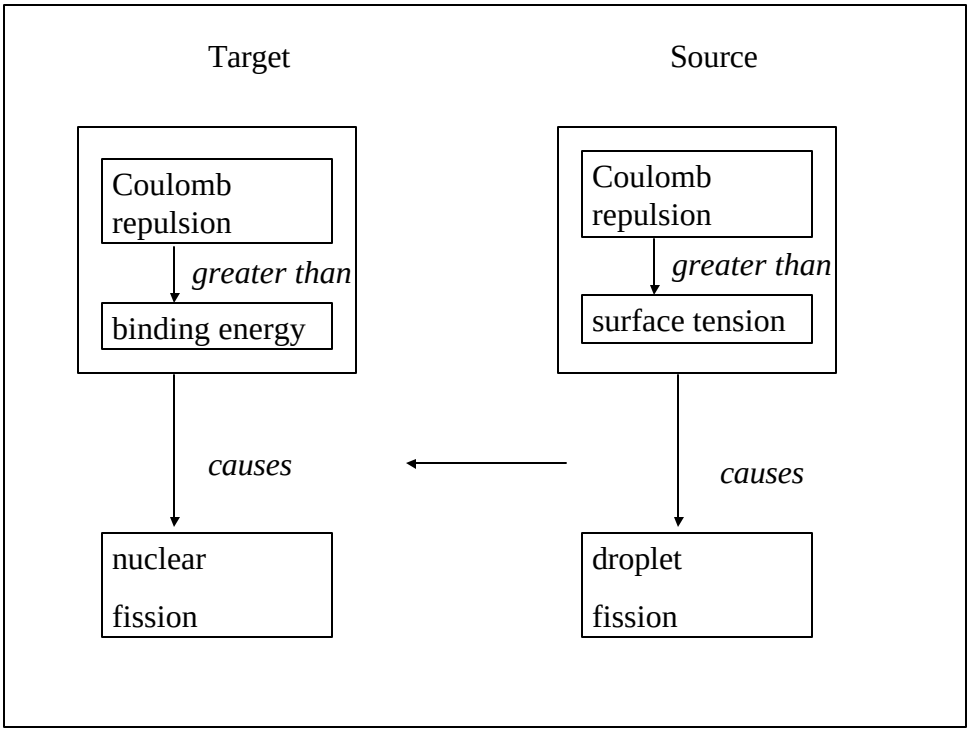


Figure 3.4. The Bohr Liquid Drop Model of Nuclear Fission

Historical analogies can be created and used by politicians and journalists to sway public opinion or to suggest the inevitability of a decision or course of action. The September 11, 2001 attack on the World Trade Center and Pentagon has been compared to the Japanese Navy's attack on Pearl Harbor in 1941 [Cox, 2002]. Without evaluating the merits of the analogy, we see that the analogy can be partially expressed as a higher order relational map. The target consists of an action (al Qaeda attacks WTC) causing (or implying a desired decision) an action (US declares war on terrorists); the source is similarly structured – (Japanese Navy attacks Pearl Harbor) causing (or implying a desired decision) an action (US declares war on Japan). Figure 3.5 illustrates this map.

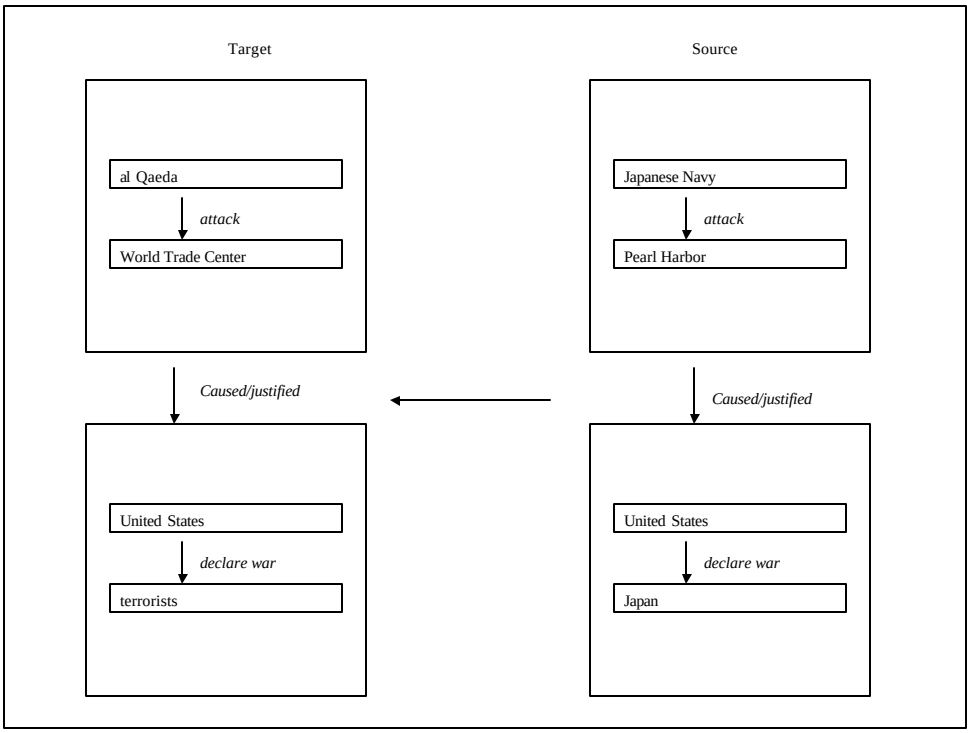


Figure 3.5. Historical analogy – September 11 = Pearl Harbor

We observe that although analogies may be quite elaborate or complex [Arthur, 2002], [Grandin, 2000], they may be decomposed into a small number of structured component types such as those illustrated above within the target analog, each of which maps to an identically structured component within the source analog. We now provide in what follows the definitions of each of five types of components that can appear in either the source analog or target analog:

Definition 1 (ConceptSet): A *ConceptSet* $\boldsymbol{C} = \{c_1, c_2, \ldots, c_n\}$ is an ordered set of one or more concepts. Note that the order of the elements listed in a

ConceptSet is significant since we map corresponding concepts in the target and source. In the Rutherford analogy, for example, a possible target ConceptSet is {electron, nucleus}, and a corresponding source ConceptSet is {planet, sun}.

Definition 2 (PrimaryRelationStructure): A *PrimaryRelationStructure* $\boldsymbol{P}$ = ($\boldsymbol{C_a}$, $\boldsymbol{R}$, $\boldsymbol{C_b}$) associates ConceptSets $\boldsymbol{C_a}$ and $\boldsymbol{C_b}$ through zero or more relations in the list of relations $\boldsymbol{R}$. We say "zero or more" because relations can be implied in an analogy rather than explicitly named.

In its most common form, a PrimaryRelationStructure consists of two concepts associated by a single relation as in ({planet}, revolves, {sun}). Note that there may be multiple relations between the ConceptSets (e.g., sun is *larger than* a planet, sun is *more massive* than a planet, sun is *hotter than* a planet, etc.), and that there may be more than one element in each concept set (the concept set containing the single element *planet* could be replaced by the set {*Mercury, Venus, Earth, Mars, Jupiter, Saturn, Uranus, Neptune, Pluto*}, for example.

Definition 3 (ConceptToRelationStructure): A *ConceptToRelationStructure* is a tuple of the form ($\boldsymbol{C}$, $\boldsymbol{R}$, $\boldsymbol{P}$) where $\boldsymbol{C}$ is a ConceptSet associated by zero or

more relations in the list ***R*** to the PrimaryRelationStructure ***P***.

Definition 4 (RelationToConceptStructure): A *RelationToConceptStructure* is a tuple of the form (***P***, ***R***, ***C***) where ***P*** is a PrimaryRelationStructure associated by zero or more relations in the list ***R*** to the ConceptSet ***C***.

A RelationToConceptStructure and a ConceptToRelationStructure associates relations with concepts or concepts with relations, respectively, depending on the directionality of the higher order relation being specified. Higher order relations can include causality, implication, and sequencing, for example. In the source analog of the Rutherford analogy, for example, we observe the ConceptToRelationStructure – gravitational attraction *causes* the planet to revolve around the sun.

Definition 5 (RelationToRelationStructure): A RelationToRelationStructure is a tuple of the form (***P***, ***R***, ***P***) that associates a PrimaryRelationStructure to another PrimaryRelationStructure through zero or more relations in list ***R***.

There are thus five types of maps, each of which maps a structure (e.g., PrimaryRelationStructure) in the target analog to a structure of the *same type* in the source analog: Figures 3.6 (a) – 3.6 (e) illustrate the five types of

maps discussed above.

Consider the analogy that compares the human circulatory system to home plumbing, used by physicians to explain cardiovascular diseases to medical patients [Ruhl, 1999]. The target concepts *heart, blood,* and *blood vessels* are mapped to the source concepts *pump, water,* and *pipes*, respectively, forming a ConceptSetMap. When explaining congestive heart failure to a patient, the physician first describes the similarity of the concepts of the circulatory system to those of a home plumbing system, implicitly using the ConceptSetMap, and then describes how the concepts are related, creating and explaining relational structures and maps. For example, the physician explains that a clogged artery can *cause* the blood to back up into the lungs similar to the way that a clogged drain can *cause* the water to back up and overflow, thereby mapping the causal relation of the source analog to that of the target. Thus, we see that this analogy, as it is presented by its author, may be decomposed into several ConceptSetMaps and a PrimaryRelationStructureMaps. Note that we can create multiple ConceptSetMaps, grouped by type of concept, as shown in Figure 3.7.

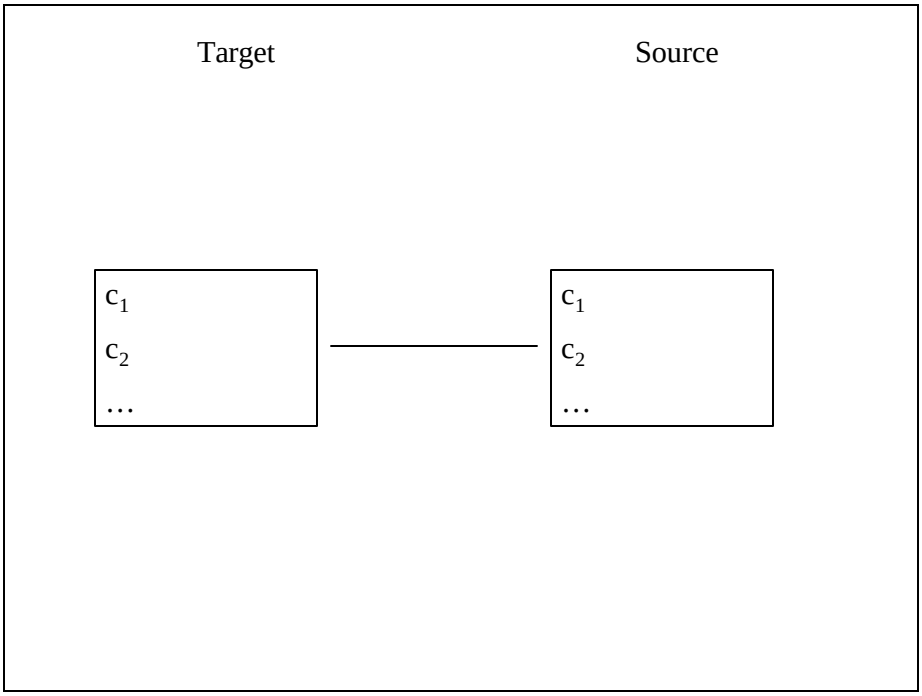


Figure 3.6 (a). ConceptSetMap

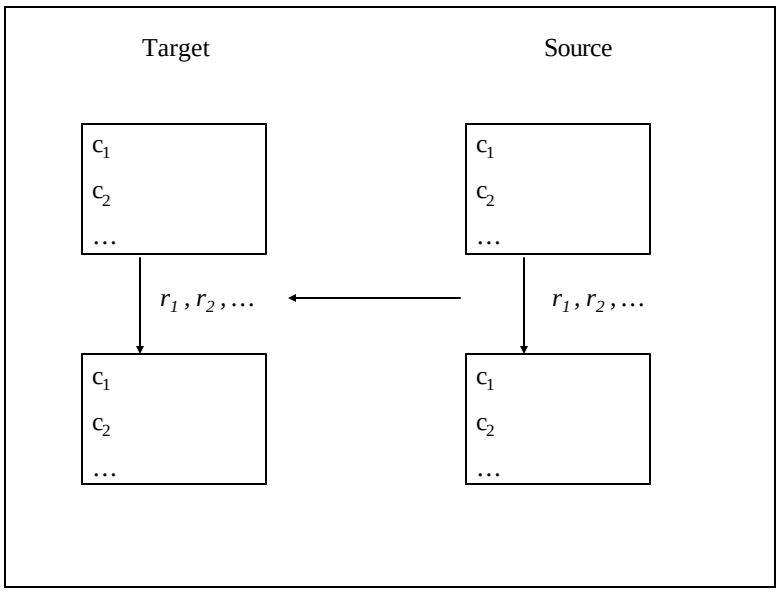


Figure 3.6 (b). PrimaryRelationStructureMap

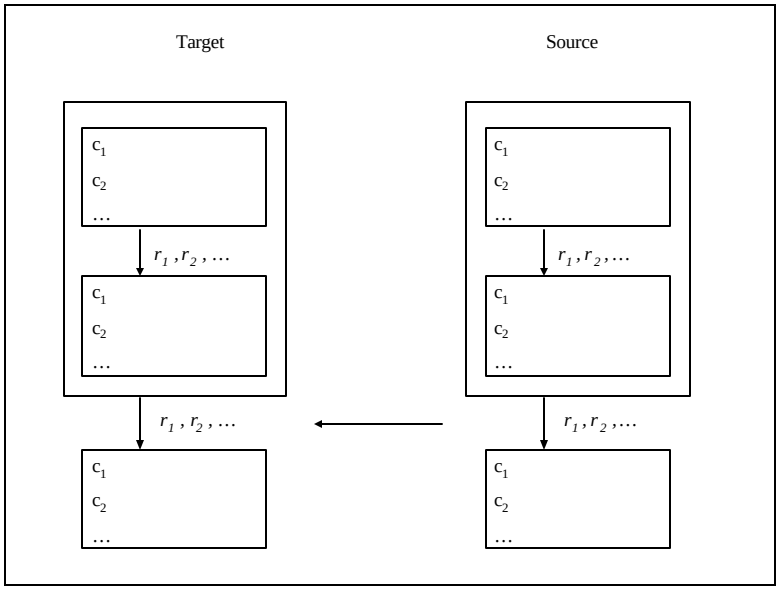


Figure 3.6 (c). RelationToConceptStructureMap

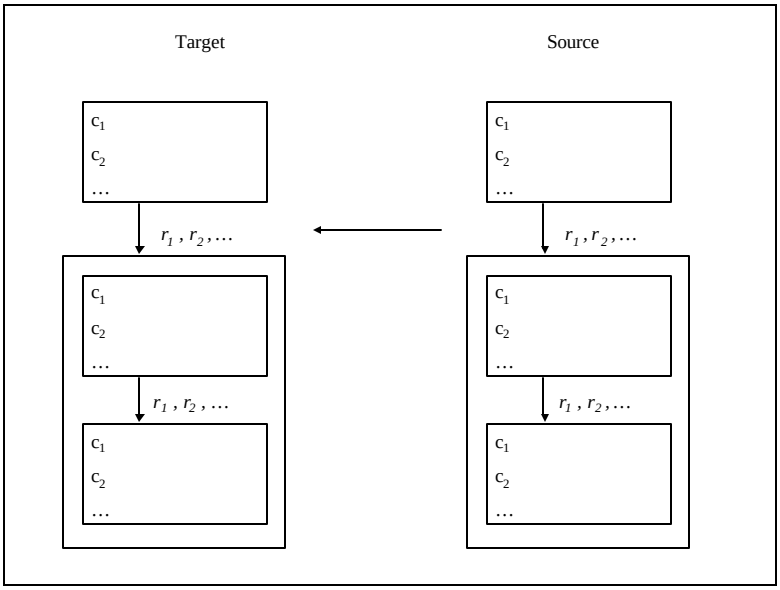


Figure 3.6 (d). ConceptToRelationStructureMap

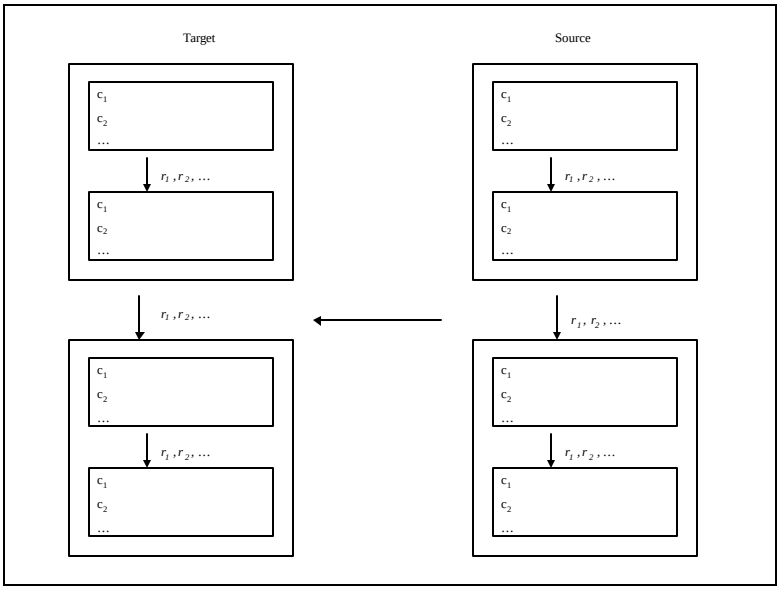


Figure 3.6 (e). RelationToRelationStructureMap

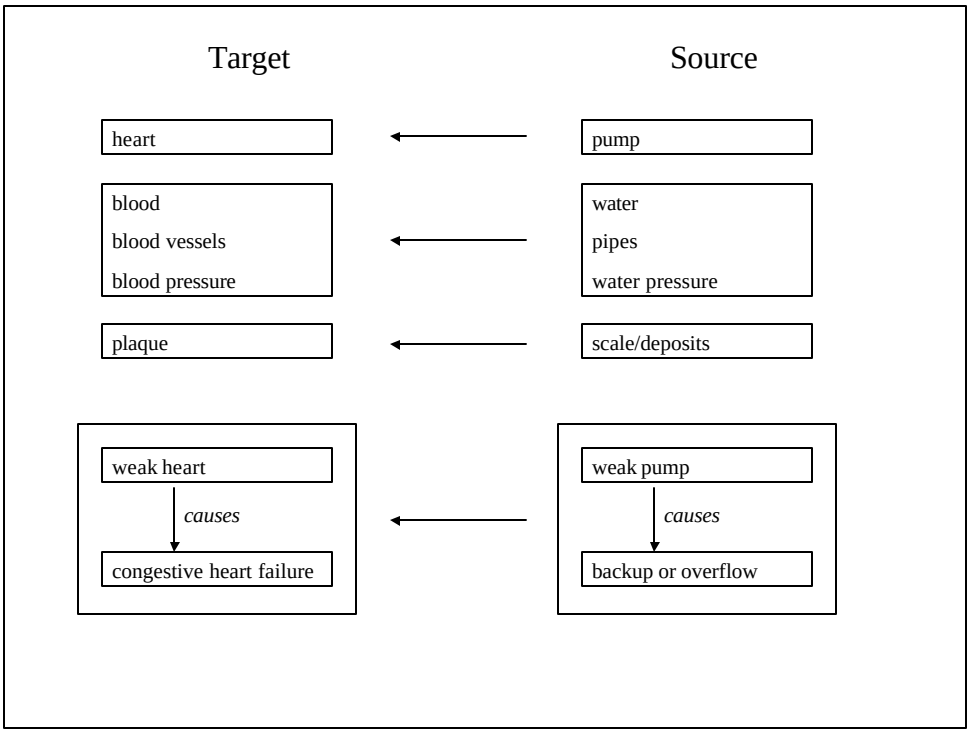


Figure 3.7. Circulatory System Analogy

This analogy and its MARVIN visualization may be found on the author's Web site [Foxwell, 2002], along with additional analogy examples taken from a variety of knowledge domains.

3.2 The Limitations of Analogies

Analogies are like chainsaws – powerful and useful tools that can injure you if you misuse them. Like incorrect maps, bad analogies can lead you away from your destination or cause you to become entirely lost. The over-reliance on limited analogical models can impair discovery and the formation of useful hypotheses. The early history of science includes many instances of misleading and inaccurate analogies. For example, alchemists were constrained in their understanding of matter and chemistry by their adherence to analogies between elements and animal or human characteristics [Gentner and Jeziorsky, 1993]. Kepler's reliance on a divinely ordained, geometrically perfect model of the universe led him initially to an erroneous explanation of the orbits of the planets [Kepler, 1596]. Only when he reluctantly abandoned that model was he able to conceive a more accurate explanation of the planets' orbits.

Practitioners of analogical explanations recognize the dangers of misleading analogies. A doctor who frequently uses analogies to explain medical

concepts notes that his patients can transfer inappropriate knowledge from the source to the target of an analogy, and suggests that good analogies should be visual, should illustrate the necessary concepts, use a familiar source, and should be clear and short [Ruhl, 1999]. Additionally, those explaining or teaching an idea should consider whether an analogy is even necessary. Analogies should be used primarily when the idea being taught is new and is hard for the learner to understand. The analogy's limitations should be discussed, and dependence on the analogy reduced as the learner progresses in understanding the target.

The use of historical analogies to guide decisions on the use of military force has been widely criticized by historians [Record, 2002]. The “lessons of history” can become obsolete or irrelevant as time progresses and as political and social conditions differ from the source event. The so-called “Munich analogy” – that appeasement of aggression by totalitarian states leads to more aggression – has been invoked as an instructive model for military and foreign policy decisions by several countries, although the model’s success and applicability has been questioned [Record, 1998].

Analogy practitioners do indeed recognize the limitations of analogies, and usually include a final recommended step in the analogical reasoning process

– to indicate where the analogy breaks down [Klein and Milligan, 2002], [Glynn, 1991], although they note that this should generally be done by the proposer of the analogy. This suggests that the process of *evaluating* the analogy is *separate* from the *depiction* of the analogy itself, as we discuss in Chapter 4.

Even those who suggest avoiding analogies can have difficulty explaining complex ideas without using analogy. For example, in criticizing the use of analogies in teaching computer concepts, [Halasz and Moran, 1982] end up simply using different analogies – replacing a "filing cabinet" model of file systems with a supposedly more general and less analogous "tree" model. Ironically, even the title of their paper, *Analogies Considered Harmful*, is purposely analogous to the title of a more famous paper, *GOTO Statement Considered Harmful*" [Dijkstra, 1968].

In spite of their dangers and misuses, analogies remain an important and widely used tool for communication and learning. Our research focuses on how to represent and visualize some of the enormous range of human-created analogies. The final evaluation of an analogy's usefulness, however, is the responsibility of its proposer and users.

4. The Representation of Analogies

As discussed in Chapter 3, an analogy may be described as a set of maps. Each map pairs the concepts and relational structures of a target analog to corresponding components of a source analog. The correspondence of concept and relation components is determined by the analogy author's *perception* of similarity of property, form, or function, or by a *hypothesis* of similarity.

Our goal here is to use this characterization of analogies to create an XML content model [W3C, 2000] capable of describing a wide variety of analogies. The design of such a model for analogies must meet several general criteria and must also meet the needs of content authors who select or create analogies, content readers who learn using the analogies, and program developers who use the model to create new ways to use the analogies.

A *representation* is a symbolic surrogate for an object that facilitates human expression about the object and serves as a medium for computations with that object [Sowa, 2000]. Note that we distinguish between an *analogy* – the human's perception of sameness, and an *analogy expression* – the

representation of that perception. The representation is necessarily a limited and imperfect model of the complete human perception, however. Our goal is to define a symbolic representation of analogies that captures a wide scope and variety of human-conceived analogies. Because the term “analogy” has a broad range of interpretations, our representation must reflect both the “common usage” of the term, as well as the more formal usage in research fields such as Cognitive Science, Artificial Intelligence, and Education. Specifically, it must be able to express both simple similarity of properties and concepts, and must preserve relational structures that are the core of analogy perceptions [Gentner, 1983]. Because analogies are remembered, reused, and extended over time to generate new inferences [Hofstadter, 2001][Keane and Costello, 2001], our representation must also be extendable. That is, it must be flexible enough to permit the addition of new analogy components as needed by the author, user, or developer.

Authors of instructional or explanatory content frequently use analogies to assist the student or reader in understanding new ideas. An analogy representation must therefore be compact yet expressive enough to allow the author to record the essential components of the analogy, and must also permit some indication of the validity of the proposed comparisons. In our analogy expression, we therefore use simple words and short phrases as

expression “primitives” along with expressions that describe the structure of the relations among the primitives. Because we use XML to define analogy expressions, authors can use XML editors or other tools to assist them in producing valid XML files. Chapter 7 describes an analogy expression editor created for this purpose.

We want to provide access to analogy expressions using familiar, Web-based tools and technologies such as browsers. Moreover, we need to separate the representation of the structure of the analogy expression from its visualization. XML was designed specifically for this purpose [W3C, 2002], allowing multiple forms of visualization based on a common representation model.

Researchers need programmatic access to knowledge expressions in order to store, retrieve, and manipulate them. The use of XML enables developers to access analogy expressions using standard, Web programming tools and methods such as XML parsers, XSLT stylesheets [W3C, 2001] and processors, Java programs, HTML, and SVG [W3C, 2001] to visualize these expressions in a variety of forms. Our representation is implemented as an XML DTD (Document Type Definition), permitting a compact, common format for describing, searching, and transforming analogy expressions.

Terminology used to describe the components and structure of analogies should be compact yet descriptive. In designing the DTD we stress the distinction between the target analog components and the source analog components by giving them separate but similar names. We also observe that in order to map only identical analogy structures, we must enumerate the possible structures rather than using more compact or recursive element definitions that would allow mapping of unlike components.

Knowledge representations defined using XML may be expressed using either DTDs or schemas. Like XML DTDs, XML Schemas are used to define the structure, content, and semantics of XML documents [W3C, 2001]. Schemas provide for strong data typing and validation, explicit cardinality controls, and constraints on attribute values [W3C, 2001]. But for representations that do not require strong data typing or cardinality controls, DTDs are sufficient and simpler [Mertz, 2001]. Tools for creating, validating, and transforming DTD-based XML files are mature and widely available [Cover, 2002] while the XML schema standard and tools are still evolving [Garshol, 2002]. Because the analogy expressions discussed in this document consist exclusively a relatively small number of text-based elements with *any number* of repeated components, a DTD was developed. XML analogy expressions based on our DTD may be transformed as needed into other

forms using XSL stylesheets or other tools, including transforming the DTD into an XML schema for further extension [Koike, 2000].

Figure 4.1 shows the definition of an analogy expression in modified Extended Backus-Naur Form (EBNF) [ISO, 1996]. We use this form to provide an initial formal definition of analogy expressions that includes the syntactic components of such expressions, the sequence and number of occurrences of these components, and the syntactic structure of the expression. We extend this definition in the next section to create an XML content model for analogy expressions.

As discussed in Chapter 3, we define an analogy expression as a set of maps that associate target concepts and relational structures with source concepts and relational structures. We include two general descriptive components, the Target Description and the Source Description, which may contain words or brief phrases about each major component of the analogy. We then define each of the five types of structures and maps that can occur in analogy expressions. Any type of map can appear any number of times in an expression. This permits the author of the expression to group concepts or to repeat components as needed to emphasize relations.

```
Analogy                          = TargetDescription SourceDescription Map*
Map                              = ConceptSetMap | PrimaryRelationStructureMap |
                                   ConceptToRelationStructureMap |
                                   RelationToConceptStructureMap |
                                   RelationToRelationStructureMap
ConceptSetMap                    = TargetConceptSet SourceConceptSet
PrimaryRelationStructureMap      = TargetPrimaryRelationStructure
                                   SourcePrimaryRelationStructure
ConceptToRelationStructureMap    = TargetConceptToRelationStructure
                                   SourceConceptToRelationStructure
RelationToConceptStructureMap    = TargetRelationToConceptStructure
                                   SourceRelationToConceptStructure
RelationToRelationStructureMap   = TargetRelationToRelationStructure
                                   SourceRelationToRelationStructure
TargetConceptSet                 = ConceptDescription+
TargetPrimaryRelationStructure   = ConceptSet Relation* ConceptSet
TargetConceptToRelationStructur  = ConceptSet Relation* PrimaryRelationStructure
TargetRelationToConceptStructure = PrimaryRelationStructure Relation* ConceptSet
TargetRelationToRelationStructure = PrimaryRelationStructure Relation*
PrimaryRelationStructure
SourceConceptSet                 = ConceptDescription+
SourcePrimaryRelationStructure   = ConceptSet Relation* ConceptSet
SourceConceptToRelationStructure = ConceptSet Relation* PrimaryRelationStructure
SourceRelationToConceptStructure = PrimaryRelationStructure Relation* ConceptSet
SourceRelationToRelationStructure = PrimaryRelationStructure Relation*
PrimaryRelationStructure
PrimaryRelationStructure         = ConceptSet Relation* ConceptSet
ConceptSet                       = ConceptDescription+
TargetDescription                = TEXT*
SourceDescription                = TEXT*
ConceptDescription               = Concept Property*
Concept                          = TEXT*
Property                         = TEXT*
Relation                         = TEXT*
TEXT                             = (any legal XML character)
```

Figure 4.1. Modified EBNF Definition of an Analogy Expression

A ConceptSetMap associates a TargetConceptSet with a SourceConceptSet.

A ConceptSet consists of any number of ConceptDescriptions, each of which

consists of a Concept and any number of associated Properties.

A PrimaryRelationStructureMap associates a TargetPrimaryRelationStructure with a SourcePrimaryRelationStructure; a PrimaryRelationStructure consists of two ConceptSets, associated by zero or more Relations. Analogies typically contain relations, but in some cases they may be implied rather than explicit.

ConceptToRelationStructures and RelationToConceptStructures associate a PrimaryRelationStructure to a ConceptSet using additional or higher order relations; the ConceptToRelationStructureMap and RelationToConceptStructureMap then pair these structures in the analogy expression.

The RelationToRelationStructure associates two PrimaryRelationStructures with zero of more higher order relations, and so the PrimaryRelationStructureMap pairs similar structures in the Target and in the Source.

XML DTDs can use elements or attributes to represent content. Components describing containment and repetition should be modeled as elements, the

remainder as attributes [Daconta, 2001]. The primary components of analogy expressions are the words used to describe concepts and relations, and these are represented by elements in the DTD. Words describing properties of concepts are also represented as elements because the words themselves may have attributes. The words selected are those explicitly used or implied in the narrative description of the analogy.

4.1 Analogy Expression DTD Elements

The Analogy Expression DTD defines the Analogy element as

```
<!ELEMENT Analogy (TargetDescription, SourceDescription, Map* )>
<!ATTLIST Analogy AnalogyName CDATA #IMPLIED
                  Reference CDATA #IMPLIED >
<!ELEMENT TargetDescription (#PCDATA )>
<!ATTLIST TargetDescription Domain CDATA #IMPLIED >
<!ELEMENT SourceDescription (#PCDATA )>
<!ATTLIST SourceDescription Domain CDATA #IMPLIED >
```

That is, an analogy expression consists of a pair of text descriptions, followed by zero or more map elements. The mapped elements are modeled separately from the descriptions because we do not want to imply that there are any mapped elements in the target or source text descriptions. The Analogy attributes include AnalogyName for indicating a general title or name for the analogy, and Reference for indicating the source of the analogy. The

Reference attribute may include a URL. The description elements are brief phrases describing the target and source analogs. By convention, target elements in the expression are always defined first. A Domain attribute is included for both the target and for the source to indicate the general knowledge domains being referenced in the analogy. All elements modeled as words or phrases use the #PCDATA (untagged character data) type. For example, the start of the Rutherford analogy expression can be tagged as follows:

```
<Analogy AnalogyName="Rutherford Analogy"
         Reference=" Bohr, Niels. Nobel Lecture, December 11, 1922,
                     http://www.nobel.se/physics/laureates/1922/bohr-lecture.html">
<TargetDescription Domain="Physics">
atomic structure</TargetDescription>
<SourceDescription Domain="Astronomy">
solar system</SourceDescription>
…
</Analogy>
```

The "primitive" elements of an analogy expression are concepts, properties, and relations, defined using #PCDATA as follows:

```
<!ELEMENT ConceptDescription (Concept, Property* )>
<!ELEMENT Concept (#PCDATA )>
<!ELEMENT Property (#PCDATA )>
<!ATTLIST Property Validity (high|low|none) #IMPLIED >
<!ELEMENT Relation (#PCDATA )>
```

We include an optional *validity* attribute for Properties to allow for expression of concept properties that may not be relevant to the analogy. For example, in the Rutherford analogy, *hot* may be specified as a property of the sun, but there is no corresponding relevant temperature property for the nucleus. Thus, if the temperature property for the sun concept is represented in the expression, the analogy expression author can label it with a validity indicator of “none”, while the mass property “heavy” can be labeled with a validity indicator of “high” since the relative masses of the sun/planet and nucleus/electron are essential to the analogy. This can be tagged as follows:

```
<ConceptDescription>
 <Concept>sun</Concept>
  <Property Validity=”high”>heavy</Property>
  <Property Validity=”none”>hot</Property>
</ConceptDescription>
```

Alternatively, irrelevant properties in the target and source analogs can simply be omitted from the expression.

Analogy expressions can contain any number of each of the five map types discussed earlier, defined in our DTD as follows:

```
<!ELEMENT Map (ConceptSetMap | PrimaryRelationStructureMap |
```

```
ConceptToRelationStructureMap | RelationToConceptStructureMap |
RelationToRelationStructureMap )>
```

The ConceptSetMap consists of a TargetConceptSet and a SourceConceptSet, each of which must contain at least one concept description, listed in the order that pairs like concepts of the target and the source. We include a MapLabel attribute for each type of map to allow the author to assign a generic name to the type of concepts or relations being compared in the analogy.

```
<!ELEMENT ConceptSetMap (TargetConceptSet, SourceConceptSet )>
<!ATTLIST ConceptSetMap MapLabel CDATA #IMPLIED >
<!ELEMENT TargetConceptSet (ConceptDescription+ )>
<!ELEMENT SourceConceptSet (ConceptDescription+ )>
```

The PrimaryRelationStructure, used in the definition of the remaining map structures, consists of two ConceptSets associated with zero or more Relations:

```
<!ELEMENT PrimaryRelationStructure
(ConceptSet, Relation*, ConceptSet )>
```

The remaining maps that model the structures discussed in Chapter 3 are defined as follows

```
<!ELEMENT PrimaryRelationStructureMap
          (TargetPrimaryRelationStructure,
           SourcePrimaryRelationStructure )>
<!ATTLIST PrimaryRelationStructureMap MapLabel CDATA #IMPLIED >
<!ELEMENT ConceptToRelationStructureMap
          (TargetConceptToRelationStructure,
           SourceConceptToRelationStructure )>
<!ATTLIST ConceptToRelationStructureMap MapLabel CDATA #IMPLIED >
<!ELEMENT RelationToConceptStructureMap
          (TargetRelationToConceptStructure,
           SourceRelationToConceptStructure )>
<!ATTLIST RelationToConceptStructureMap MapLabel CDATA #IMPLIED >
<!ELEMENT RelationToRelationStructureMap
          (TargetRelationToRelationStructure,
           SourceRelationToRelationStructure )>
<!ATTLIST RelationToRelationStructureMap MapLabel CDATA #IMPLIED >
```

Note that, like the ConceptSetMap, each of these maps also has a MapLabel attribute. Markups of additional analogy examples are shown in the Appendix, and at the author's Web site [Foxwell, 2002].

Figure 4.2 shows the complete analogy expression DTD.

```
<!-- AnalogyExpression.dtd, Copyright 2002 by Harry J. Foxwell and Daniel A. Menasce -->
<!-- http://cs.gmu.edu/~hfoxwell/MARVIN/Docs/AnalogyExpression.dtd -->
```

```
<!ELEMENT Analogy (TargetDescription, SourceDescription, Map* )>
<!ATTLIST Analogy AnalogyName CDATA #IMPLIED
                  Reference CDATA #IMPLIED >

<!ELEMENT Map (ConceptSetMap | PrimaryRelationStructureMap |
ConceptToRelationStructureMap |
               RelationToConceptStructureMap | RelationToRelationStructureMap )>

<!ELEMENT ConceptSetMap (TargetConceptSet, SourceConceptSet )>
<!ATTLIST ConceptSetMap MapLabel CDATA #IMPLIED >

<!ELEMENT PrimaryRelationStructureMap (TargetPrimaryRelationStructure,
                                       SourcePrimaryRelationStructure )>
<!ATTLIST PrimaryRelationStructureMap MapLabel CDATA #IMPLIED >

<!ELEMENT ConceptToRelationStructureMap (TargetConceptToRelationStructure,
                                         SourceConceptToRelationStructure )>
<!ATTLIST ConceptToRelationStructureMap MapLabel CDATA #IMPLIED >

<!ELEMENT RelationToConceptStructureMap (TargetRelationToConceptStructure,
                                         SourceRelationToConceptStructure )>
<!ATTLIST RelationToConceptStructureMap MapLabel CDATA #IMPLIED >

<!ELEMENT RelationToRelationStructureMap (TargetRelationToRelationStructure,
                                          SourceRelationToRelationStructure )>
<!ATTLIST RelationToRelationStructureMap MapLabel CDATA #IMPLIED >

<!ELEMENT TargetConceptSet (ConceptDescription+ )>
<!ELEMENT TargetPrimaryRelationStructure (ConceptSet, Relation*, ConceptSet )>
<!ELEMENT TargetConceptToRelationStructure (ConceptSet, Relation*,
PrimaryRelationStructure )>
<!ELEMENT TargetRelationToConceptStructure (PrimaryRelationStructure, Relation*,
ConceptSet )>
<!ELEMENT TargetRelationToRelationStructure
                           (PrimaryRelationStructure, Relation*,
PrimaryRelationStructure )>

<!ELEMENT SourceConceptSet (ConceptDescription+ )>
<!ELEMENT SourcePrimaryRelationStructure (ConceptSet, Relation*, ConceptSet )>
<!ELEMENT SourceConceptToRelationStructure (ConceptSet, Relation*,
PrimaryRelationStructure )>
<!ELEMENT SourceRelationToConceptStructure (PrimaryRelationStructure, Relation*,
ConceptSet )>
<!ELEMENT SourceRelationToRelationStructure
                           (PrimaryRelationStructure, Relation*,
PrimaryRelationStructure )>

<!ELEMENT PrimaryRelationStructure (ConceptSet, Relation*, ConceptSet )>
<!ELEMENT ConceptSet (ConceptDescription+ )>

<!ELEMENT TargetDescription (#PCDATA )>
<!ATTLIST TargetDescription Domain CDATA #IMPLIED >
<!ELEMENT SourceDescription (#PCDATA )>
<!ATTLIST SourceDescription Domain CDATA #IMPLIED >

<!ELEMENT ConceptDescription (Concept, Property* )>
<!ELEMENT Concept (#PCDATA )>
<!ELEMENT Property (#PCDATA )>
<!ATTLIST Property Validity (high|low|none) #IMPLIED >
<!ELEMENT Relation (#PCDATA )>
```

Figure 4.2. The Analogy Expression DTD.

An example markup for the Rutherford analogy is listed in Figure 4.3.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Analogy SYSTEM
"http://cs.gmu.edu/~hfoxwell/MARVIN/Docs/AnalogyExpression.dtd"[
]>
<Analogy AnalogyName="Rutherford Model of the Atom">
<TargetDescription Domain="Physics">Atomic
Structure</TargetDescription>
<SourceDescription Domain="Astronomy">Solar System</SourceDescription>

<Map>
<ConceptToRelationStructureMap>
<TargetConceptToRelationStructure>
 <ConceptSet>
   <ConceptDescription><Concept>electrical attraction</Concept>
   </ConceptDescription>
 </ConceptSet>
 <Relation>causes</Relation>
 <PrimaryRelationStructure>
   <ConceptSet>
     <ConceptDescription><Concept>electron</Concept>
       <Property>small</Property>
       <Property>light</Property>
     </ConceptDescription>
   </ConceptSet>
   <Relation>orbit</Relation>
   <ConceptSet>
     <ConceptDescription><Concept>nucleus</Concept>
       <Property>large</Property>
       <Property>heavy</Property>
     </ConceptDescription></ConceptSet>
 </PrimaryRelationStructure>
</TargetConceptToRelationStructure>
<SourceConceptToRelationStructure>
 <ConceptSet>
   <ConceptDescription><Concept>gravitational attraction</Concept>
   </ConceptDescription>
 </ConceptSet>
 <Relation>causes</Relation>
 <PrimaryRelationStructure>
   <ConceptSet>
     <ConceptDescription>
       <Concept>planet</Concept>
         <Property>small</Property>
         <Property>light</Property>
     </ConceptDescription>
   </ConceptSet>
 <Relation>revolve</Relation>
   <ConceptSet>
      <ConceptDescription>
        <Concept>sun</Concept>
          <Property>large</Property>
          <Property>heavy</Property>
      </ConceptDescription>
   </ConceptSet>
```

```
 </PrimaryRelationStructure>
</SourceConceptToRelationStructure>
</ConceptToRelationStructureMap>
</Map>
</Analogy>
```

Figure 4.3. A Rutherford analogy expression XML document

Note that there is no single "correct" model for expressing this or any other analogy. Depending on the needs of the author, the target and level of the audience, and the purpose of the analogy, the author may choose to include or exclude concepts, properties, and relations. The markup for a "minimal" expression for this analogy (*electron:nucleus::planet:sun* in the traditional ratio form), omitting optional elements and simply pairing the concepts and implying their relationship is shown in Figure 4.4.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Analogy SYSTEM
"http://cs.gmu.edu/~hfoxwell/MARVIN/Docs/AnalogyExpression.dtd"[
]>
<Analogy>
<TargetDescription>Atomic Structure</TargetDescription>
<SourceDescription>Solar System</SourceDescription>
<Map>
<ConceptSetMap>
<TargetConceptSet>
<ConceptDescription><Concept>electron</Concept></ConceptDescription>
<ConceptDescription><Concept>nucleus</Concept></ConceptDescription>
</TargetConceptSet>
<SourceConceptSet>
<ConceptDescription><Concept>planet</Concept></ConceptDescription>
<ConceptDescription><Concept>sun</Concept></ConceptDescription>
</SourceConceptSet></ConceptSetMap></Map>
```

Figure 4.4. A "minimal" Rutherford analogy expression XML document

4.2 Creating an Analogy Expression XML Document

A content author who proposes an analogy should first thoroughly

understand the target and source analogs, along with the limitations or possible misconceptions that the analogy may create, and then describe it in narrative or diagrammatic form. Using the Analogy Expression DTD next requires the identification of each of the key words that will be selected to describe the analogy's prominent concepts, relations, and properties that the author wishes to map.

For example, the narrative form of the Rutherford analogy [Bohr, 1922], begins:

> "According to our present conceptions, an atom of an element is built up of a nucleus that has a positive electrical charge, and is the seat of by far the greatest part of the atomic mass, together with a number of electrons, all having the same negative charge and mass, which move at distances from the nucleus that are very great compared to the dimensions of the nucleus or of the electrons themselves. In this picture we at once see a striking resemblance to a planetary system, such as we have in our own solar system. Just as the simplicity of the laws that govern the motions of the solar system is intimately connected with the circumstance that the dimensions of the moving bodies are small in relation to the orbits, so the corresponding relations in atomic structure provide us with an explanation of an essential feature of natural phenomena in so far as these depend on the properties of the elements."

The target structure to be learned or explained here is the atom, and the source structure, or example, is the solar system. Short phrases, "Atomic

Structure" and "Solar System", are then selected to serve as the TargetDescription and SourceDescription elements respectively. Next, although not all of them are explicit in the narrative text, the content author selects words to represent each of the concepts in the target and in the source that he wishes to place in correspondence. Thus, "nucleus" and "sun" can be paired concepts in a ConceptSetMap element. Similarly "electron" and "planet" can be paired. The key idea of the analogy is that electrons are hypothesized to orbit around the nucleus, similar in some manner to the revolution of the planets around the sun. Corresponding properties of the paired concepts may be specified, in this case the words "small" and "light" being selected for both electrons and planets, and "large" and "heavy" for both the nucleus and the sun.

Having identified all the key words for concepts, properties, and relations in the analogy, and having identified the proposed concept and relation maps, an XML file containing the maps can be created. A content author familiar with XML syntax can use a text editor for this purpose, but most authors will prefer an XML editor [XML-Spy, 2002], [Morphon, 2002], [epcEdit, 2002]. XML editors use the DTD file to define selectable element tags, simplify the markup process for the author, and ensure the creation of a valid XML file.

Figure 4.5 shows the creation of the Rutherford XML file using epcEdit. Like other XML editors, epcEdit displays the available tags for the author to select, and shows the tags used in the marked up document. Additional marked-up examples are shown in the Appendix and on the author's Web site [Foxwell, 2002]. These examples demonstrate that our analogy expression DTD has expressive power to represent a wide range of analogies.

Analogy practitioners are unlikely to be skilled in understanding and creating XML expressions. For this reason, we include with the MARVIN system an analogy expression editor [Ruan, 2002], described in Chapter 7, and available on the MARVIN website [Foxwell, 2002].

4.3 Summary

We formally defined a compact and general representation of the common components and relational structures of analogies, consisting of concepts, concept properties, relations, relational structures, and analogical structure maps. We then used this definition to design a general XML content model for analogy expressions for use in Web-based environments, and demonstrated through examples how it may be used to represent a wide range of analogies from diverse knowledge domains. This representation is a key component of the MARVIN system, supporting the visualization of analogy expressions in a variety of forms and the storage and retrieval of

analogy representations from online archives.

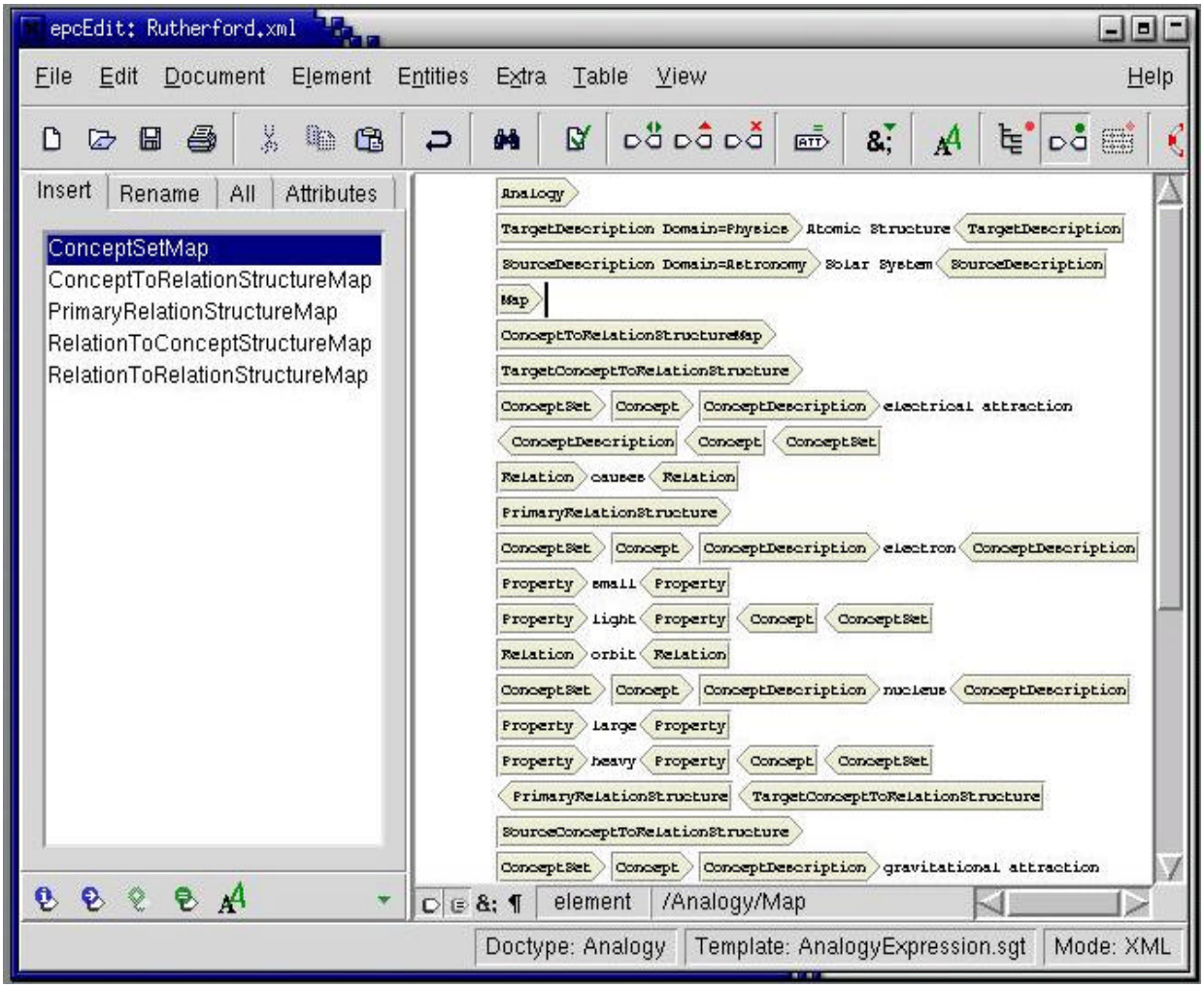


Figure 4.5: Creating a Rutherford XML file using the XML editor epcEdit

5. Visualizing Analogy Expressions

"A picture is worth a thousand words" is a common observation. Indeed, visualizations accompanying verbal or printed explanations have been shown to augment learners' internal visualizations and to enhance the development of internal visualization abilities [Hegarty, 2002]. Visualizations reduce the cognitive load on short-term memory [Tversky,et al., 2002], and can assist in the transfer of perceptions from the source analog to the target analog [Craig, et al., 2002].

An analogy can be understood when the person hearing or reading it knows something about the source analog and can observe and map the components of the source to the target analog. Understanding of the analogy is significantly improved when the relational structures present in the analogy can be visualized in a tabular or graphical form [Paris, 2000][Tversky, et al., 2002][Craig, et al., 2002].

The first use of our XML analogy expression is to create visualizations of the analogy proposed by the content author. An analogy visualization is any *spatially structured* representation of the components of the analogy, showing the concepts, properties, and relations of the target analog and how they map to those of the source analog. A general method is needed that is not specific to each individual analogy expression, but one that works with *all* such expressions that use the Analogy Expression DTD. For Web-based visualizations, we need to transform the XML expressions into HTML or compatible graphical form for display in standard Web browsers. For this purpose we use XSLT, the XML Stylesheet Language for Transformations [W3C, 1999]. This approach emphasizes the motivation behind using XML to express analogies – separating the representation of the analogy from the type of display we wish to create. Once the analogy is represented using the DTD, then tabular, graphical, and other types of displays may be generated from the same representation and displayed on any HTML-capable output device.

We create several XSLT stylesheets to demonstrate the variety of visualization aids that may be produced. One stylesheet constructs an HTML table representation of the components and structure of any analogy expression that uses our DTD. Figure 5.1 shows this type of visualization of

the Rutherford analogy. A modified version of this stylesheet produces HTML query strings linking the concepts in the visualization to ontology sites such as WordNet or WebKB. Other forms of visualization are possible; we developed another stylesheet that produces an SVG graphical representation of the analogy components. Figure 5.2 shows a graphical visualization of the Galileo example analogy expression displayed in the Mozilla browser. Additional examples and visualizations may be found in the Appendix and at the authors' Web site [Foxwell, 2002].

The tabular visualization displays the components of the analogy, placing the target analog components on the left and the corresponding source components on the right. Corresponding concepts, properties, and relations appear in the same row of the table, colors are used to highlight and organize matching components, and table cell borders of varying color and thickness are used to visually group matching components.

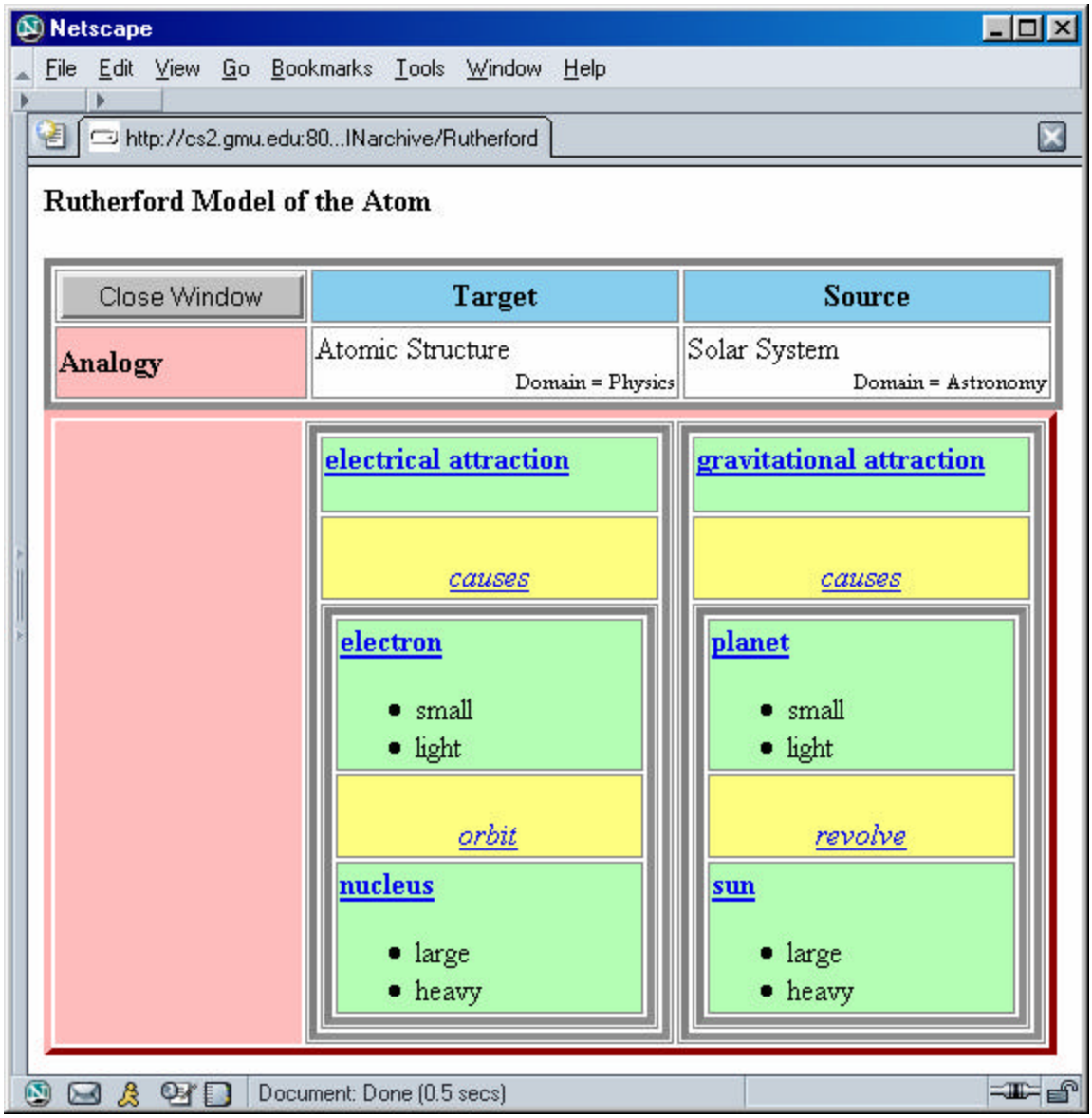


Figure 5.1 – Tabular visualization of Rutherford's analogy.

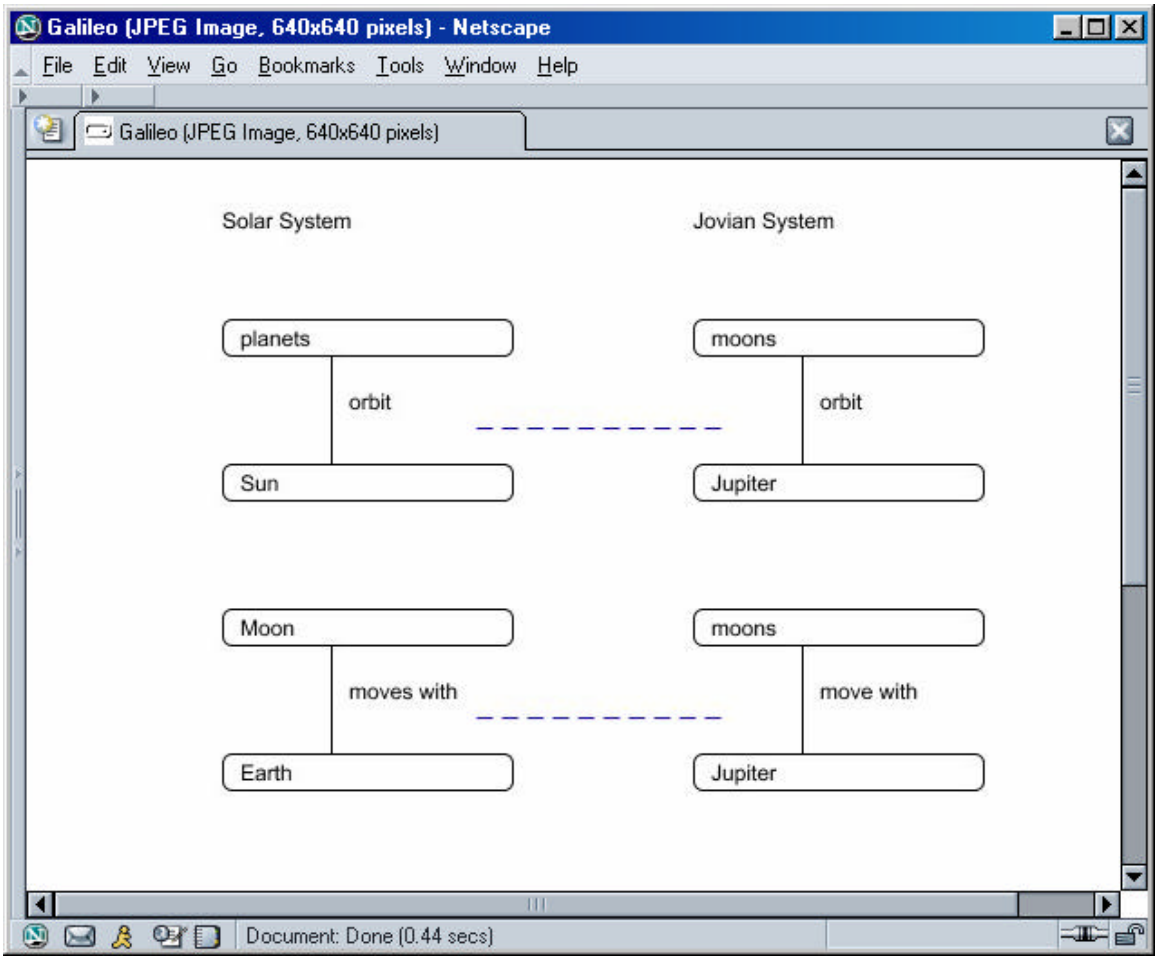


Figure 5.2 – Graphical visualization of the Galileo analogy.

The graphical visualization displays the target analog structures on the left of the image and the corresponding source analog structures on the right. Concepts appear in rectangles, and relations are indicated using labeled lines connecting the concepts. As discussed in Chapter 8, some users prefer the tabular visualization of analogies, while some prefer the graphical style; both authors and users have found such visualizations helpful in understanding

analogies presented in this manner.

5.1 Visualization Stylesheets

An Analogy Expression XSLT stylesheet consists of templates, each of which matches element nodes in parsed Analogy Expression XML documents. These templates specify the HTML or SVG code to be output for each node. The XSL processor traverses the XML document tree, activating each template's instructions when the specified *match* of a document component is found, and outputting the specified HTML or SVG code, analogous to the way messenger RNA binds to specific sites on a DNA molecule to generate new proteins [Piez, 2002][Foxwell, 2002], as was mentioned in section 2.1.7. Note that this is a good example of an analogy where the reader may not have background knowledge about the source (RNA), in which case the analogy, although clever, cannot be understood. A process whereby an alternate source analog could be obtained to help explain the target would be useful. Indeed, that is what is described in Chapter 6 and implemented in the MARVIN system.

The first template of the visualization stylesheet always matches the start, or *root* node of the XML document, outputs the <html> and <body> tags of the

new document, and then continues with the remaining templates. Thus, the *tabular* visualization stylesheet begins:

```
<xsl:template match="/">
  <html><head><B>
     <xsl:apply-templates select="Analogy/@AnalogyName"/>
  </B><br/>
     <xsl:apply-templates select="Analogy/@Reference"/>
  <br/></head>
  <body>
     <xsl:apply-templates/>
  </body>
  </html>
</xsl:template>
```

Each 'apply-templates' instruction selects a component of the XML document and applies the appropriate named template. In this example stylesheet, the template that matches the Analogy element outputs HTML tags to construct a table that contains the description and map components of the analogy expression:

```
<xsl:template match="Analogy">
<table border="4">
<tr>
<th width="128" style="font-size:70%">
<form>
<input type="button" value="Close Window"
onClick="window.close()"/></form>
</th>
<th bgcolor="#87CEEE" width="192">Target</th>
<th bgcolor="#87CEEE" width="192">Source</th>
</tr>
<tr>
<td width="128" bgcolor="#FFBCBC"><B>Analogy</B></td>
<td width="192"><xsl:apply-templates select="TargetDescription"/><br/>
      <div align="right" style="font-size:70%">
Domain = <xsl:apply-templates
select="TargetDescription/@Domain"/></div></td>
<td width="192"><xsl:apply-templates select="SourceDescription"/><br/>
```

```
      <div align="right" style="font-size:70%">
Domain = <xsl:apply-templates
select="SourceDescription/@Domain"/></div></td>
</tr>
</table>
<xsl:apply-templates select="Map"/>
</xsl:template>
```

A new subtable is started for each occurrence of any of the five different Map elements in the XML document. For example, we have the following templates:

```
<xsl:template match="Map">
<xsl:apply-templates select="ConceptSetMap"/>
<xsl:apply-templates select="PrimaryRelationStructureMap"/>
<xsl:apply-templates select="ConceptToRelationStructureMap"/>
<xsl:apply-templates select="RelationToConceptStructureMap"/>
<xsl:apply-templates select="RelationToRelationStructureMap"/>
</xsl:template>

<xsl:template match="ConceptSetMap">
<table border="4" bordercolor=”red”>
<tr><td width="128" bgcolor="#FFBCBC">ConceptSetMap</td>
<td width="192" bgcolor="#B4FFB4">
   <xsl:apply-templates select="TargetConceptSet"/></td>
<td width="192" bgcolor="#B4FFB4">
   <xsl:apply-templates select="SourceConceptSet"/></td>
</tr></table></xsl:template>
```

Similarly, each analogy expression component has a matching template that produces HTML tags and content. The final result is a complete HTML document stream generated from the XML expression that can be displayed

in the user's browser. The complete tabular visualization XSL stylesheet example is listed in the Appendix, and is also available at the author's Web site [Foxwell, 2002].

The tabular Analogy Expression XSLT stylesheet can be used for *any* analogy expression that uses the Analogy Expression DTD. This stylesheet produces an HTML document containing a table that displays the components and structure of the analogy expression, including concepts, properties, relations, and the five types of maps. The table organizes the expression into table cells, and groups the cells using colored borders of varying width to aid the reader in identifying analogy expression structures. Multiple maps are repeated in the display if they are defined in the analogy's XML document. Figure 5.3 shows a generic display of the analogy component visualization table and its contents as produced by the stylesheet and rendered by the Mozilla browser. Corresponding analogy components are displayed in the same row of the table. Concepts and relations can be displayed in contrasting table cell background colors for emphasis.

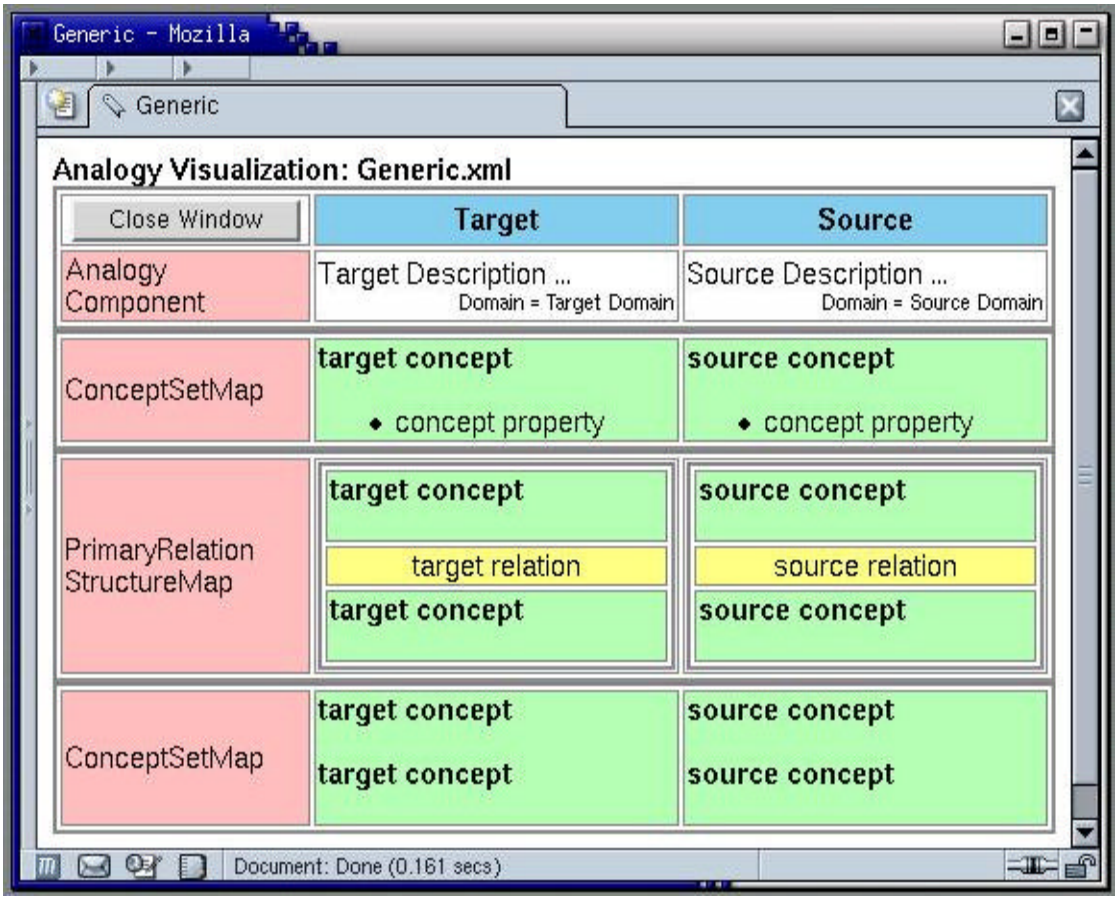


Figure 5.3 – Generic tabular visualization of an analogy.

For a graphical rather than tabular display of the analogy expression, a stylesheet was created to produce an SVG document that can be directly interpreted by a browser via an SVG plugin [Adobe, 2002]. Alternatively, the SVG output can be further transformed by the server into a graphic format such as GIF or JPG. The SVG visualization stylesheet is similar in structure to the tabular visualization stylesheet in that it consists of templates for each expression component that produce SVG graphics instructions instead of

HTML table tag instructions. Figure 5.2 shows a sample SVG visualization of the Galileo analogy using the graphical stylesheet.

Figure 5.4 describes the complete sequence for creating and visualizing analogy expressions. Using the Analogy Expression DTD file and an XML editor, a content author creates an XML file for the analogy. The XSLT processor takes the XML file and stylesheet as input data, processes it using the templates in the XSLT file, and outputs an HTML document that can be viewed with a browser. Xalan [Apache, 2001], a separate XSLT processor, can be used to create HTML or SVG output documents from the stylesheets. The MARVIN system, discussed in detail in Chapter 7, uses the XSLT processor embedded in Cocoon, along with our stylesheets, to transform XML analogy expressions into visual representations.

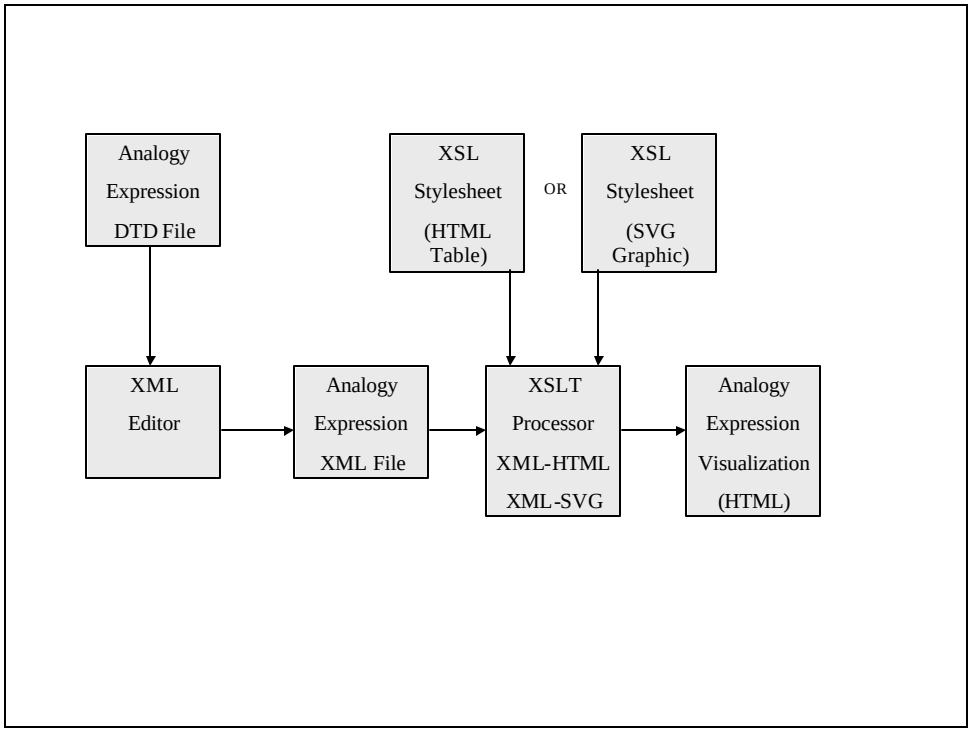


Figure 5.4. Process for creating and visualizing analogy expressions

The HTML document produced by the XSLT processor can, of course, be edited or transformed to enhance the document with links to other Web documents or images. A key step to understanding an analogy is to understand the *meaning* of the terms used to describe it. A small modification to the XSLT stylesheet can be made to create links for each concept to an explanation or definition of that concept. Such definitions can be created explicitly, or can be extracted by querying appropriate Web resources such as Google [Google, 2002], WordNet [WordNet, 2002], or WebKB [Martin, 2002]. For example, the template that matches Concept

nodes can be rewritten to append the Concept text to the character string that evaluates to a WebKB query:

```
http://www.webkb.org/bin/termDisplay.cgi?term=
```

The following Concept template accomplishes this:

```
<xsl:template match="Concept">
<td width="128">
<A TARGET="new">
<xsl:attribute name="HREF">
<xsl:text>http://www.webkb.org/bin/termDisplay.cgi?term=</xsl:text>
<xsl:apply-templates/>
</xsl:attribute>
<xsl:apply-templates/>
</A></td></xsl:template>
```

The resulting HTML document will contain concept terms that are links to WebKB. Clicking on any of the concept terms sends that term as a query to WebKB and returns a definition for that term in a new browser window. Figure 5.5 shows the visualization for the Rutherford analogy with concepts expressed as links to WebKB, and the resulting query result for the *electron* concept.

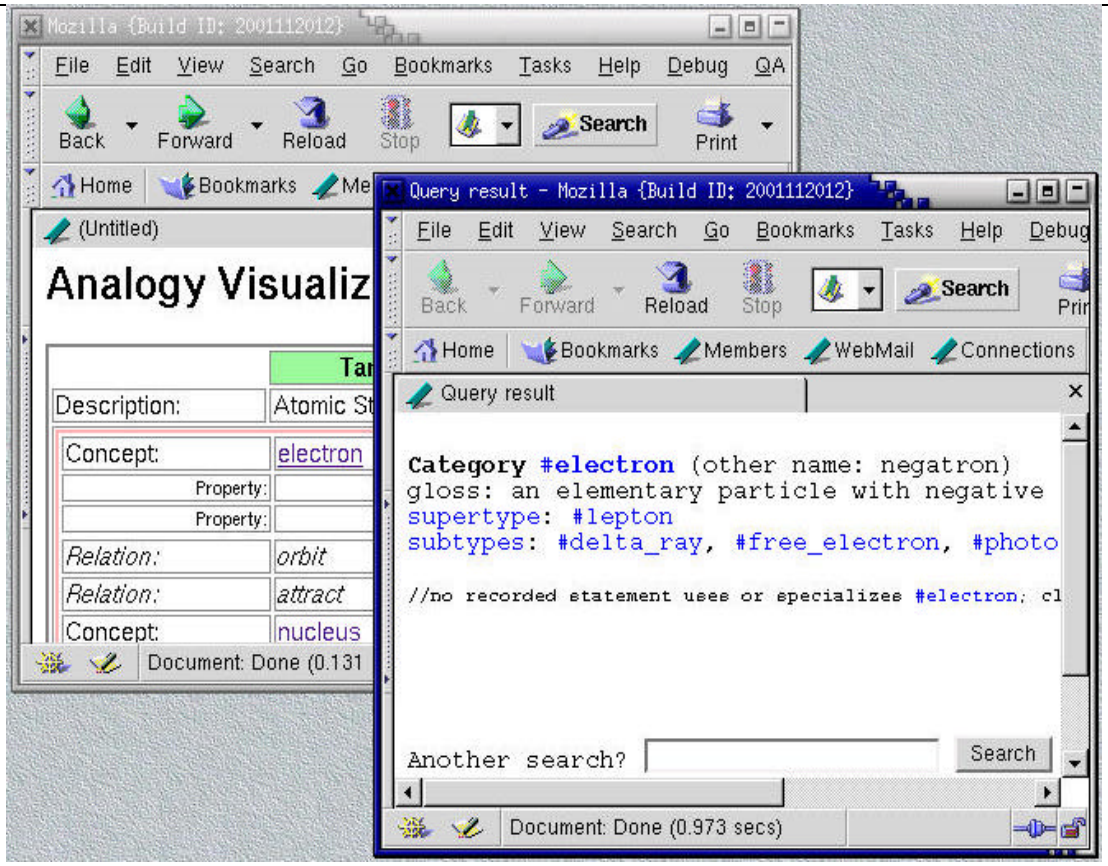


Figure 5.5: Rutherford Analogy visualization using concept links to WebKB.

5.2 Summary

We developed, using XSLT stylesheets, a general method for transforming XML analogy expressions into HTML or SVG for visualization in Web browsers. We presented several example visualizations, and showed how the stylesheets can be enhanced to link analogy expression components to Web-based tools such as WebKB. The stylesheets are integrated into the MARVIN system described in Chapter 7 to provide Web-based visualizations of our analogy expressions.

6. Storing, Retrieving, and Ranking Analogy Expressions

Having developed general methods for representing and visualizing analogies, we can now create archives of locally stored and Web-based XML analogy expressions. One motivation for creating such archives is to provide a mechanism for sharing creative and useful analogies among those who use them in their work. Like a traveler who uses a road atlas to find a map of his destination, an analogy user might want to look up analogies in an "analogy atlas." For example, a biology teacher may want to look up and evaluate analogies about the heart and circulatory system. While this is certainly possible now with search engines like Google [Google, 2002] by adding "analogy" to the other search terms, a system such as MARVIN, described in the next chapter, specifically stores analogy expressions in a standard XML format and provides a variety of mechanisms for retrieving and visualizing them, and for evaluating their pertinence to the topic under discussion.

An additional motivation for such retrievals concerns the *understanding* of

proposed analogies. If we present to a student that "the eye is like a camera," the analogy will be understood *only* if the student has background knowledge about cameras and how they work. When the student lacks such knowledge, the analogy is utterly meaningless. If, however, the student can extract from an archive *several* analogies that have the *same target* as the original analogy, he may find one or more that he understands. The presentation of multiple analogies for the same target has been shown to expand the range of hypotheses that students consider [Catrambone and Holyoak, 1989].

Although analogy creation, evaluation, and analogical reasoning are key components of scientific investigation, researchers typically forget their reasoning processes and the analogies they used, and have difficulty recreating them or explaining them to others [Dunbar, 2001]. A system for representing and recording the analogies considered during research can assist scientists in documenting and remembering their thought processes.

We have characterized our collection of analogy expressions as an "archive," much like a database. Retrieving an expression from the archive in one sense is like a database *query* – any retrieval process is expected to return only "records" (analogy expressions) that match the request criteria in some

manner. In database queries, there is generally no notion of *approximate* record match, either the record matches the criteria or it does not [Ramakrishnan and Gehrke, 2003]. In *searches* of text documents, on the other hand, there is a notion of approximate matching and a need to order the results by some relevance metric.

We cannot expect all analogy expressions dealing with the same general target subject to have the same structure or even to use the same words to describe the concepts and relations. We therefore require some characterization of analogy expressions that allows us to rank one expression as "better" than another in some manner. Unlike typical text document searches, however, we are not interested in word frequency within the expression. Rather, we are primarily interested in *where* the retrieval keywords occur *within* the analogy expression structure because the type of map in which a concept word or relation word occurs conveys additional information about the role of that word in the analogy.

We now consider the problem of retrieving analogy expressions from an archive and ordering the results according to some meaningful metric. Recall that the primary components of an analogy are concepts and relations, and

that these are explicitly represented as XML elements in our analogy expressions. Note also that the main focus of interest in an analogy is the *target* analog and its component concepts and relations. We therefore require a procedure to query an archive of analogy expressions, specifically their XML Concept and Relation element contents, for expressions whose target analogs contain some form of the query keywords of interest.

It is important to emphasize at this point that for the purposes of retrieval we are not necessarily interested in the overall *structure* of an analogy expression, only that it *contains* concepts or relations in the target analog elements that are relevant in some way to the query keywords. Thus, when we query the analogy archive for alternates to a proposed analogy, we are not seeking analogical similarity in a structural sense because the alternates may have different structures, different levels of detail or complexity, or different ordering of the map elements.

6.1 Retrieval Queries

We now examine various retrieval strategies for querying an archive of analogy expressions. An archive may include XML files stored in a local disk directory, and URLs for Web-based XML files. In either case, the files must

conform to our Analogy Expression DTD to permit querying based on their element contents. We examine two types of queries – keyword match queries, and generalization queries.

6.1.1 Keyword Match Queries

Recall that our analogy expressions are collections of *simple* and *compound* XML elements. *Simple* XML elements contain only parsed character data (`#PCDATA`); *compound* XML elements contain other elements. To illustrate, the Map element in our DTD is a compound element consisting of a ConceptSetMap or other type of map, while the Relation element is a simple element consisting of `PCDATA`. For our queries, we are primarily interested in the contents of the *simple* elements defined in the DTD, along with their optional attributes. These elements contain words representing concepts and relations that are structured using map elements.

The *simple* elements of interest for queries in an Analogy Expression DTD are elements of the analogy expression that are of one of the following types:

```
TargetDescription
SourceDescription
Concept
```

`Property`

`Relation`

Attributes of interest for queries include:

`AnalogyName` (of the `Analogy` element)

`Reference` (of the `Analogy` element)

`Domain` (of the `TargetDescription` element)

`Domain` (of the `SourceDescription` element)

`Validity` (of the `Property` element)

Let A be a set (an archive) of n analogy expressions:

$A = \{a_1, a_2, \ldots, a_n\}$

where each a_i is an analogy expression. Let T be the set of possible simple element types in a_i :

T = {Concept, Property, Relation, TargetDescription, SourceDescription},

$S(a_i) = \{ e_{i1}, e_{i2}, e_{i3}, \ldots \}$ is the set of simple elements of a_i, and

t (e) be the element type of simple element e, where $t \in T$.

Let w (e) represent the contents (word or phrase) of simple element e.

Then for a query on archive A for a keyword *q* contained in simple element type *s*, we want the result set of all a_j in A containing element type whose

contents are equivalent to the query keyword q:

$$R(A,s,q) = \{a_j \mid (a_j \in A) \wedge (\exists e \in S(a_j) \mid t(e) = s \wedge q \approx w(e))\}$$

The notation $q \approx w(e)$ means that query keyword q "matches" the contents of element e in some sense. Such a match can include an exact character string match, case-insensitive matching, and wildcard (regular expression) matching. That is, a query for the keyword expression 'atom*' should return matches for 'atom', 'Atom', and 'atomic', for example. Most text search tools permit this type of matching; the search engine used in the MARVIN system allows regular expression and case-insensitive searches.

To illustrate, a user might wish to find all analogy expressions in an archive that contain the concept *nucleus*, or the relation *orbits*, or to find all expressions in the archive whose target description domain attribute is *physics*. Examples of such queries are discussed in section 6.3.

6.1.2 Keyword Generalization Queries

A keyword match query will only return analogy expressions in the archive that contain the *identical* words (or regular expression approximations of the words) in the specified query elements. We may also be interested, however, in retrieving expressions that contain synonyms or other generalizations of

the query keywords, because an analogy expression author may have used synonyms or more general terms to describe the analogy.

Let $g_k(w(e))$ be a set of k-level generalizations of the content word of element e. By *k-level* we mean the number of parent generalization steps required to obtain the set of generalizations of the query keyword. Such generalizations can include *synonyms* (words having the same or similar meaning, which are therefore *0-level* generalizations), or *hypernyms* (words that are more generic than the given query keyword word and include it as an instance or example). Synonyms, hypernyms, and other *k-level* generalizations can be obtained manually or from online vocabulary ontologies such as WordNet [Miller, 2001]. For example, WordNet gives "serpent" as a synonym (0-level generalization) for the concept query keyword "snake", "reptile" as a hypernym (1-level generalization), "vertebrate" (2-level generalization), and "animal" (3-level generalization).

Then, for a k-level keyword generalization query of archive A for a query keyword *q* contained in simple element type *s*, we want the result set of all a_j in A containing elements of type *s* whose set of k-level generalizations of their contents are equivalent to the query keyword *q*:

$$R_k(A,s,q) = \{a_j \mid (a_j \in A) \wedge (\exists e \in S(a_j) \mid t(e) = s \wedge (g_k(q) \cap g_k(w(e)) \neq \Phi)) \vee (q \approx r \mid r \in g_0(w(e)))\}$$

For example, we may be interested in analogy expressions containing at least one Concept element whose contents match the keyword *nucleus* or generalizations of *nucleus* such as *center* or *core*. Some search engines, such as SHOE (Simple HTML Ontology Extensions) can be configured to perform such queries [Heflin, 1999]. The Lucene search tool used in the MARVIN system does not currently have this capability, but it is being investigated by the Lucene developers [Apache Software Foundation, 2002].

Archive retrieval capability must also, of course, include the capability to formulate complex queries, such as Boolean combinations of simple queries. For example, we may be interested in analogy expressions with a Concept element containing the keyword *nucleus* AND a Relation element containing the keyword *orbit*. Again, most search engines, including the Lucene implementation used in the MARVIN system, permit such combinations.

6.2 Ordering Analogy Expression Query Result Sets

Query result sets of analogy expressions should be ordered according to their

relevance to the query keyword expression and according to their pertinence to the user's goal for the query. Relevance is an objective measure of closeness of a query result to the query; pertinence is a subjective assessment by the user of the usefulness of a query result [Salton and McGill, 1983].

When searching for analogy expressions that are related in some sense to a query containing concept or relation keywords of interest, we are primarily interested in *topic relevance* rather than any structural similarity that any two expressions may have. And, while we are interested in the pertinence of our analogy expressions, the notion of relevance as it applies to analogy expression queries needs definition beyond the usual keyword frequency measures used in document searches.

Note that a target concept can occur within various map elements in an analogy expression. When it occurs within a ConceptSetMap element, there is little additional information provided about the concept beyond the source concept to which it is mapped. When it occurs within a PrimaryRelationStructureMap, however, we have more information about the role of that concept within the analogy. Target concepts occurring within ConceptToRelationStructureMaps or RelationToConceptStructureMaps

provide still more information about the concept's role. Generally, such maps indicate higher order relations within the analogy, and therefore provide critical information about the basic relational structures being mapped between the target and source. We can therefore propose a measure based on the level of information provided by the type of map in which a concept occurs as a metric for ordering query result sets.

To summarize, we will order expressions according to the type of map element in which the concept query keyword occurs, giving priority to concepts occurring within RelationToConceptStrucureMap and ConceptToRelationStructureMap elements, then to those occurring within PrimaryRelationStructureMap elements, and then those occurring only within ConceptSetMap elements.

To accomplish this ordering of expressions, assign a rank metric r to an expression a_i as follows:

For a given query keyword q and expression a_i, let $Q = \{q_1, q_2, \ldots\}$ be a set of query keywords. Let the rank value for the expression a_i containing those keywords be

$$r(Q, a_i) = \sum_{q \in Q} \sum_{j=1}^{m_i} f_j(q) \times w_j$$

where m_i is the number of maps in expression a_i, $f_j(q)$ is the number of occurrences of q in map j of expression a_i, and w_j is the weight factor assigned to map j.

The result set of expressions is sorted according to this rank metric, highest to lowest. Thus, expressions providing more information about the role of the relations associated with the concept in the analogy will rank higher in the sorted list, but the user must still inspect the analogy and evaluate its pertinence for the user's purposes.

The weights in the above formula reflect the number of *relations* for a given concept implied within each type of analogy expression map element. That is, for a concept that occurs only in a ConceptSetMap element, there is only one association described – the map between the target concept and the source concept. For a concept that occurs in a PrimaryRelationStructureMap, there are two relations implied. Thus, we assign the weight factors discussed above as follows:

$w_j = 1$ if the map type is ConceptSetMap

$w_j = 2$ if the map type is PrimaryRelationStructure

$w_j = 4$ if the map typeis ConceptToRelationStructureMap

$w_j = 4$ if the map type is RelationToConceptStructureMap

$w_j = 6$ if the map type is RelationToRelationStructureMap

$w_j = 0$ otherwise.

We might consider other indicators of relevance or information content for an analogy expression, but the order, structure, and detail various authors include in analogy expressions may vary significantly even for the same basic analogy. For example, how would we order two analogy expressions with identical target and source analogs, but differing level of concept details such as properties or property validity? Greater detail does not necessarily imply greater information about the role of the concept in the analogy.

The system user may want to specify that some expression characteristic is pertinent to the query purpose. For example, the user may want to select expressions having a specific Domain attribute for the target or source descriptions, such as “Biology” or “Computer Science.” The system must therefore permit ordering according to such user pertinence preferences, and

then order the result set according to relevance measures within the pertinence categories.

6.3 Query Examples

For our MARVIN archive queries we use the Lucene text search engine, which permits construction of queries, based on the content of specific elements in the XML expressions. A simple keyword match query, Concept:nucleus, for example, would return all expressions in the archive whose Concept elements contain at least one instance of the *nucleus* keyword. A generalization search would return expressions whose Concept elements contain a generalization of *nucleus*, such as *nucleon* or *center*. Of course, we are interested in relations as well in the analogy expressions, so we can construct a compound query, for example

`Concept:nucleus AND Relation:cause`, which would return expressions whose Concept elements contain the keyword *nucleus* and whose Relation element contains the keyword *cause*. The query syntax defined by Lucene also permits query of element attributes. The query `SourceDescription@Domain:Biology`, for example, returns all analogies in the archive whose Domain attribute for the SourceDescription element contains the keyword *Biology*. Lucene also permits the importance of a term

in a search to be increased using a “boost factor”, which increases the default relevance of the search term when the result set is sorted.

6.4 Summary

We have examined the process of storing, retrieving, and ranking analogy expressions. The primary motivation for this is to allow the sharing of analogy expressions in Web-accessible archives and to enable the retrieval of alternate analogy expressions. We described exact match and generalized match queries, recognizing that analogy expression authors may use synonyms or more general terms to describe their analogies. The retrieved analogy expressions must be ranked by their pertinence to the user's purpose and by the amount of information they provide about the structure of the analogy; we developed such a ranking method, and describe its implementation in Chapter 7.

7. MARVIN: A Prototype System for Representing, Retrieving, and Visualizing Analogy Expressions

MARVIN (Markup for Analogy Representation and Visualization for the InterNet) is a prototype system we designed to support representation and visualization of *human-conceived* analogies in a common format accessible to any Web browser. It is designed to help analogy practitioners, such as authors and teachers who use analogies in instructional Web content, to represent and display the *concepts* and *relations* contained in an analogy. Additionally, it helps end users of the system to display *tabular*, *graphical*, and other forms of visualization of the analogy, enhancing their understanding of the target analog. MARVIN also enables users to *query* an analogy *archive* for analogy suggestions and to obtain *alternatives* to the analogy proposed by the content author.

7.1 Design Goals

The MARVIN system's overall design goal is to enable representing and sharing human-conceived instructional analogies using standard, open-source Web technologies. Specifically, we want to

- represent analogy structure and content using a compact, common knowledge structure that is separate from display markup and syntax
- visualize these analogy representations in a variety of useful forms
- store and retrieve the representation in order to share them with others.

Implementing MARVIN using standard and emerging Web technologies ensures global accessibility through any HTML-capable browser, and allows developers who wish to use MARVIN analogy expressions to use widely available programming tools to produce new types of visualizations. The use of XML to describe analogies allows authors to represent the essential components of the analogies independent of their visualization. This enables programmers to create many forms of visualization from the same representation, using XML transformation technologies such as XSLT stylesheets.

Although MARVIN does not require integration with instructional Web systems, it can be used as a companion to such systems. Authors of any Web content that uses analogies can create and link to a MARVIN analogy

expression to augment the explanations of the analogies. Referencing the link in this way allows the user to read about the analogy in the author's primary Web page, and then follow the link to the MARVIN expression to obtain visualizations of the analogy or look up alternate analogies for the same target. Figure 7.1 shows this process – the user follows the analogy link in the main content page, and the MARVIN system references the analogy expression and the MARVIN DTD along with the appropriate visualization stylesheet. The user can also query the analogy expression archive to obtain additional analogies referring to the presented target analog.

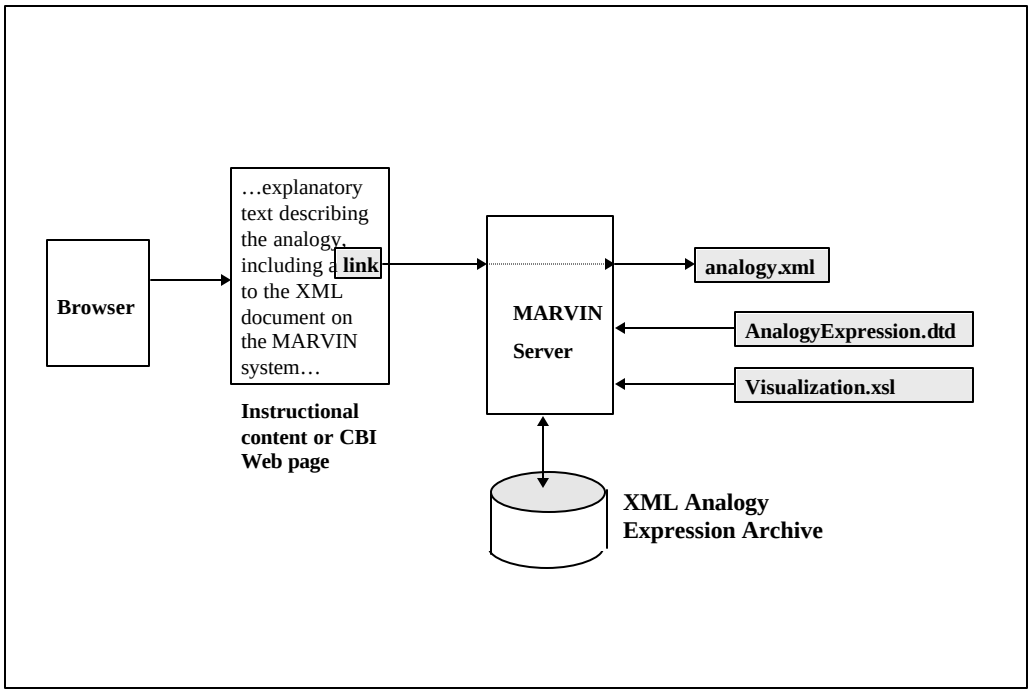


Figure 7.1. Integrating MARVIN into an Instructional or CBI Web Page

Content authors should first identify an analogy they wish to represent using the MARVIN system, and then proceed as follows:

- identify the analogy's concepts, properties, and relational structures
- create a MARVIN analogy expression using an XML editor or the MARVIN Analogy Editor
- upload the analogy expression to the MARVIN analogy archive
- create a link to the analogy expression on their main content Web page

MARVIN users then access the expression through the MARVIN web site, which provides various visualizations of the analogy and the ability to look up alternatives to the original analogy.

7.2 System Architecture

The MARVIN system is designed to use a programmable proxy server that can process Web-based Analogy Expression documents as well as those stored in the MARVIN archive. The proxy server runs an XML transformation servlet that processes the XML Analogy Expression, using an XSL stylesheet; when the user selects a specific type of visualization, the appropriate stylesheet is called. Programmers can add new stylesheets to the system to produce other visualizations and transformations. The stylesheet transforms the XML document into HTML for the tabular visualization, or into SVG for the graphical visualization. An additional transformer is used to convert the SVG graphic into JPEG format for inclusion as an image in the generated HTML page. Figure 7.2 shows the general architecture of the MARVIN system.

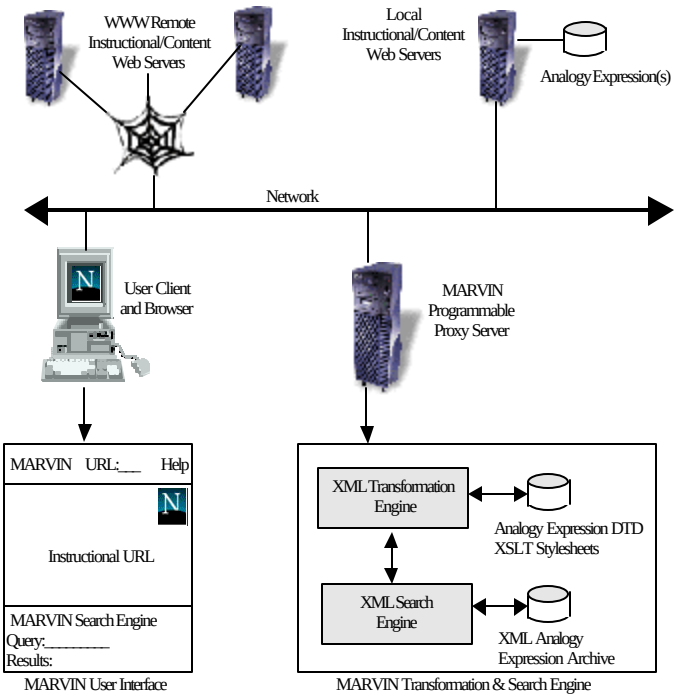


Figure 7.2. The MARVIN System Architecture

Note that although modern browsers such as Microsoft Internet Explorer 6 and Netscape 7 are starting to include support for XSL and SVG transformations, these efforts are still incomplete. We therefore generate standard HTML and JPG output from the MARVIN stylesheets on the server, which requires no browser processing of the analogy expressions. This server-based approach reduces the demands on the client browser, and permits generation of other output formats, such as WML [W3C, 2002] for display on portable wireless devices such as PDAs and Web-capable cellular phones.

The MARVIN system server is implemented using the Apache Tomcat servlet container [Apache, 2002]. An XML transformation engine, Cocoon [Apache, 2002], runs as a servlet within the container and manages the selection and processing of XML documents and XSL stylesheets. When a user requests an Analogy Expression XML document, the Tomcat servlet container passes the request to the Cocoon servlet which retrieves the XML document from the archive, retrieves the appropriate stylesheet for the type of transformation selected, and transforms the XML document contents into HTML or SVG using its XSLT processor (Xalan or Saxon). The resulting HTML is returned to the user's browser through the servlet container. Figure 7.3 shows the sequence diagram for this process.

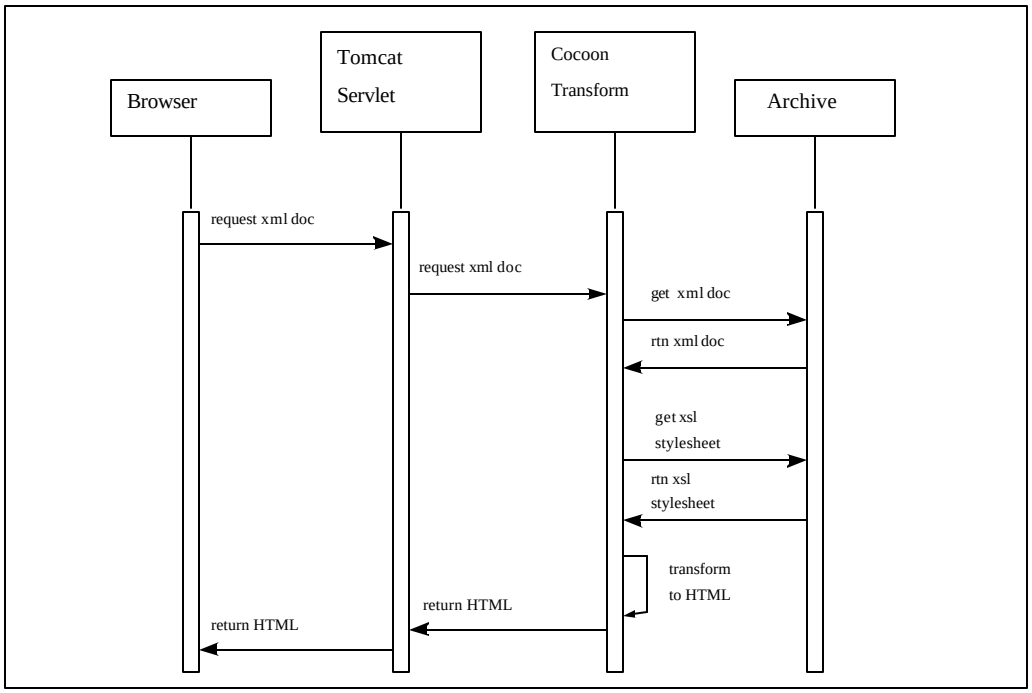


Figure 7.3. Sequence diagram: analogy expression transformation, XML to HTML

This implementation achieves the primary design goals as follows:

- *representation*: the use of a general XML content model provides a means of describing the structure and components of an analogy independent of any presentation syntax
- *visualization*: enables developers to produce a variety of visualization types by adding other XSL stylesheets to the system that can be processed by Cocoon
- *storage/retrieval:* enables the sharing of interesting and useful analogies with other practitioners and users by providing tools to author expressions and to add them to the MARVIN archive.

It is possible to use MARVIN analogy expressions directly without going through the MARVIN server if the browser is capable of XSL stylesheet retrieval and interpretation and supports the display of SVG graphics. As noted earlier, however, current browsers are incomplete or buggy in their support and implementation of such features, although this is expected to improve [Mozilla, 2003]. Figure 7.4 illustrates direct access of an analogy expression along with the retrieval of the necessary DTD and stylesheet, which can be explicitly referenced within the XML analogy expression document. Also note that such direct access limits the type of visualization available (by referencing only one stylesheet), and bypasses the MARVIN archive query capabilities.

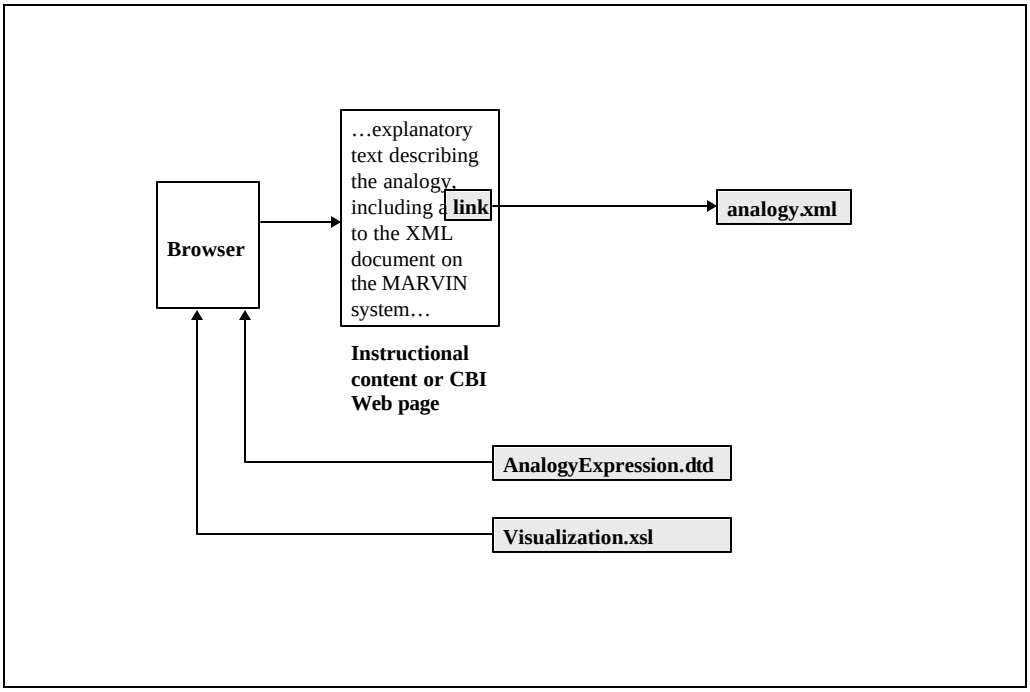


Figure 7.4. Direct Web access of a MARVIN Analogy Expression

The MARVIN system stores analogy expressions in an archive that can be queried by users. The query process is implemented with the Jakarta Lucene text search engine [Apache, 2002], a Java servlet which includes a programmable XML document indexer and query syntax that can select for terms according to their XML element tag and content. For the MARVIN system, this permits analogy expression queries by any expression element or attribute. For example, if we wish to search the archive for analogy expressions whose concepts include *nucleus*, the query string would be 'Concept:nucleus'. As discussed in Chapter 6, Lucene permits Boolean query expressions such as 'Concept:nucleus AND Relation:orbit'. Figure 7.5

illustrates the sequence diagram for a MARVIN query. The user enters the query through the MARVIN search interface. The Tomcat and Cocoon processes pass the query to the Lucene servlet, which queries the document index (generated earlier), sorts the result set, and returns the result set to the Cocoon servlet. Cocoon then requests the search result stylesheet, transforms the list of XML documents in the result set to an HTML Web page, and passes the page back to Tomcat and then to the user's browser.

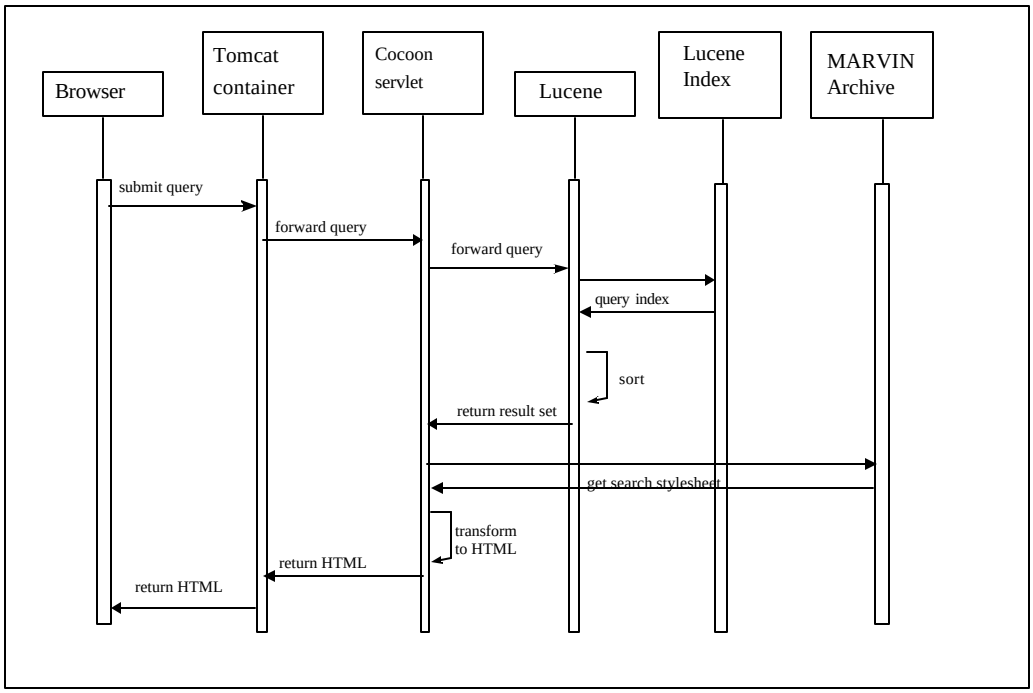


Figure 7.5. Sequence Diagram for MARVIN archive query

The MARVIN user interface, shown in Figure 7.6, consists of a primary Web page divided into three frames. The top frame includes a link to the

MARVIN Help page as well as a link to a pop-up window providing a brief general description of the system. There is also a form for the user to enter a URL for any instructional content Web page that contains links to MARVIN analogy expressions.

The middle frame displays links to a collection of example analogies, used to illustrate MARVIN visualizations. Each example includes a brief description, a link to a reference for the analogy, and one or more links to visualizations produced by the MARVIN system. Figure 7.7 illustrates a tabular visualization of the Rutherford analogy, and Figure 7.8 illustrates a graphical visualization (a JPG image produced from the SVG rendering of the XML analogy expression, of Galileo's analogy.

The bottom frame of the main Web page includes a link to the MARVIN Archive Search (Figure 7.9), a link to a Contributor's page that allows analogy authors to upload new analogy expressions, and links to the Author Evaluation and User Evaluation surveys (discussed in Chapter 8).

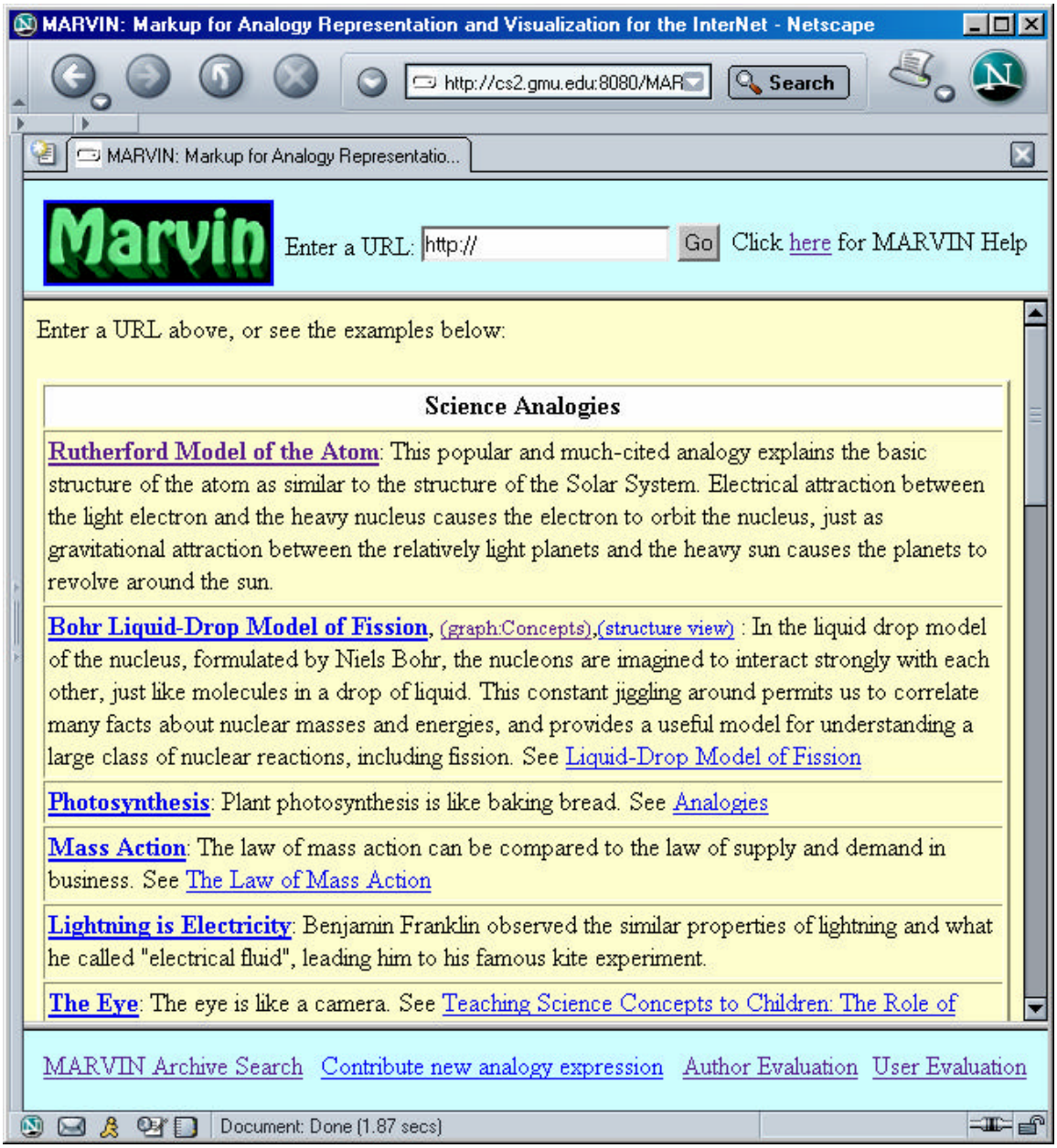


Figure 7.6 The MARVIN User Interface

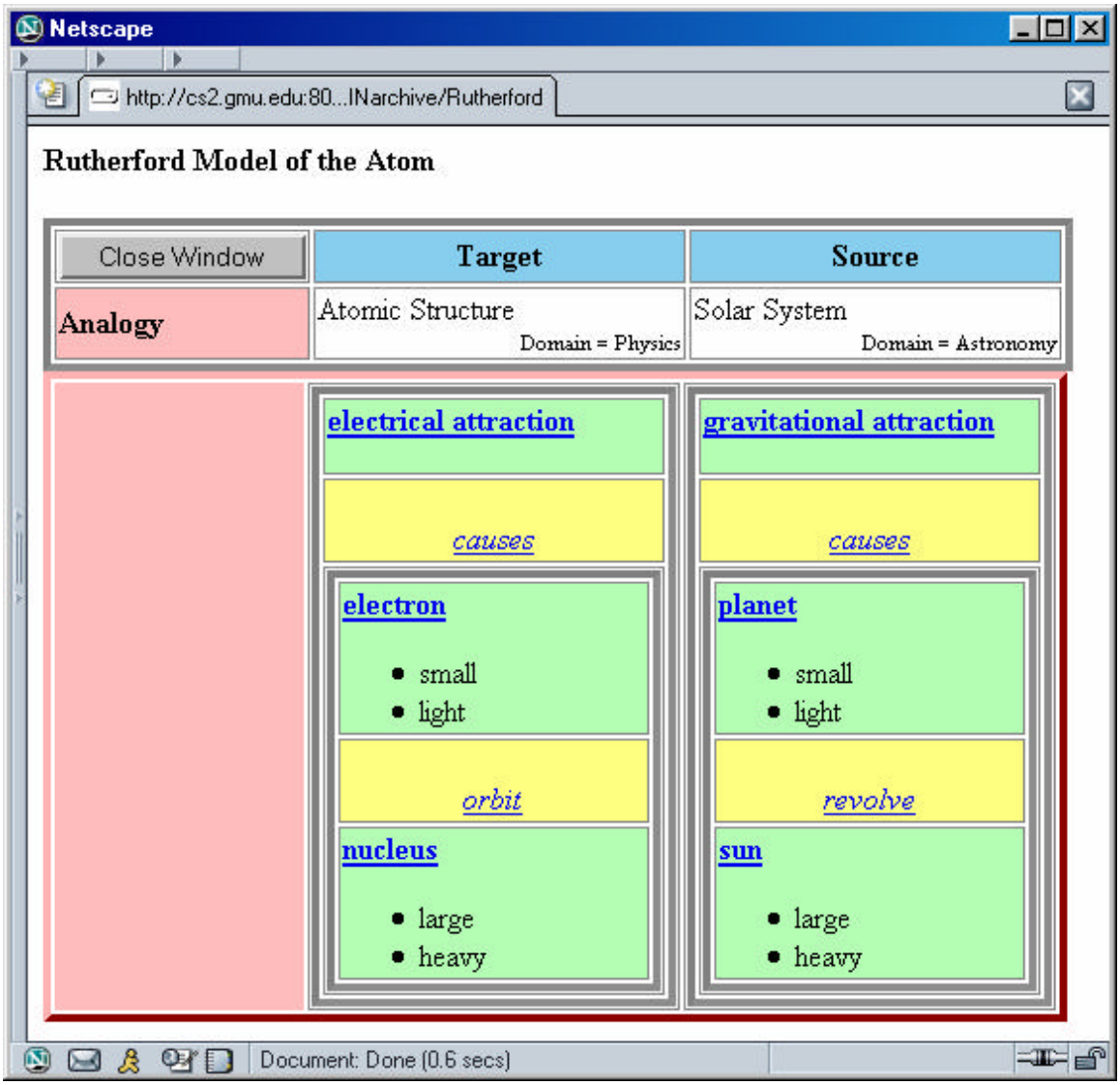


Figure 7.7. A tabular visualization of the Rutherford Analogy

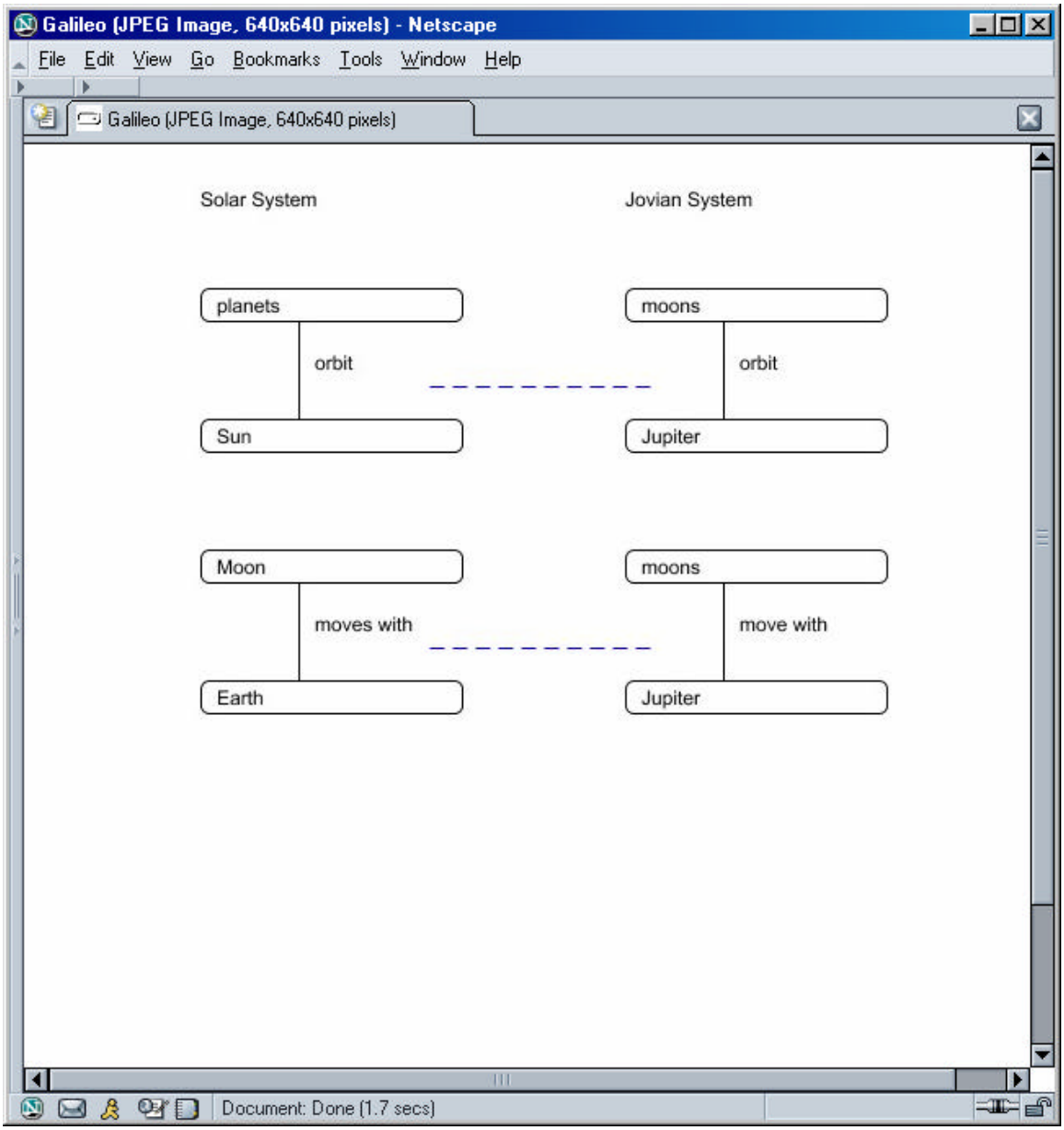


Figure 7.8. A graphical visualization of the Galileo analogy

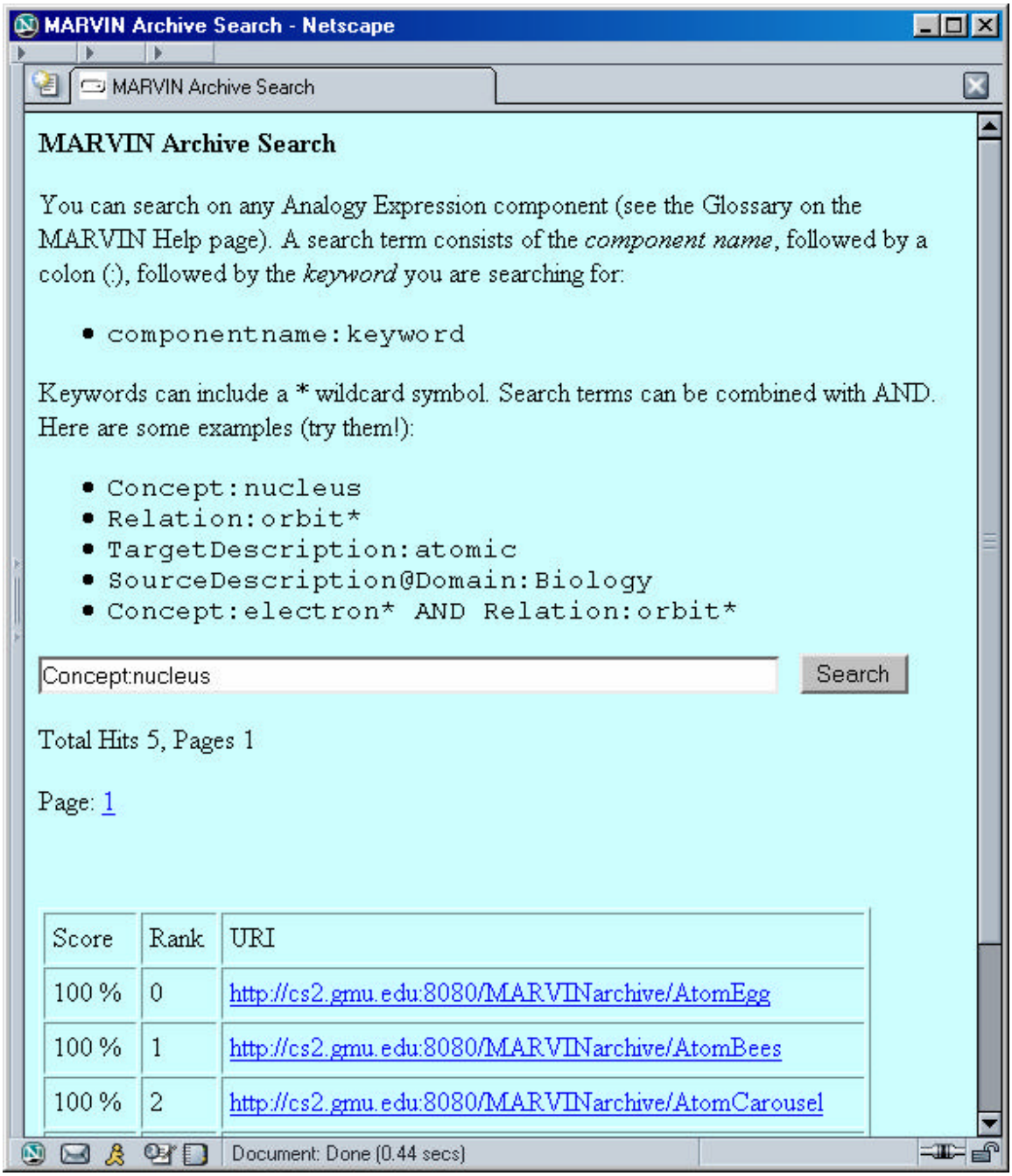


Figure 7.9. The MARVIN Archive Search interface

7.3 Creating MARVIN Analogy Expressions

Analogy practitioners who want to use the MARVIN system must identify the components of any analogy they want to present, including its concepts, concept properties, relations, and especially any of the five analogy structure maps (described in Chapter 3) that they wish to use. These structures include the ConceptSetMap, PrimaryRelationStructureMap, ConceptToRelationStructureMap, RelationToConceptStructureMap, and RelationToRelationStructureMap. Authors familiar with the process of creating XML files using a supplied content model (that is, a DTD) can then create a MARVIN analogy expression using an XML editor such as [XML-Spy, 2002], or [epcEdit, 2002], using the MARVIN analogy expression DTD. For example, the Galileo analogy discussed in Chapter 3 contains a PrimaryRelationStructureMap. Figure 7.10 in the next section shows the MARVIN analogy expression editor being used to create this structure in an analogy expression.

7.3.1 The MARVIN Analogy Editor

Most authors of instructional Web content will probably not be familiar with the structure and syntax of XML. Such users will need a tool to create MARVIN analogy expressions that enable them to easily select the desired

analogy structures and to enter the concepts, properties, and relations.

An Analogy Expression Editor was designed and implemented [Ruan, 2002] that can be downloaded and used by analogy practitioners to create MARVIN analogy expressions [Foxwell, 2002]. The editor lists the five types of maps and displays a set of prestructured panels for each type of map. Each map panel contains labeled text entry fields for the concept and relation components of the map. Figure 7.10 shows the editor being used to enter the PrimaryRelationStructureMap of the Galileo analogy discussed earlier. The editor allows expression authors to easily copy and paste structure maps to create complex analogy expressions. The editor is written in Java, using J2SE 1.4 [Sun, 2002] so that it may be installed and run on any system for which that version is available, including Windows, Mac OS X, Linux, and UNIX.

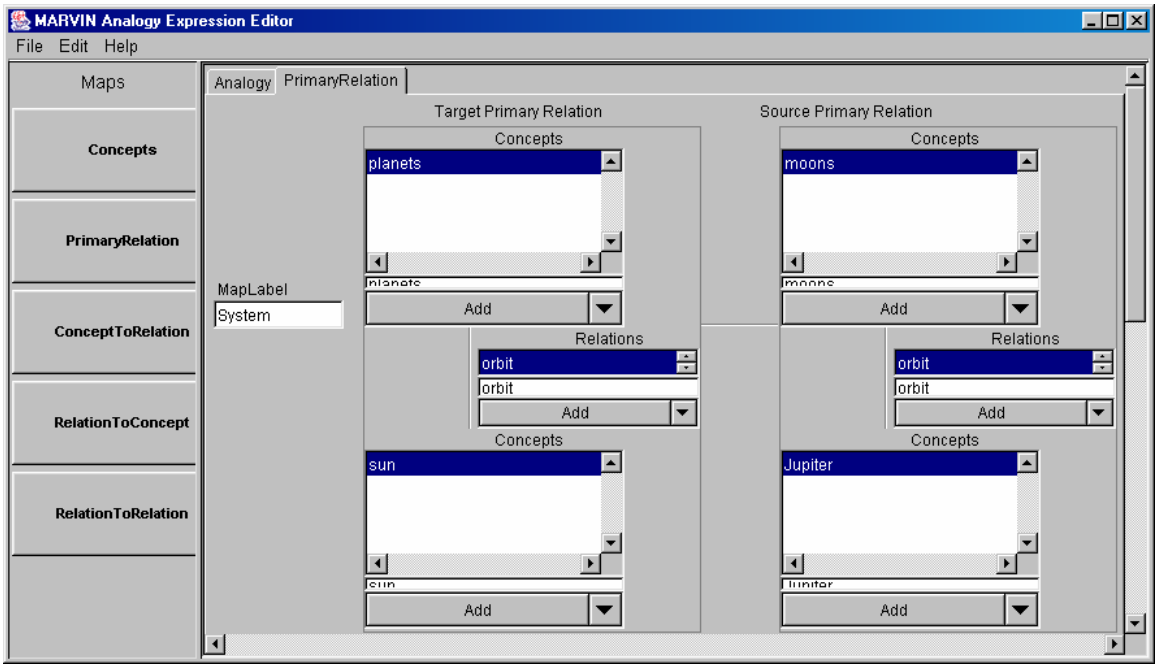


Figure 7.10. The MARVIN Analogy Expression Editor

7.4 Summary

The MARVIN prototype system integrates the necessary components to represent, visualize, and retrieve analogy expressions:

- an XML content model (DTD) describing the structure of analogies
- authoring tools for the creation of XML analogy expressions
- server software supporting a Web interface for the retrieval and visualization of XML analogy expressions

- transformation components that dynamically create different forms of visualizations from the XML expressions

- a query component that permits the retrieval and ranking of expressions

Although the system is still in prototype stage, it has been successfully demonstrated and positively evaluated by potential analogy practitioners. The results of the evaluations are described in the next chapter.

8. MARVIN System Evaluation

The MARVIN system is a prototype, designed as a proof-of-concept for Web-based representation, visualization, and retrieval of analogies implemented within a single system. Functionally, the system provides those three capabilities, but we also need feedback from analogy practitioners about the potential usefulness of such a system. To provide this feedback, a *formative evaluation* of the MARVIN prototype was conducted. A formative evaluation is an evaluation conducted during the development of a system or instructional module [Tessmer, 1993]. It can consist of one or more of the following steps

- expert review by practitioners, authors, or instructors
- one-to-one evaluation by a user with the developer
- small group evaluation by a group of learners or users
- field test of the system or module under realistic learning conditions

The purpose of such evaluations is to provide guidance to the developer concerning the content, functionality, and usability of the system or module under review. This provides an opportunity to adjust or to enhance the

system before making final decisions about implementation. Data collected from formative evaluations can consist of developer notes from evaluator discussions and interviews, user comments, or data gathered using evaluation instruments such as questionnaires. Section 8.1 of this chapter describes the results of a formative evaluation of the MARVIN system prototype.

Additionally, because MARVIN is a Web based system with the potential to serve a large number of users and to archive a large number of analogy expressions, system sizing and retrieval performance must be considered. Section 8.2 of this chapter discusses general system performance issues and presents the results of retrievals from various size archives.

8.1 Formative Evaluation

The formative evaluation of the MARVIN system included demonstrations and discussions with analogy practitioners and users, followed by online evaluation of the MARVIN prototype using a questionnaire for MARVIN analogy expression users and a questionnaire for MARVIN system authors. The general goals of this formative evaluation include

- validation of the target audiences for the MARVIN system

- validation of the usefulness of analogies in practitioner's instructional material
- validation of the general usefulness of MARVIN analogy visualizations, and evaluation of several specific types of visualizations (tabular and graphical)
- validation of the general usefulness of archive retrieval of MARVIN analogy expressions
- validation of the general usefulness of archive retrieval of *alternate* MARVIN analogy expressions

To accomplish these goals, several analogy practitioners (content experts who use analogies in their work or instructional material) were identified, presented with a brief demonstration of the MARVIN system, and interviewed to obtain their comments and suggestions. Additionally, students in several instructional technology courses at GMU's Graduate School of Education (GSE) were given a brief overview and demonstration of the MARVIN prototype and were asked to complete an online questionnaire about the system.

Interviews were conducted with three researchers whose work involved the use and analysis of analogies:

- a professor of Philosophy & Religion at GMU who teaches a

course on the Philosophy of Science, wherein the use of analogy in science discovery is discussed [Rothbart, 2002].

- a professor of Education at GMU/GSE with research interests in problem solving and reasoning in young children (using analogies) [White, 2002]
- a professor of Education at GMU's GSE with research interests in educational technology [Kelly, 2002]

Each of the professors was given a brief description and demonstration of the MARVIN prototype system, and was asked to comment. Each agreed that the MARVIN system would be a useful tool for the presentation and visualization of analogies used in instruction, especially the archive retrieval feature.

Two of the professors [Kelly, 2002][Rothbart, 2002] commented extensively on the use and potential abuse of analogies in instructional material. There was some discussion concerning whether the *representation* of the analogy should include indications of the *validity* of the proposed mappings. As a result of these discussions, and discussed in Chapter 3 and 4, the MARVIN analogy expression representation includes the capability of labeling the relevance of concept *properties* to the analogy, but the validity of the

proposed concept and relation maps themselves belongs outside the representation of the analogy.

Students in three GSE instructional technology courses (Fall'02: EDIT797: Instructional Strategies for the Web, Spring'03: EDIT732: Advanced Instructional Design, and Spring'03: EDIT611: Distance Learning via Networks and Telecommunications) were given brief demonstrations and presentations of the MARVIN prototype system, and were asked to volunteer to provide user or author evaluations of the system. The user evaluation questionnaire, shown in Figure 8.1, was directed at those who are presented with analogies in their instructional material. The author evaluation questionnaire, shown in Figure 8.2, was directed at those who create or select analogies for inclusion in their instructional material. Items on each questionnaire were presented using a 5-point Likert Scale [Babbie, 1983] response: 5: Agree Strongly; 4: Agree Somewhat; 3: Uncertain; 2: Disagree Somewhat; and 1: Disagree Strongly.

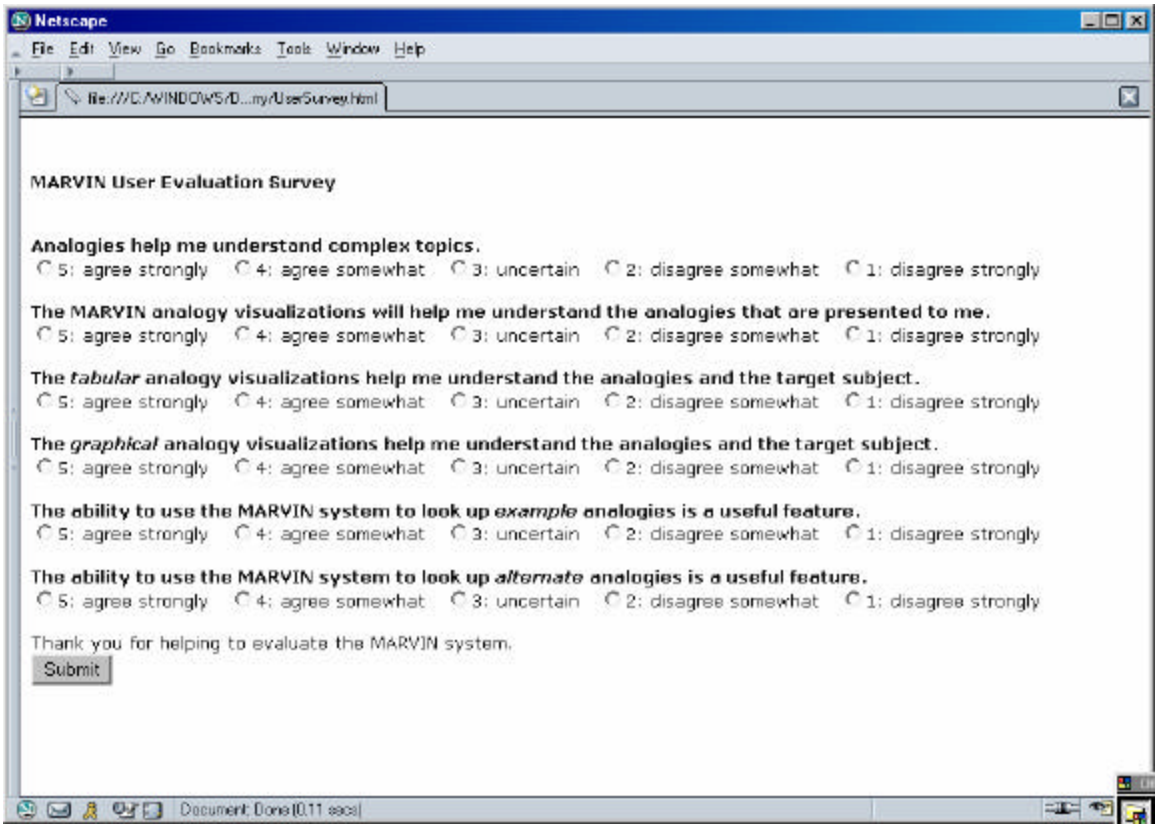
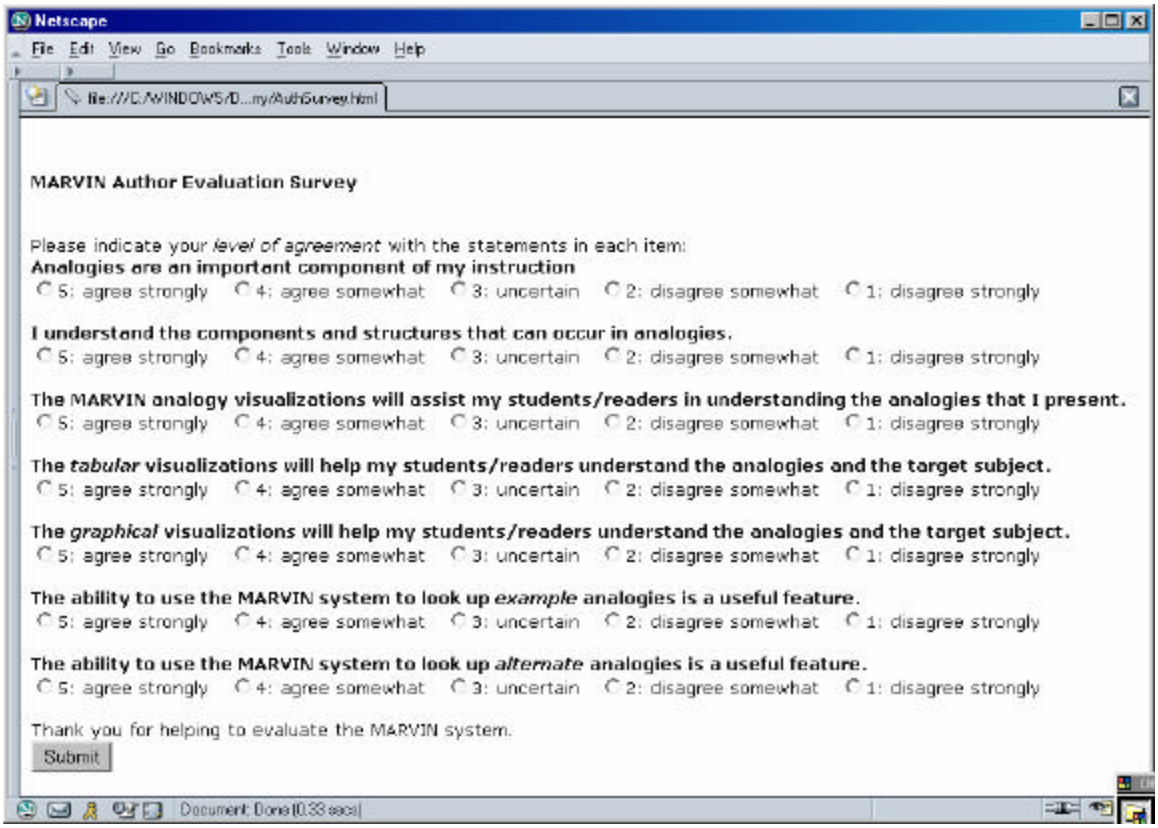


Figure 8.1. The MARVIN User Evaluation Survey



Netscape

File Edit View Go Bookmarks Tools Window Help

file:///C:/WINDOWS/D...my/AuthSurvey.html

MARVIN Author Evaluation Survey

Please indicate your *level of agreement* with the statements in each item:

Analogies are an important component of my instruction

5: agree strongly 4: agree somewhat 3: uncertain 2: disagree somewhat 1: disagree strongly

I understand the components and structures that can occur in analogies.

5: agree strongly 4: agree somewhat 3: uncertain 2: disagree somewhat 1: disagree strongly

The MARVIN analogy visualizations will assist my students/readers in understanding the analogies that I present.

5: agree strongly 4: agree somewhat 3: uncertain 2: disagree somewhat 1: disagree strongly

The *tabular* visualizations will help my students/readers understand the analogies and the target subject.

5: agree strongly 4: agree somewhat 3: uncertain 2: disagree somewhat 1: disagree strongly

The *graphical* visualizations will help my students/readers understand the analogies and the target subject.

5: agree strongly 4: agree somewhat 3: uncertain 2: disagree somewhat 1: disagree strongly

The ability to use the MARVIN system to look up *example* analogies is a useful feature.

5: agree strongly 4: agree somewhat 3: uncertain 2: disagree somewhat 1: disagree strongly

The ability to use the MARVIN system to look up *alternate* analogies is a useful feature.

5: agree strongly 4: agree somewhat 3: uncertain 2: disagree somewhat 1: disagree strongly

Thank you for helping to evaluate the MARVIN system.

Submit

Document: Done (0.33 secs)

Figure 8.2. The MARVIN Author Evaluation Survey

8.1.1 Survey Results

Twenty students responded to the MARVIN User Evaluation Survey.

Results were as follows:

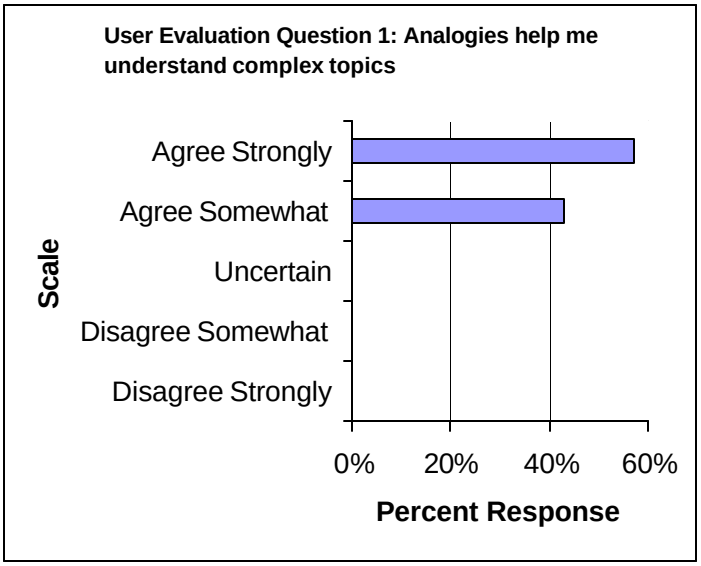


Figure 8.3. Question 1: Analogies help me understand complex topics

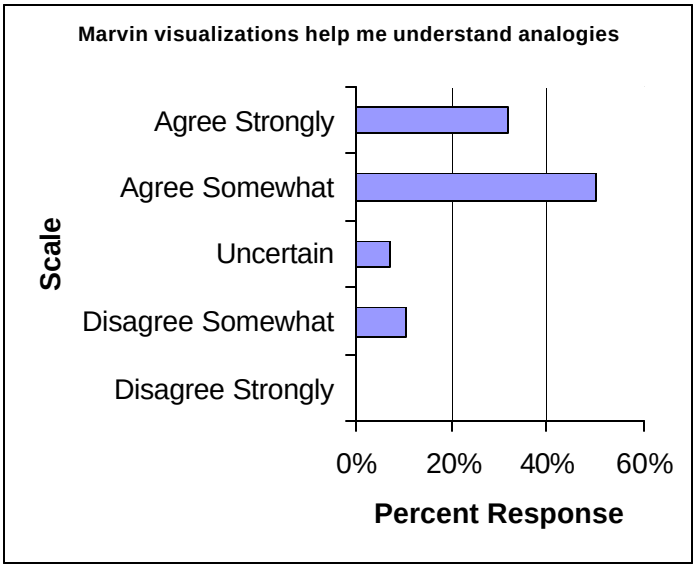


Figure 8.4. Question 2: The MARVIN analogy visualizations will help me understand the analogies that are presented to me.

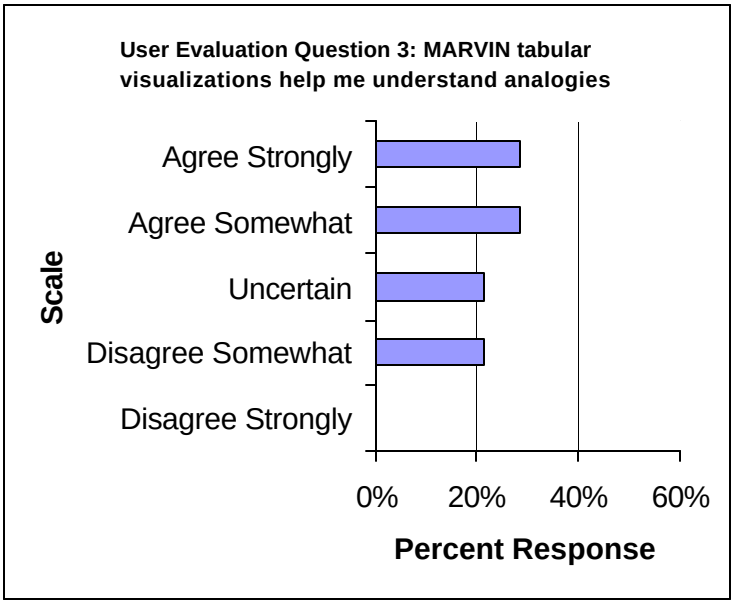


Figure 8.5. Question 3: The tabular analogy visualizations help me understand the analogies that are presented to me

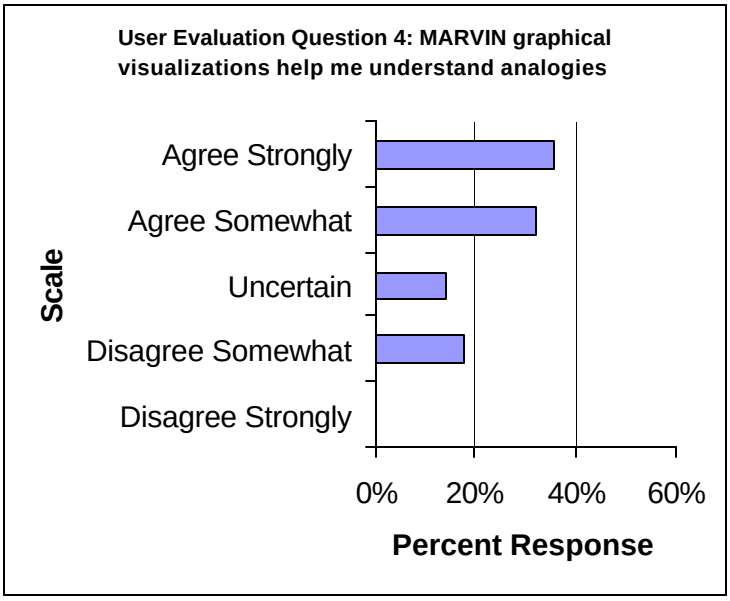


Figure 8.6. Question 4: The graphical analogy visualizations help me understand the analogies and the target subject

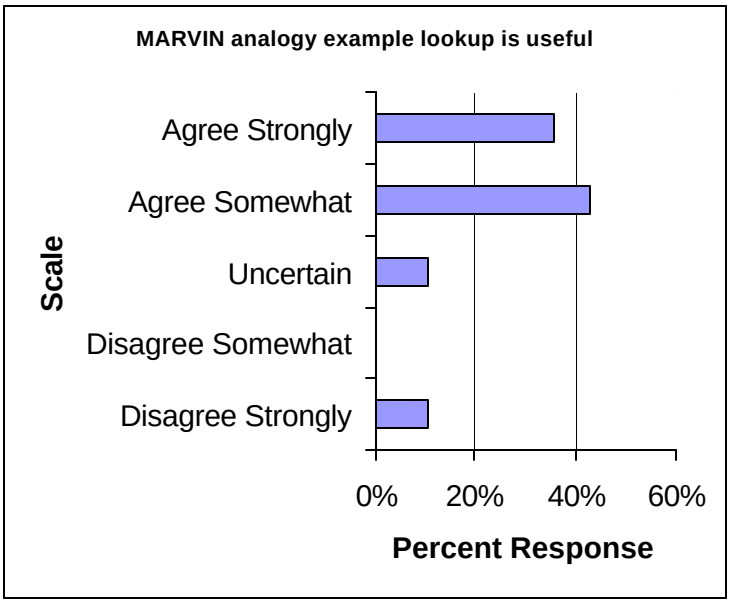


Figure 8.7. Question 5: The ability to use the MARVIN system to look up example analogies is a useful feature

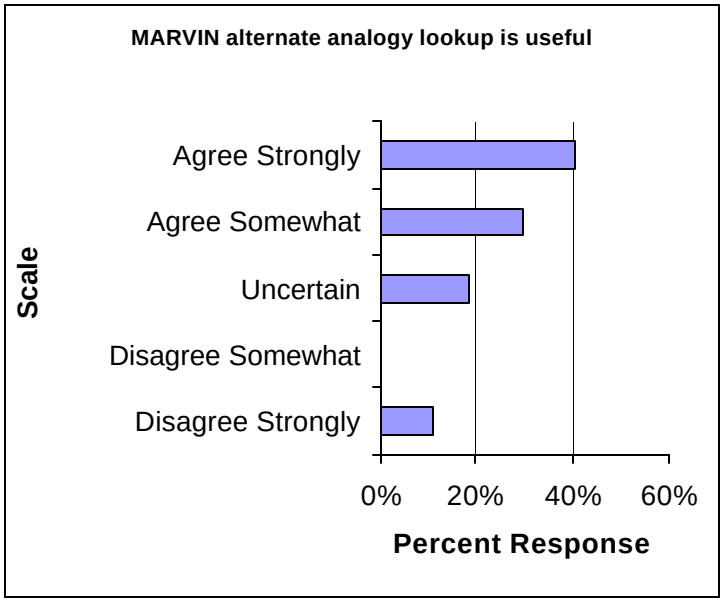


Figure 8.8. Question 6: The ability to use the MARVIN system to look up alternate analogies is a useful feature

8.1.1.1 Discussion of User Survey Results

User Evaluation Question 1 validates the target audience – users of analogies. All respondents agreed (either strongly or somewhat) that analogies helped them understand complex topics. Eighty percent of user respondents agreed that visualizations produced by the MARVIN system would help them understand analogies presented to them; when asked about the specific types of example visualizations presented by the system, 65% agreed that both the tabular and graphical visualizations would be helpful.

There was strong agreement that the MARVIN archive search features were useful, with 80% agreeing about the ability to look up example analogies, and 70% agreeing about the ability to look up alternate analogies. Figure 8.9 summarizes the survey results.

	Question 1	Question 2	Question 3	Question 4	Question 5	Question 6
Responses	28	28	27	28	28	27
Mean	4.57	4.04	3.81	3.86	3.93	3.89
StdDev	0.50	0.92	0.92	1.11	1.21	1.28
Lower and upper bounds for 95% confidence intervals (see below)						
Lower Bound	4.34	3.61	3.38	3.35	3.37	3.28
Upper Bound	4.80	4.46	4.25	4.37	4.49	4.49

Figure 8.9 User Survey response summary

8.1.1.2 Discussion of Author Survey Results

Four respondents identified themselves as content experts -- authors of instructional content that uses analogies. Their responses concerning the usefulness of the MARVIN visualizations were varied, but there was strong agreement concerning the usefulness of looking up example analogies and looking up alternate analogies. Figures 8.10 through 8.16 show the results of the author survey.

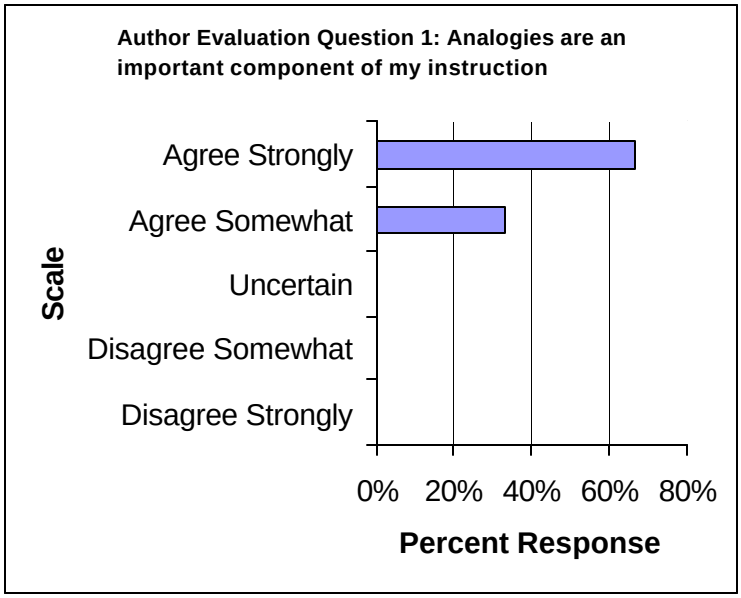


Figure 8.10. Question 1: Analogies are an important component of my instruction

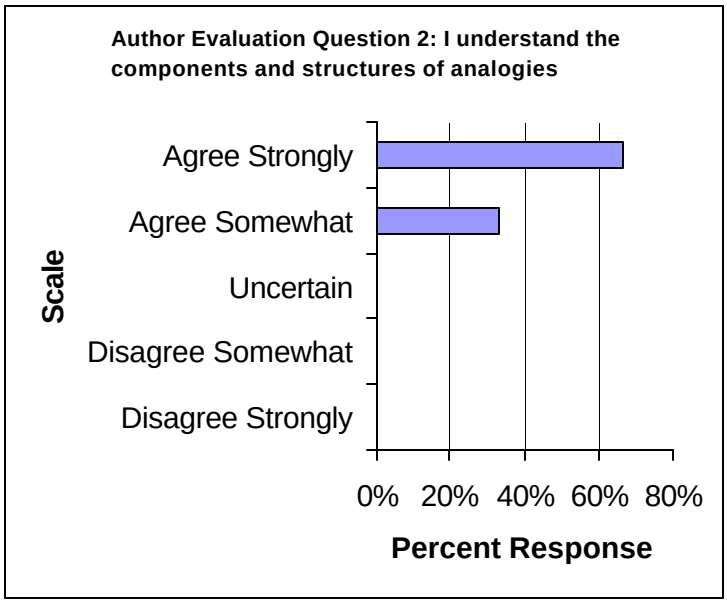


Figure 8.11. Question 2: I understand the components and structures that can occur in analogies

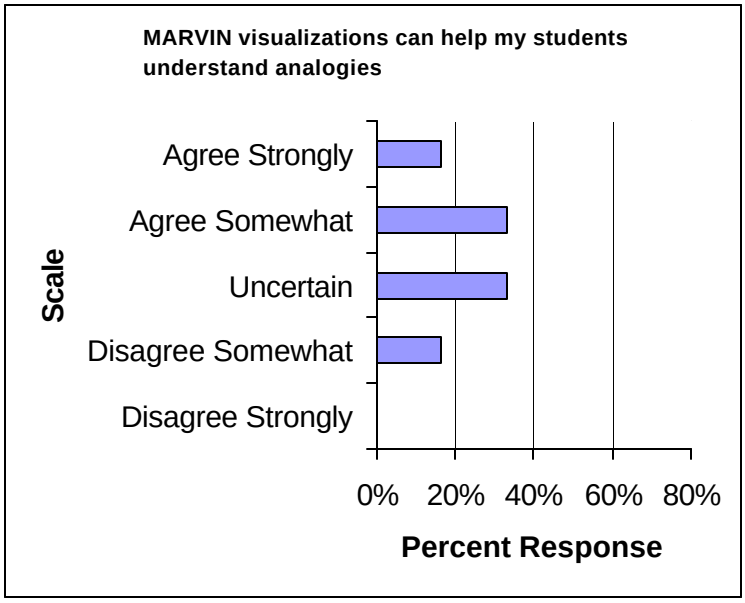


Figure 8.12. Question 3: The MARVIN analogy visualizations will assist my students/readers in understanding the analogies that I present

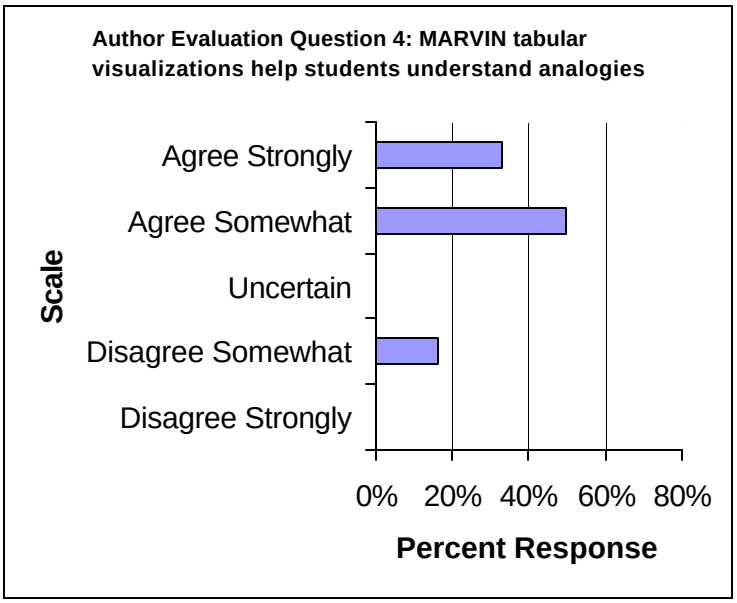


Figure 8.13. Question 4: The tabular visualizations will help my students understand the analogies and the target subject

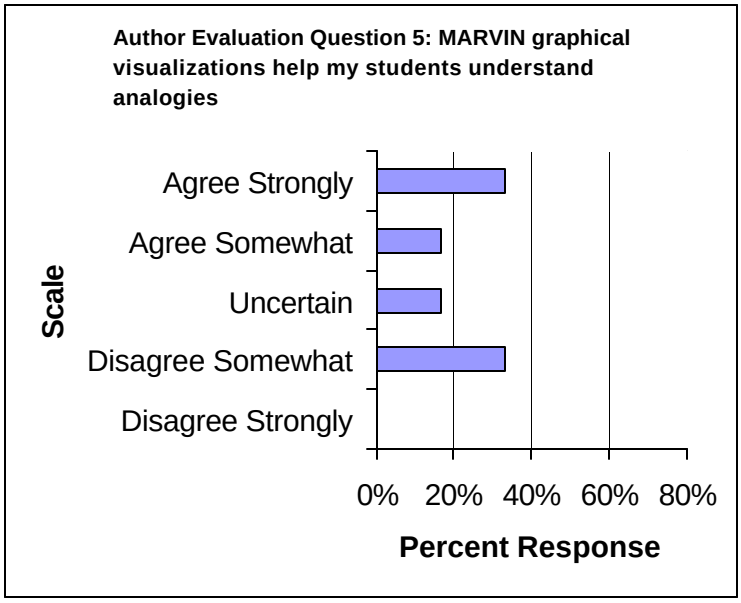


Figure 8.14. Question 5: The graphical visualizations will help my students understand the analogies and the target subject

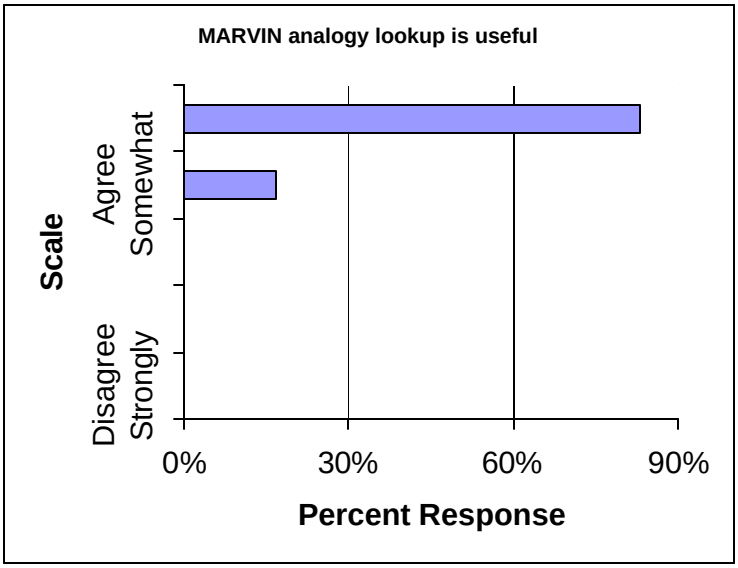


Figure 8.15. Question 6: The ability to use the MARVIN system to look up example analogies is a useful feature

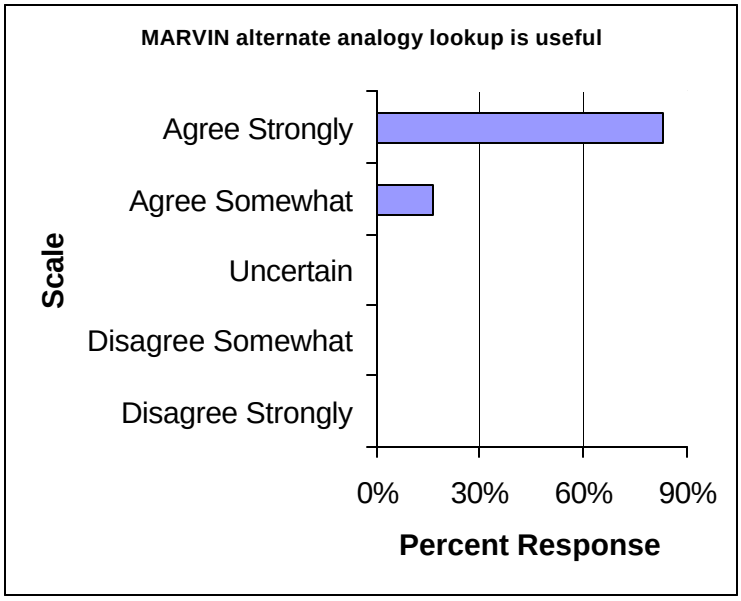


Figure 8.15. Question 7: The ability to use the MARVIN system to look up alternate analogies is a useful feature

8.2 Performance and Scalability

The MARVIN system architecture is designed to use the processing power of the server, avoiding any XML transformation processing by the browser; only HTML and JPG display rendering are required at the browser. The MARVIN system server is implemented using Java servlets (Cocoon and Lucene) running within the Apache Tomcat servlet container. Tomcat performance is sensitive to a variety of factors, notably the speed and number of processors on the server, and the version (and therefore the performance) of the JVM that runs the container. Sub-second response times at hit rates of more than 100/sec are possible even on single-CPU servers [WebPerformance, 2002]. The MARVIN system does not place a high processing burden on the server/container because

- MARVIN Analogy Expression documents are small (generally, a few KB)
- when the Cocoon servlet is first started by the Tomcat container, the servlet parameters are read from the sitemap and the main servlet is compiled *once* by the JVM; subsequent calls to the servlet do not require recompilation
- servlet classes are compiled once as needed when they are requested. In the case of MARVIN requests, the classes that translate XML documents to HTML tables and to SVG

graphics are compiled once, and reused on subsequent calls to translate different documents

- Cocoon caches compiled servlet components such as sitemaps and recompiles them only when generating content from updated sitemaps or stylesheets; changes to the site data do not require servlet recompilation. MARVIN files are static, so once accessed, there is no need for the servlet class that generates a visualization to be recompiled
- there are multiple caching mechanisms in place between the user's browser and the MARVIN system -- the browser cache, the user's proxy cache (if configured), the Tomcat container cache, and the Cocoon cache. Because a transformed analogy expression (that is, an HTML or SVG/JPG visualization) does not change unless the XML expression itself is changed, the browser and intermediate systems are essentially dealing with static, cacheable content. That is, once an analogy expression is transformed, no further processing is required.
- the Lucene servlet used in the MARVIN system is a high-performance text search engine designed for large document collections. There are two components of Lucene performance to consider -- index creation, and searching. In the MARVIN

system, index creation is initiated manually only when a new set of expressions has been added to the archive, and thus does not affect the interactive performance of the system. Lucene is capable of indexing 100 documents per second [Su, 2002] or more; once indexed, query performance has been reported to be fast, according to informal Lucene e-mail user archives, and sub-second query times for large XML archives (larger than 100 MB) have been reported [Jockman, Kimber, and Reynolds, 2002]. We ran experiments, described in the next section, to verify this assertion.

8.2.1 MARVIN Archive Performance

As the surveys indicate, the MARVIN analogy lookup feature is a key component of the system, and therefore must be scalable in order to eventually accommodate a large number of expressions. The Jakarta Lucene text search engine used in MARVIN is an ongoing open source project, with contributions and development managed under the Apache Jakarta Project [Apache Software Foundation, 2002]. Little has been published concerning Lucene performance, other than user-contributed benchmarks [Apache, 2003], a student project paper [Su, 2002], and a commercially sponsored whitepaper [Jockman, et al., 2002]. To characterize the scalability of the

archive lookup component of the MARVIN system, the Lucene engine was tested using various size archives of MARVIN analogy expressions.

To conduct the test, a procedure was needed to generate sample MARVIN analogy expressions. For this purpose, the XML Generator [IBM, 2001] was used. This tool can create test cases for XML applications, generating valid XML documents with random content from an XML DTD. The following steps were repeated ten times for our test:

- create archives of 1000, 2000, 4000, 8000, 16000, 32000, and 64000 random analogy expressions using XML Generator, each conforming to the MARVIN Analogy Expression DTD, but containing random character string content within each element
- copy additional sample MARVIN analogy expressions to each archive
- create an index of each archive
- perform three queries for random content against the index and record the time required to complete each query
    - a simple query (Concept:random-string)
    - a Boolean OR query (Concept:random-string Relation:random-string)

- a Boolean AND query (Concept:random-string AND Relation:random-string)

Figure 8.17 summarizes the results of the query tests

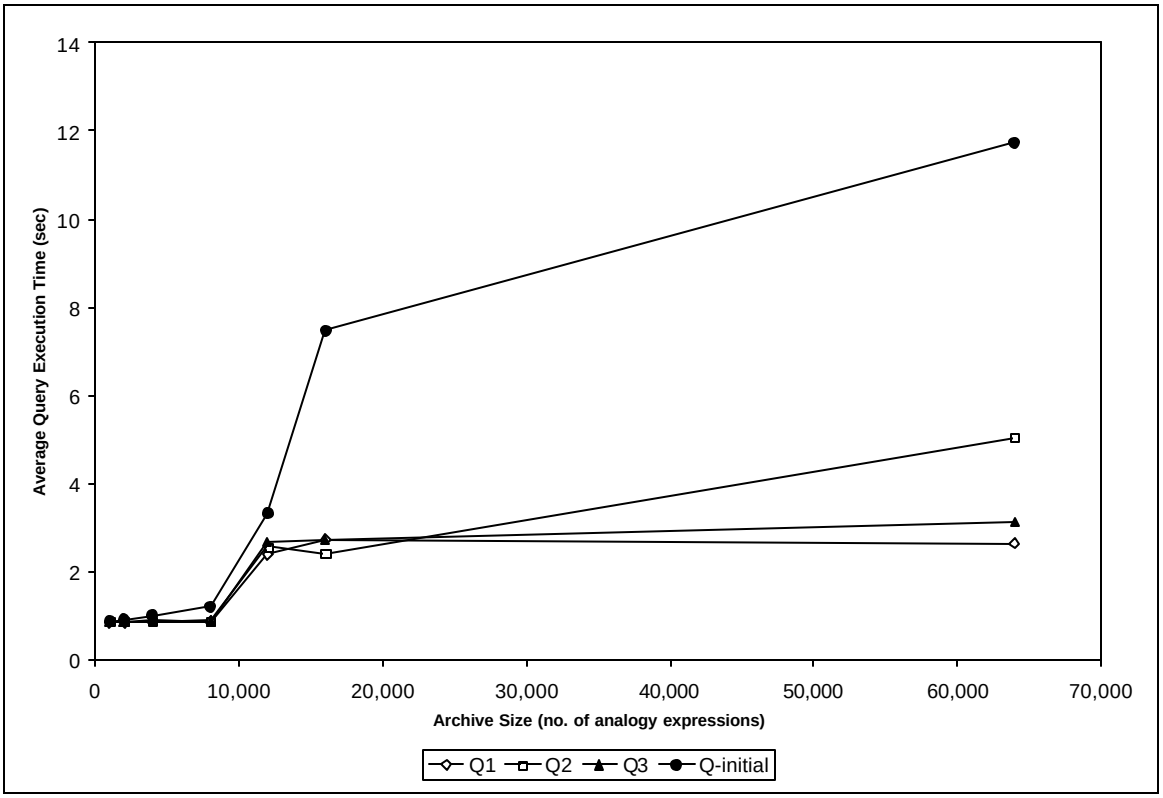


Figure 8.17 Average Query Execution Time (seconds)

The test results show an increase in the query execution time as the size of the archive increases. The initial query for each test had the greatest execution times, but subsequent queries took substantially less time due to file system caching. Queries against a 64000 file archive were generally

completed in less than 5 seconds (except for the initial query of each test run). Lucene is extremely efficient, and MARVIN analogy expression files are relatively small (averaging 10KB each for our examples). We therefore expect that as real analogy expressions are contributed by practitioners to the MARVIN archive, query performance will be acceptable as the archive grows. These tests were conducted using a Sun Microsystems Ultra 60 server, with dual 450 MHz processors, 512 MB memory, and NFS mounted file systems, running the Solaris 9 Operating Environment and version 1.4.1 of the Java JVM.

8.3 Summary

The purpose of the formative evaluation of the MARVIN prototype was to obtain feedback from analogy practitioners and users about the general features of the system. Questionnaire responses and narrative feedback were obtained from both potential end users and authors. Collectively, both users and authors agreed that the MARVIN visualizations would be helpful in learning, although users ranked this feature higher than did authors. Both users and authors agreed that the MARVIN analogy lookup, both for example expressions and for alternate expressions, would be helpful in learning from the presented analogies.

In the narrative comments, respondents commented that the user interface needed improvement, especially the instructions and interface for the archive query. The current MARVIN prototype interface consists of enough functionality for the selection of visualization types and for the explicit text entry of queries; future research into a more efficient and informative interface is needed.

MARVIN system performance is highly dependent on the capabilities of the server supporting the Java servlet environment for the Tomcat and Cocoon components, but testing by the user communities for these components indicates sufficient scalability to support thousands of users given the appropriate equipment. The tests we conducted concerning the performance of the Lucene search engine used in the MARVIN archive search also indicates that the system scales well for large archives of analogy expressions, because of the relatively small document size of the expressions, the efficiency of the Lucene servlet, and file system caching of the archive index.

9. Conclusions and Further Research

In collecting example analogies for this research, we observed an amazing range of human perception and thinking. And what people call analogies ranges from simple similes and metaphors to legitimate, complex, instructional analogies, from the humorous [Willis, 1999] to the truly useful.

A good analogy can generate a cascade of new ideas and hypotheses about its target, not simply for explanatory purposes but for developmental and problem solving purposes as well. For example, comparing a desired form of computer system management to the human autonomic nervous system [Shaw, 2003] can guide engineers in creating subsystems that mimic self-managing biological organs and processes. Abstraction of the basic concepts and relations in an analogy can lead to great discoveries, such as the extension of the eye/camera analogy to the idea of a gravitational lens bending and focusing light from distant galaxies [HubbleSite, 1999].

Our goal was not to simply review and discuss such analogies, however, but to capture them in some useful form, to create methods for expressing them,

visualizing their structure as an aid for remembering and learning, and finding them when we need them.

We started with the problem of representing analogies. Our goal was not complex completeness, for such a representation of the full human mental process and perception is probably not possible. Rather, we wanted to represent the essence of the analogies, to create a compact, human-readable representation. It's hard to make things simple and to keep them simple. Throughout the development of the MARVIN Analogy Expression model there were constant opportunities and suggestions to add a component or feature to the representation, each of which might have contributed to a fuller description of a given analogy but would have added complexity. We wanted to be able to capture analogy simply and quickly because, as we discussed in Chapter 1, an analogy is like a map of the land but it is not the land itself. We use the map and the analogy as a reminder or for orientation, but must eventually get away from it to explore the reality of what we are trying to explain or to learn.

Our representation therefore expresses the basics of an analogy -- its concepts, concept properties, and essential relational structures. We use XML to create a vocabulary for using this expression so that the rich

collection of available XML programming tools and technologies allows us to display and manipulate interesting analogies using the pervasive and growing World Wide Web, enabling communities of analogy practitioners to share analogies using a common language. We designed and implemented the system using an intermediate server between the user's browser and the author's instructional content, that stores, retrieves, and transforms analogies described using our model into a variety of visual forms. This approach frees the end user from the requirement to install special software or browser plugins, and executes all XML transformations on the MARVIN server, and thus places no additional processing burden on the user's system.

The MARVIN prototype was implemented using the Tomcat servlet container and the Cocoon XML framework, allowing the transformation of XML analogy expressions into Web browser viewable HTML tables and JPG graphic diagrams. In evaluating the MARVIN prototype, both analogy users and authors generally agreed that the system is helpful in understanding analogies through the various forms of visualization, and through the ability to look up new or alternate analogies.

The MARVIN prototype provides a minimal user interface sufficient to demonstrate the basic ideas of the system -- representation, visualization,

and retrieval. Comments from the evaluators indicated that further development of the interface is needed to improve the ease and understanding of querying the analogy archive.

Additional research is also needed in characterizing the "quality" or "pertinence" of an analogy, a difficult task because perceptions of analogies are personal and contextual for each user, and highly dependent on the user's background knowledge and experience. If a person claims, for example, that his perception that the earth's crust is like a piecrust helps him understand plate tectonics, the existence of the perception and its initial utility to the user can't be denied, even though more "accurate" analogies might be constructed. Whether a characterization of "analogical accuracy" even belongs in a representation of perceived analogy requires further research; currently, we only characterize in our expression the validity of concept property comparisons.

Finally, the full range of human conceived analogies encompasses more than the types we have characterized here. Geometric, process, and even auditory or linguistic analogies might be able to be represented using XML content models. Representing them in that way would permit the application of the techniques explored in the MARVIN system to a wider range of human

conceived analogies, benefiting the growing community of analogy practitioners and researchers in computer science, cognitive science, education, and general practice.

References

References

Apache Software Foundation, Xerces2 Java Parser 2.0.1 Release, http://xml.apache.org/xerces2-j/, 2001

Apache Software Foundation, Jakarta Lucene, http://jakarta.apache.org/lucene/docs/index.html, 2002

Apache Software Foundation, Xalan-Java Version 2, http://xml.apache.org/xalan-j/, 2002

Ashley, K., Modeling Legal Arguments: Reasoning with Cases and Hypotheticals, MIT Press, Cambridge, MA, 1991

Ausubel, D., *The use of advance organizers in the learning and retention of meaningful verbal material*, Educational Psychology, 51, 1960

Babbie, E., The Practice of Social Research, Wadsworth Publishing Company, Belmont CA, 1983

Berners-Lee, T., Hendler, J., Lassila, O., *The Semantic Web*, Scientific American, May 2001, http://www.scientificamerican.com/2001/0501issue/0501berners-lee.html

Bohr, N., Nobel Lecture, December 11, 1922, http://www.nobel.se/physics/laureates/1922/bohr-lecture.html.

Bransford, J., Brown, A., and Cocking, R., How People Learn: Brain, Mind, Experience, and School. National Academy Press, Washington, DC, 2000

Bruner, J. C., Actual minds, possible worlds. Harvard University Press, Cambridge, MA, 1986

Carbonell, J., *Derivational Analogy: A theory of reconstructive problem solving and expertise acquisition*. In Machine Learning: An Artificial Intelligence Approach (Vol. 81, Michalski, R.S., Carbonell, J.G., and Mitchell, T.M., Eds., Morgan Kaufmann, 1986

Catrambone, R., & Holyoak, K., *Overcoming contextual limitations on problem-solving transfer.* Journal of Experimental Psychology: Learning, Memory, and Cognition, *15*, 1147-1156, 1989

Clark, R., *To the Moon and Back: The Doc Tires of Bad Analogies*, http://legacy.poynter.org/dr_ink/030602.htm, 2002

Collins, A., and Loftus, E., *A Spreading-activation Theory of Semantic Processing*, Psychological Review, 82(6), 407-428, 1975

Cover, R., *Cover Pages*, http://xml.coverpages.org, 2002

Cox, M., *Paradigm Shifts and 9/11: International Relations After the Twin Towers*, Security Dialog 33(2), 2002

Craig, D, Nersessian, N., and Catrambone, R., *The Role of Diagrammatic Affordances in Analogy*, Proceedings of the 24th Annual Conference of the Cognitive Science Society, Fairfax, VA, 2002

Daconta, M., *Are Elements and Attributes Interchangeable?*, XML Journal, Vol. 2, Issue 7, July 2001

Darwin, C., Origin of Species, 1859

Decker, S., Melnik, S., et al., *The Semantic Web: The Roles of XML and RDF*, IEEE Internet Computing, September-October 2000

Dijkstra, E., *GOTO Statement Considered Harmful*, Communications of the ACM, March 1968

Dorin, H., Demmin, P., and Gabel, D., Chemistry: The Study of Matter, Prentice Hall, Inc., Englewood Cliffs, NJ, 1990

Dunbar, K., *The analogical paradox: Why analogy is so easy in naturalistic settings, yet so difficult in the psychology laboratory*. In D. Gentner, Holyoak, K.J., ,& Kokinov, B. (eds) Analogy: Perspectives from Cognitive Science. MIT Press, Cambridge MA, 2001

Dunn, T., *The Leap to a More Meaningful Web*, IEEE Internet Computing, September 2002

epcEdit, http://www.epcedit.com/, March 2002

Eisenberg, J., SVG Essentials, O'Reilly and Associates, San Francisco, CA, 2002

Evans, T., *A heuristic program to solve geometry analogy problems*, Semantic Information Processing, Minsky, M., ed., MIT Press, Cambridge MA, 1968

Forbus, K., Gentner, D., and Law, K., *MAC/FAC: A Model of Similarity Based Retrieval*, Cognitive Science 19(2):141-205, 1995

Forbus, K., *Exploring Analogy in the Large*, in The Analogical Mind: Perspectives from Cognitive Science. MIT Press, Cambridge, MA, 2001

Forbus, K., Mostek, T. and Ferguson, R., *An analogy ontology for integrating analogical processing and first-principles reasoning.* Proceedings of IAAI-02, July 2002

Foxwell, H., *Web-based Representation, Retrieval, and Visualization of Analogies*, http://cs.gmu.edu/~hfoxwell/, March 2002

French, R., The Subtlety of Sameness: A Theory and Computer Model of Analogy-Making. MIT Press, Cambridge, MA, 1995

Galileo, G. 1632, *Dialogues concerning the two chief world systems*, trans. S. Drake, 2nd edition, University of California Press, 1967

Gentner, D., The *mechanisms of analogical learning*. In S. Vosniadou & A. Ortony (Eds.), Similarity and Analogical Reasoning. New York: Cambridge University Press, 1989

Gentner, D., and Jeziorski, M., *The Shift from Metaphor to Analogy in Western Science*. In Ortony, A. (ed), Metaphor and Thought, Cambridge University Press, Cambridge, UK, 1993

Gentner, D., Holyoak, K., and Kokinov, J. (Eds.), The Analogical Mind: Perspectives from Cognitive Science. MIT Press, Cambridge, MA, 2000

Gentner, D., Stevens, A., Editors. Mental Models. Lawrence Erlbaum Associates, London, England, 1983

Gentner, Dedre, and Gentner, Donald, *Flowing Waters or Teeming Crowds: Mental Models of Electricity*, in Gentner 1993

Gentner, D., *Structure-mapping: A Theoretical Framework for Analogy*. Cognitive Science, 7, 155-170, 1983

Gentner, D., *The Structure-Mapping Engine: Algorithm and Examples.* Artificial Intelligence, 41, 1-63, 1989

Glynn, S., *Explaining science concepts: A teaching with analogies model*. In S. M. Glynn, R. H. Yeany, B. K. Britton, (Eds.), The Psychology of Learning Science (pp. 219- 239). Lawrence Erlbaum Associates, Hillsdale, NJ, 1991

Glynn, S. M., & Takahashi, T., *Learning from analogy-enhanced science text*. Journal of Research in Science Teaching, 35, 1129-1149, 1998

Glynn, S. M., Teaching Science with Analogies: A Resource for Teachers and Textbook Authors, http://curry.edschool.virginia.edu/go/clic/nrrc/scin_ir7.html, 1997

Google, *Google Technology*, http://www.google.com/technology/index.html, 2002

Google, http://www.google.com/, March 2002

Gordin, D., Gomez, L., et al., *Using the World Wide Web to build learning communities in K-12*. Journal of Computer-Mediated Communications, 2(3): http://www.ascusc.org/jcmc/vol2/issue3/gordin.html, 1996

Halasz F., and Moran, T, *Analogy Considered Harmful*, Proceedings of the ACM Conference on Human Factors in Computer Systems, March 15-17, 1982

Hecht, A., XSL *Transformations (XSLT) in Mozilla*, http://www.mozilla.org/projects/xslt/, 2002

Heflin, J., Hendler, J., and Luke, J., *SHOE: A Knowledge Representation Language for Internet Applications*, Technical Report CS-TR-4078 (UMIACS TR-99-71). http://www.cs.umd.edu/projects/plus/SHOE/pubs/#tr99, 1999

Hegarty, M., Mental *Visualizations and External Visualizations*, Proceedings of the 24th Annual Conference of the Cognitive Science Society, Fairfax, VA, 2002

Herr, N., *Sourcebook for Teaching Science: Strategies, Activities, and Internet Resources*, http://www.csun.edu/~vceed002/, 2001

Hesse. M., Models and Analogies in Science. University of Notre Dame Press, Notre Dame, IN, 1966

Hoffman, R., *Monster Analogies*, AI Magazine, Fall 1995

Hofstadter, D., *Epilog: Analogy as the Core of Cognition*, in The Analogical Mind: Perspectives from Cognitive Science. MIT Press, Cambridge, MA, 2001

Hofstadter, D., Fluid Concepts and Creative Analogies. Basic Books, NY, NY, 1995

Holyoak, K., and Thagard, P., Mental Leaps: Analogy in Creative Thought. MIT Press, Cambridge, MA, 1995

HubbleSite, *Natural Lenses in Space Stretch Hubble's View of the Universe*, http://hubblesite.org/newscenter/archive/1999/18/text, 1999

IEEE, 1484.12.1 - 2002 Learning *Object Metadata (LOM) Standard*, http://ltsc.ieee.org/wg12/index.html, 2002

ISO, Extended BNF, ISO/IEC 14977:1996(E), ISO: Geneva, 1996

ISO, ISO-1087: Terminology Vocabulary, ISO: Geneva, 1990

Jockman, B., Kimber, W., and Rynolds, J., *Case Study: Enabling Low-Cost XML-Aware Searching Capable of Complex Querying*, Isogen, International, http://www.isogen.com/papers/xml-aware_searching.html, March 2002

Joy, W., The Java Language Specification, Addison-Wesley Publishing Co., 2000

Kay, M., XSLT, Second Edition. Wrox Press LTD, Birmingham, UK, 2001

Holyoak, K., and Thagard, P., *The Analogical Mind*. American Psychologist, 52(1):35-44, 1997

Kelly, A., Personal communication, September 16, 2002

Kepler, J., Mysterium Cosmographicum, 1596

Klein, P., and Milligan, C., *The Role of Analogies in Writing to Learn*, International Conference on Ontological, Epistemological, Linguistic and Pedagogical Considerations of Language and Science Literacy, Victoria, British Columbia, Canada, http://www.educ.uvic.ca/faculty/lyore/sciencelanguage/, 2002

Koike, V., *A Conversion Tool from DTD to XML Schema*, http://www.w3.org/2000/04/schema_hack/, 2000

Koushiappas, S., and Cohen, J., *Fermi Molecular Dynamics in Artificial Fission of Light Nuclei*, Eastern Oregon University Science Journal, 1999-02-14, 1999

Lakoff, G., and Johnson, M., Philosophy in the Flesh. Basic Books, NY, NY, 1999

Lenat, D. and Guha, R., Building Large Knowledge Based Systems: Representation and Inference in the Cyc Project. Addison-Wesley Publishing, Reading, Massachusetts, 1990

Marshall, J., *Metacat: A Self-Watching Cognitive Architecture for Analogy-Making and High-Level Perception*. PhD Dissertation, Indiana University. Bloomington, IN, 1999

Martin, P., and Ecklund, P., *WebKB*, http://meganesia.int.gu.edu.au/~phmartin/WebKB/, 1999

Matocha, J., Camp, T., and Hooper, R., *Extended Analogy: An Alternative Lecture Method*, Proceedings of the 29th SIGCSE Technical Symposium on Computer Science Education, pp. 262-266, March 1998

Menasce, D., Capacity Planning for Web Performance. Prentice Hall, Upper Saddle River, NJ, 1998

Mertz, D., *XML Matters #7: Comparing W3C XML Schemas and Document Type Definitions*,

http://www.ibm.com/developerworks/library/x-matters7.html, March 2001

Miller, G., *WordNet: a lexical database for the English language*, Princeton University, http://www.cogsci.princeton.edu/~wn/, 2001

Minsky, M., The Society of Mind, Simon & Schuster, New York, NY, 1985

Mitchell, M., *Analogy-Making as a Complex Adaptive System*, in Segal, L., and Cohen, I. (editors), Design Principles for the Immune System and Other Distributed Autonomous Systems. Oxford University Press, NY, NY, 1999

Morphon, http://www.morphon.com/, March 2002

Mozilla Organization, *Mozilla Browser*, http://mozilla.org/releases/, 2003

Murphy, G., The Big Book of Concepts, MIT Press, Cambridge, MA, 2002

Murray-Rust, P. *Chemical Markup Language*, in "Electronic communication technologies: techniques and technologies for the 21st century", M. Mitchard, Ed., Interpharm Press, Buffalo Grove, IL, 1998

Neustadt, R., Thinking in Time: The Uses of History for Decision Makers, Free Press, 1988

Nottis, K., and McFarland, J., *A Comparative Analysis of Pre-Service Teacher Analogies Generated for Process and Structure Concepts,* Electronic Journal of Science Education, Vol. 5, No. 4, June 2001

OpenCyc, *OpenCyc: The Project*, http://www.opencyc.org, 2003

Paris, N., *Elaborate Analogies in Science Text: Tools for Enhancing Preservice Teachers' Knowledge and Attitudes*, Doctoral Dissertation, University of Georgia, 2000

Piez, W., Personal email, January 7, 2002

Popov, D., *Using XML as the Core Language of Knowledge Representation in Artificial Intelligence,* Proceedings of the 2nd International workshop on Computer Science and Information Technologies (CSIT'2000). Ufa, Russia, 2000

Ramakrishnan, R., and Gehrke, J., Database Management Systems, 3rd Edition, McGraw-Hill Companies, Inc., New York NY, 2003

Record, J., Perils of Reasoning by Historical Analogy: Munich, Vietnam, and American Use of Force Since 1945, Air University, Montgomery, AL, 1998

Record, J., Making War, Thinking History Munich, Vietnam, and Presidential Uses of Force from Korea to Kosovo. Naval Institute Press, Annapolis, MD, 2002

Roblyer, M., and Edwards, J., Integrating Educational Technology into Teaching, Merrill, Inc., Upper Saddle River, NJ, 2000

Rothbart, D., Personal communication, September 16, 2002

Rourke, J., and Taylor, A., *Do Historical Analogies Influence Political Action?: Congress, the Persian Gulf War, and the Use of Munich and Vietnam Analogies*, Midwest Political Science Association Convention, Chicago, IL, April 1994

Ruan, H, Analogy Expression Editor, 2002

Ruhl, T., *The Altoona List of Medical Analogies*, http://www.altoonafp.org/analogies.htm, 2002

Salton, G., and McGill, M., Introduction to Modern Information Retrieval. McGraw-Hill Book Company, New York, NY, 1983

Schank, R., *Engines for Education*, Institute for the Learning Sciences, Northwestern University, http://www.ils.nwu.edu/~e_for_e/index.html, November 2000

Shaw, Jack, *The Autonomic Enterprise*,

http://www.e-com.com/autonomic.html, 2003

Staab, S., *Ontologies' KISSES in Standardization*, IEEE Intelligent Systems, March/April 2002

Su, D., *Performance Analysis and Optimization on Lucene*, http://www.stanford.edu/class/cs276a/projects/reports/, 2002

Sun Microsystems, *Java 2 Platform, Standard Edition*, *http://java.sun.com/j2se/*, 2003

Sun Microsystems, *Java Servlet Technology, Implementations and Specifications*, http://java.sun.com/products/servlet/, 2003

Suter, R., in Walker, L., *The Lord of the Webs*", Washington Post, Thursday, January 30, 2003

Tarski, A., *Logic, Semantics, Metamathematics.* Oxford University Press, London, UK, 1956

Tecuci, G., Building Intelligent Agents. Academic Press, San Diego, CA, 1998

Tessmer, M., Planning and Conducting Formative Evaluations, Kogan Page Ltd., London, UK, 1993

Thiele, R., and Treagust, D., *An Interpretive Examination of High School Chemistry Teachers' Analogical Explanations*, Journal of Research in Science Teaching, 31, 227-242, 1994

TopicMaps.Org, Authoring Group, *XML Topic Maps (XTM) 1.0*, http://www.topicmaps.org/xtm/index.html, 2001

Tversky, B., Heiser, J., Lee, P., and Zacks, J., *Diagrams to Augment Cognition*, Proceedings of the 24th Annual Conference of the Cognitive Science Society, Fairfax, VA, 2002

US Department of Justice, *Justice XML Data Model*, http://it.ojp.gov/topic.jsp?topic_id=43, 2002

VIAs, Inc., *Discovering Lewis & Clark*, http://www.lewis-clark.org, 1998

von Neumann, J., The Computer and the Brain, Yale University Press, New Haven CT, 1958

Vosniadou, S., and Ortony, A., eds., Similarity and Analogical Reasoning. Cambridge: Cambridge University Press, 1989

W3C, *Recommendation 24 December 1999*, http://www.w3.org/TR/html401/, 1999

W3C, Mathematical *Markup Language (MathML) Version 2.0*, *W3C Recommendation 21*, February 2001

W3C, *Scalable Vector Graphics (SVG) 1.0 Specification, W3C Recommendation 04 September 2001*, http://www.w3.org/TR/SVG

Watson, I., and Marir, F., *Case-Based Reasoning: A Review*, Knowledge Engineering Review, Vol. 9, No. 4, 1994, http://www.ai-cbr.org/classroom/cbr-review.html

WebPerformance, http://webperformanceinc.com, 2002

Wenger, E., Communities of practice: learning, meaning, and identity. Cambridge: Cambridge University Press, 1998

White, C., Personal communication, September 17, 2002

Willis, W., World's Worst Analogies, http://www.worsleyschool.net/socialarts/poorwriting/poorwriting.html, 1999

Winston, P., *Learning and Reasoning by Analogy*, Communications of the ACM, Volume 23, Number 12, December 1980

Wolverton, M., *Retrieving Semantically Distant Analogies*. Doctoral Dissertation, Department of Computer Science, Stanford University,May 1994

WordNet, *WordNet: A Lexical Database for the English Language*, http://www.cogsci.princeton.edu/~wn/, 2001

World Wide Web Consortium (W3C), *Extensible Markup Language (XML) 1.0 (Second Edition), W3C Recommendation 6 October 2000*, http://www.w3.org/TR/2000/REC-xml-20001006

World Wide Web Consortium (W3C), *HTML 4.01 Specification*

World Wide Web Consortium (W3C), *Semantic Web*, February 2002, http://ww.w3.org/2001/sw

World Wide Web Consortium (W3C), *XML Path Language (XPath) Version 1.0, W3C Recommendation 16 November 1999*, http://www.w3.org/TR/xpath, 1999

World Wide Web Consortium (W3C), *XML Schema*, http://www.w3.org/XML/Schema, 2001

World Wide Web Consortium (W3C), *XML-QL: A Query Language for XML*, http://www.w3.org/TR/NOTE-xml-ql/, August 1998

World Wide Web Consortium (W3C), *XSL Transformations (XSLT) Version 1.0, W3C Recommendation 16 November 1999*, http://www.w3.org/TR/xslt, 1999

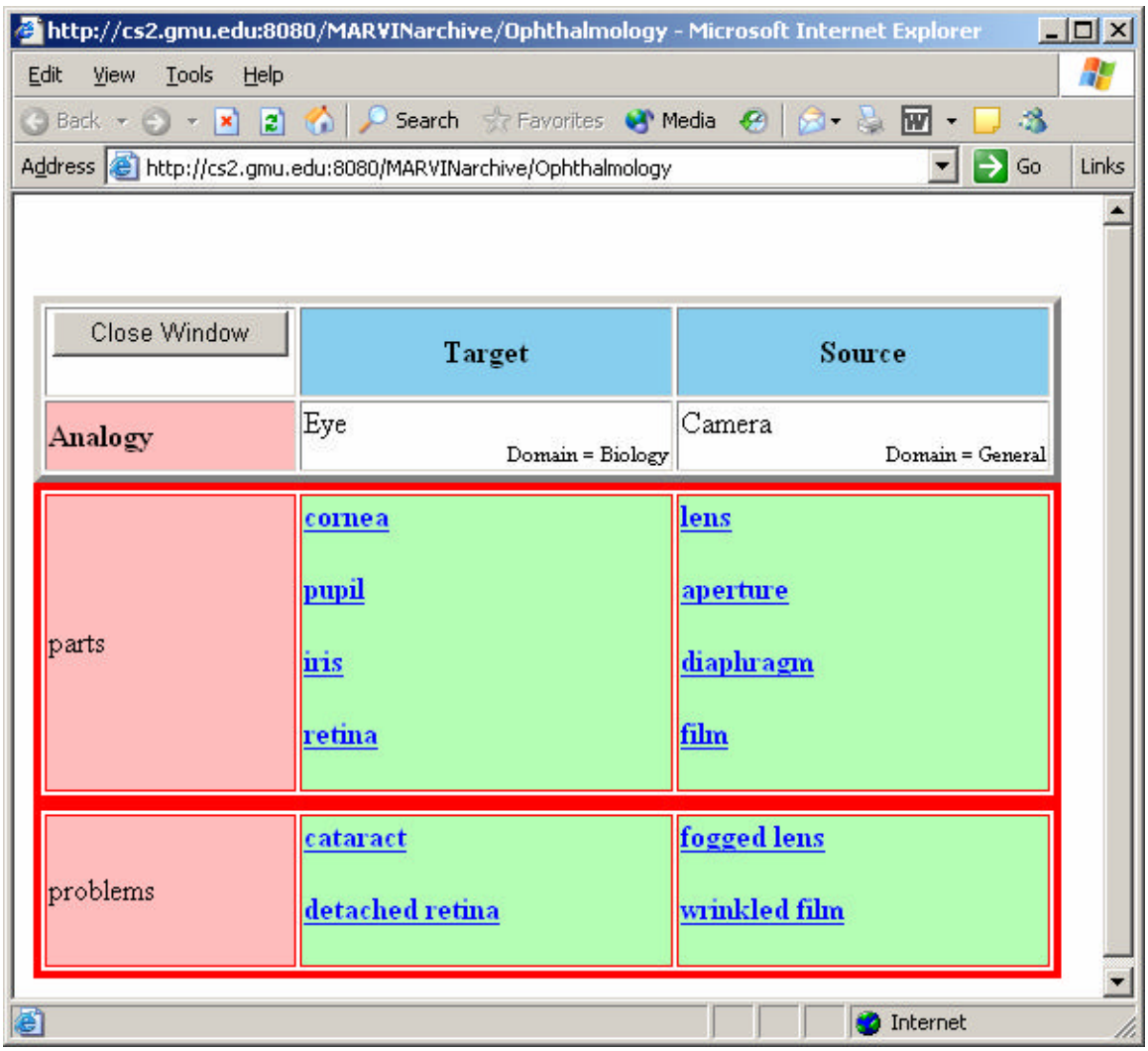
XMLgenerator, http://www.alphaworks.ibm.com/tech/xmlgenerator, March 2002

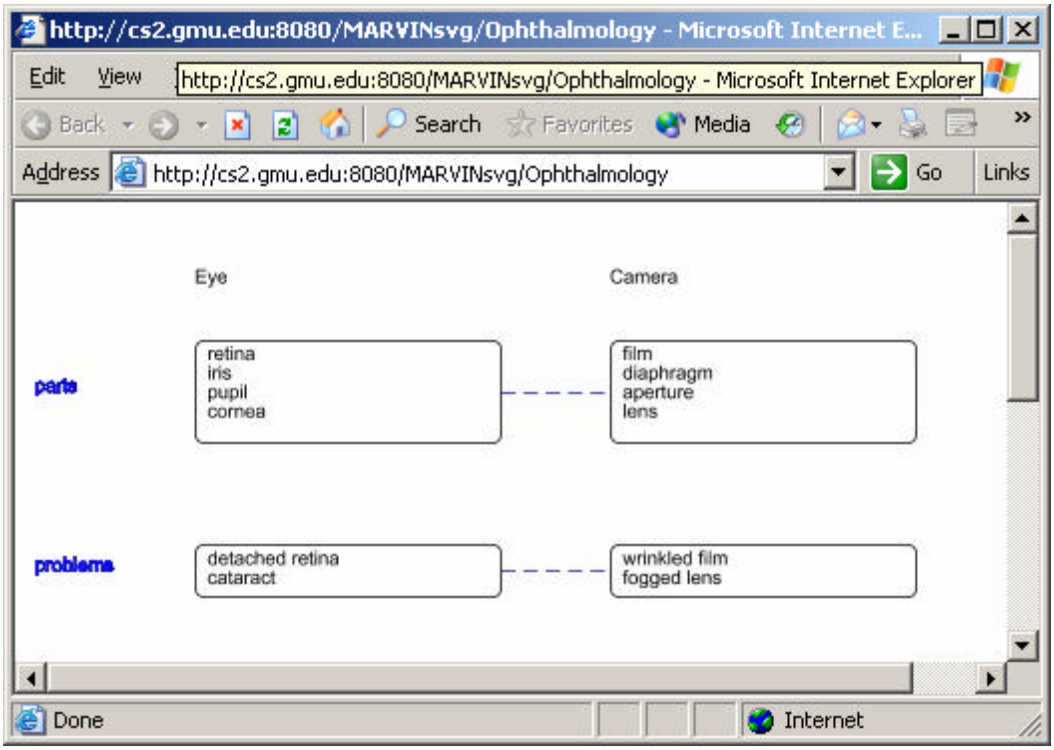
XML-Spy, http://www.xmlspy.com/, March 2002

Appendix

Example Analogies

Ophthalmology: The eye is like a camera; cataracts are like a fogged lens, a detached retina is like wrinkled film.



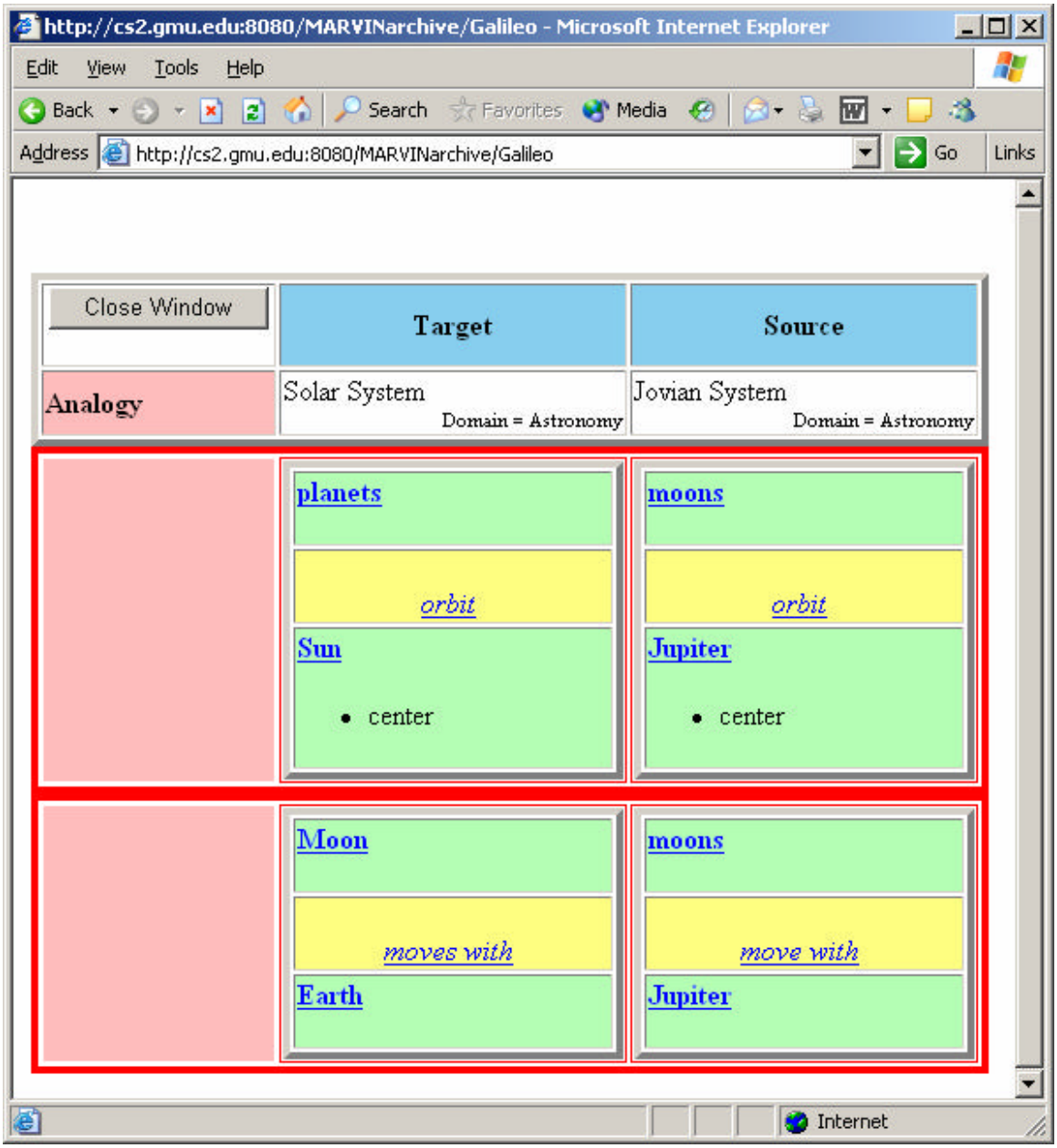


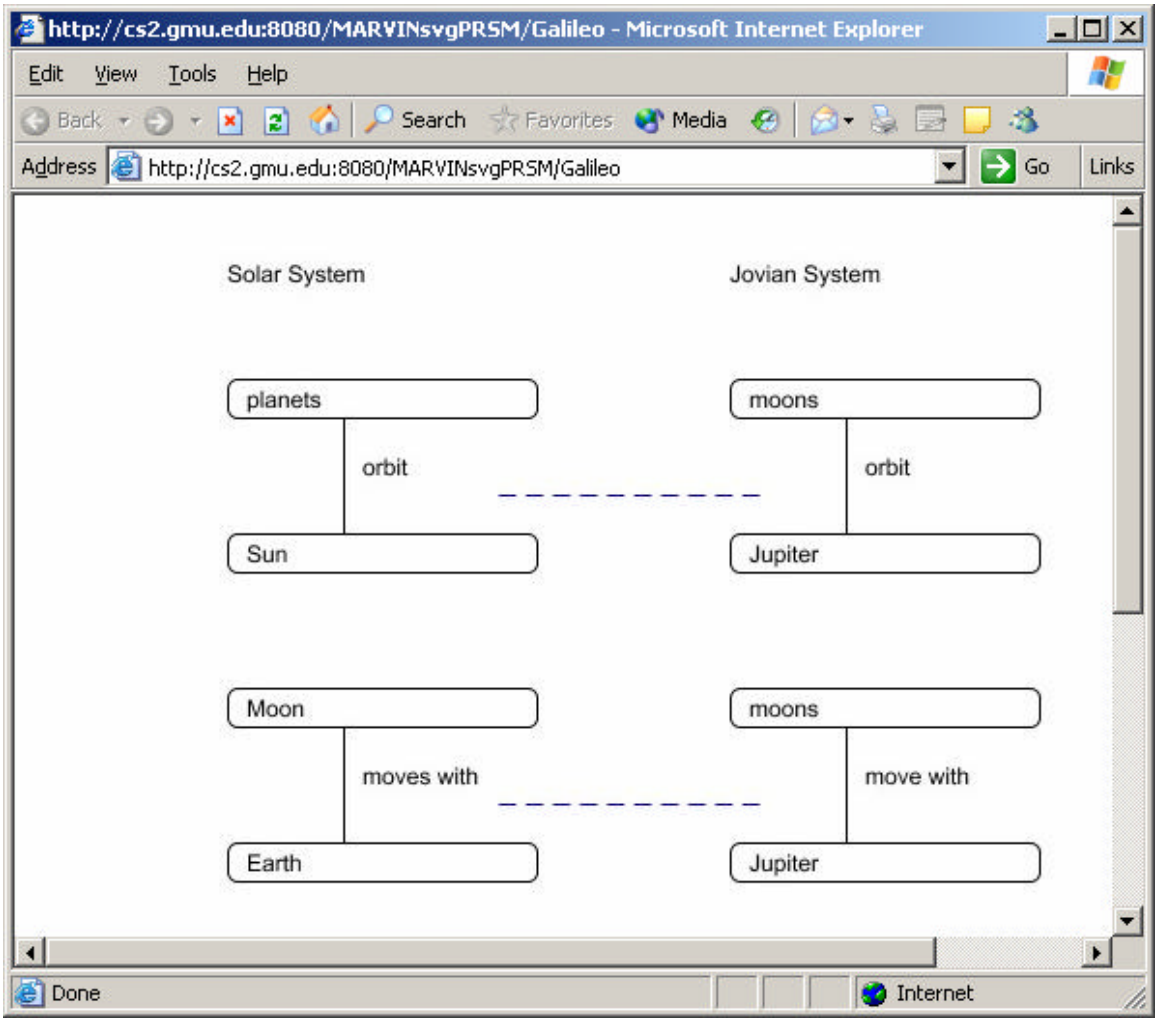
Markup for Ophthlmology.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Analogy SYSTEM
"http://cs.gmu.edu/~hfoxwell/MARVIN/Docs/AnalogyExpression.dtd"[
]>
<Analogy>
<TargetDescription Domain="Biology">Eye</TargetDescription>
<SourceDescription Domain="General">Camera</SourceDescription>
<Map>
<ConceptSetMap MapLabel="parts">
<TargetConceptSet>
<ConceptDescription><Concept>cornea</Concept></ConceptDescription>
<ConceptDescription><Concept>pupil</Concept></ConceptDescription>
<ConceptDescription><Concept>iris</Concept></ConceptDescription>
<ConceptDescription><Concept>retina</Concept></ConceptDescription>
</TargetConceptSet>
<SourceConceptSet>
<ConceptDescription><Concept>lens</Concept></ConceptDescription>
<ConceptDescription><Concept>aperture</Concept></ConceptDescription>
<ConceptDescription><Concept>diaphragm</Concept></ConceptDescription>
<ConceptDescription><Concept>film</Concept></ConceptDescription>
</SourceConceptSet>
</ConceptSetMap>
</Map>
<Map>
```

```
<ConceptSetMap MapLabel="problems">
<TargetConceptSet>
<ConceptDescription><Concept>cataract</Concept></ConceptDescription>
<ConceptDescription><Concept>detached
retina</Concept></ConceptDescription>
</TargetConceptSet>
<SourceConceptSet>
<ConceptDescription><Concept>fogged lens</Concept></ConceptDescription>
<ConceptDescription><Concept>wrinkled
film</Concept></ConceptDescription>
</SourceConceptSet>
</ConceptSetMap>
</Map>
</Analogy>
```

Galileo: Galileo's observation of Jupiter's moons strengthened his belief in the Copernican theory of the solar system.



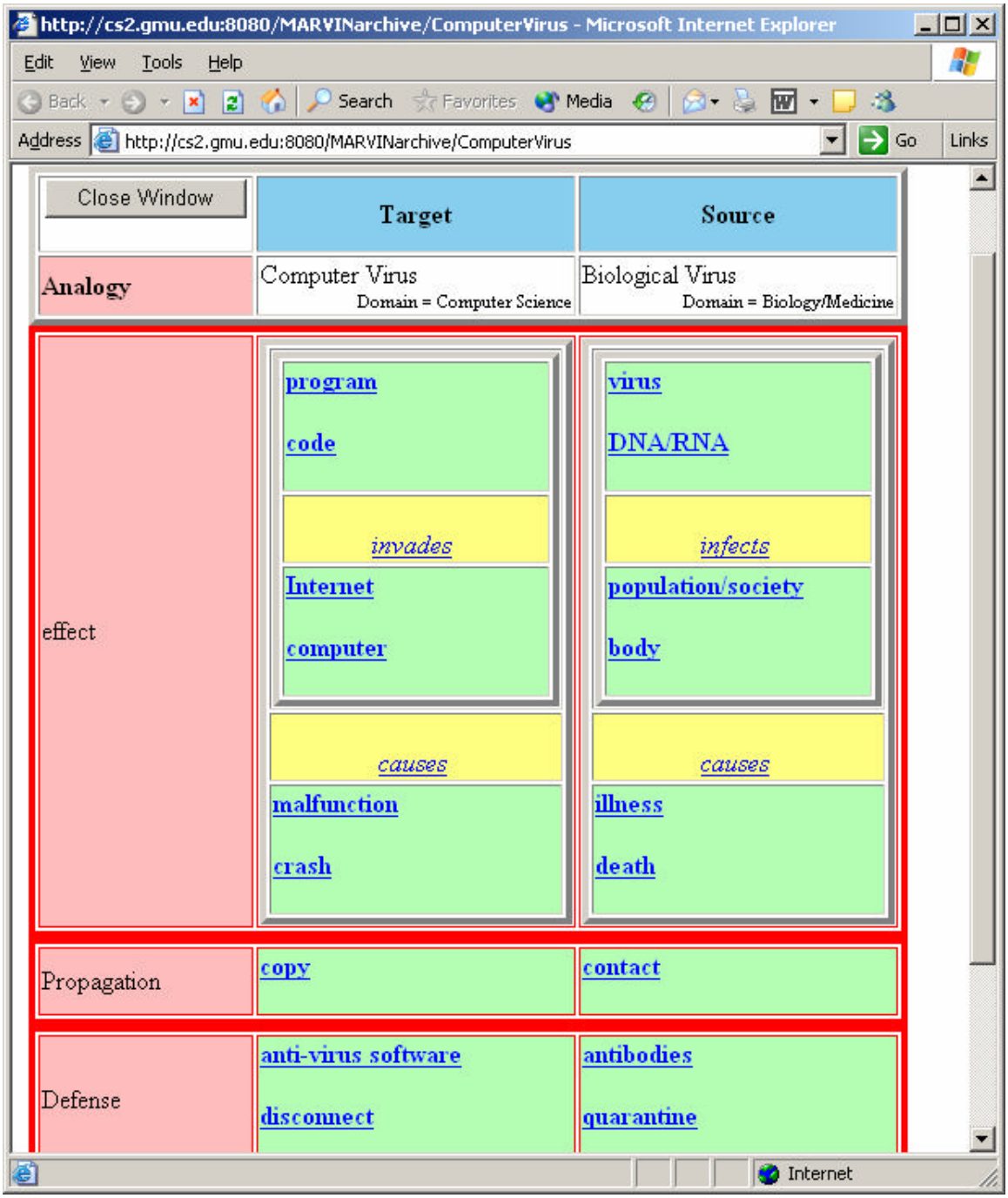


Markup for Galileo.xml

```
 <?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Analogy SYSTEM
"http://cs.gmu.edu/~hfoxwell/MARVIN/Docs/AnalogyExpression.dtd"[
]>
<Analogy>
<TargetDescription Domain="Astronomy">Solar System</TargetDescription>
<SourceDescription Domain="Astronomy">Jovian System</SourceDescription>
<Map>
<PrimaryRelationStructureMap>
<TargetPrimaryRelationStructure>
<ConceptSet>
<ConceptDescription><Concept>planets</Concept></ConceptDescription>
</ConceptSet>
<Relation>orbit</Relation>
<ConceptSet>
```

```
<ConceptDescription><Concept>Sun</Concept><Property>center</Property></
ConceptDescription>
</ConceptSet>
</TargetPrimaryRelationStructure>
<SourcePrimaryRelationStructure>
<ConceptSet>
<ConceptDescription><Concept>moons</Concept></ConceptDescription>
</ConceptSet>
<Relation>orbit</Relation>
<ConceptSet>
<ConceptDescription><Concept>Jupiter</Concept><Property>center</Propert
y></ConceptDescription>
</ConceptSet>
</SourcePrimaryRelationStructure>
</PrimaryRelationStructureMap>
</Map>
<Map>
<PrimaryRelationStructureMap>
<TargetPrimaryRelationStructure>
<ConceptSet>
<ConceptDescription><Concept>Moon</Concept></ConceptDescription>
</ConceptSet>
<Relation>moves with</Relation>
<ConceptSet>
<ConceptDescription><Concept>Earth</Concept></ConceptDescription>
</ConceptSet>
</TargetPrimaryRelationStructure>
<SourcePrimaryRelationStructure>
<ConceptSet>
<ConceptDescription><Concept>moons</Concept></ConceptDescription>
</ConceptSet>
<Relation>move with</Relation>
<ConceptSet>
<ConceptDescription><Concept>Jupiter</Concept></ConceptDescription>
</ConceptSet>
</SourcePrimaryRelationStructure>
</PrimaryRelationStructureMap>
</Map>
</Analogy>
```

Computer Virus, Destructive computer programs that spread among computers behave very much like biological viruses.

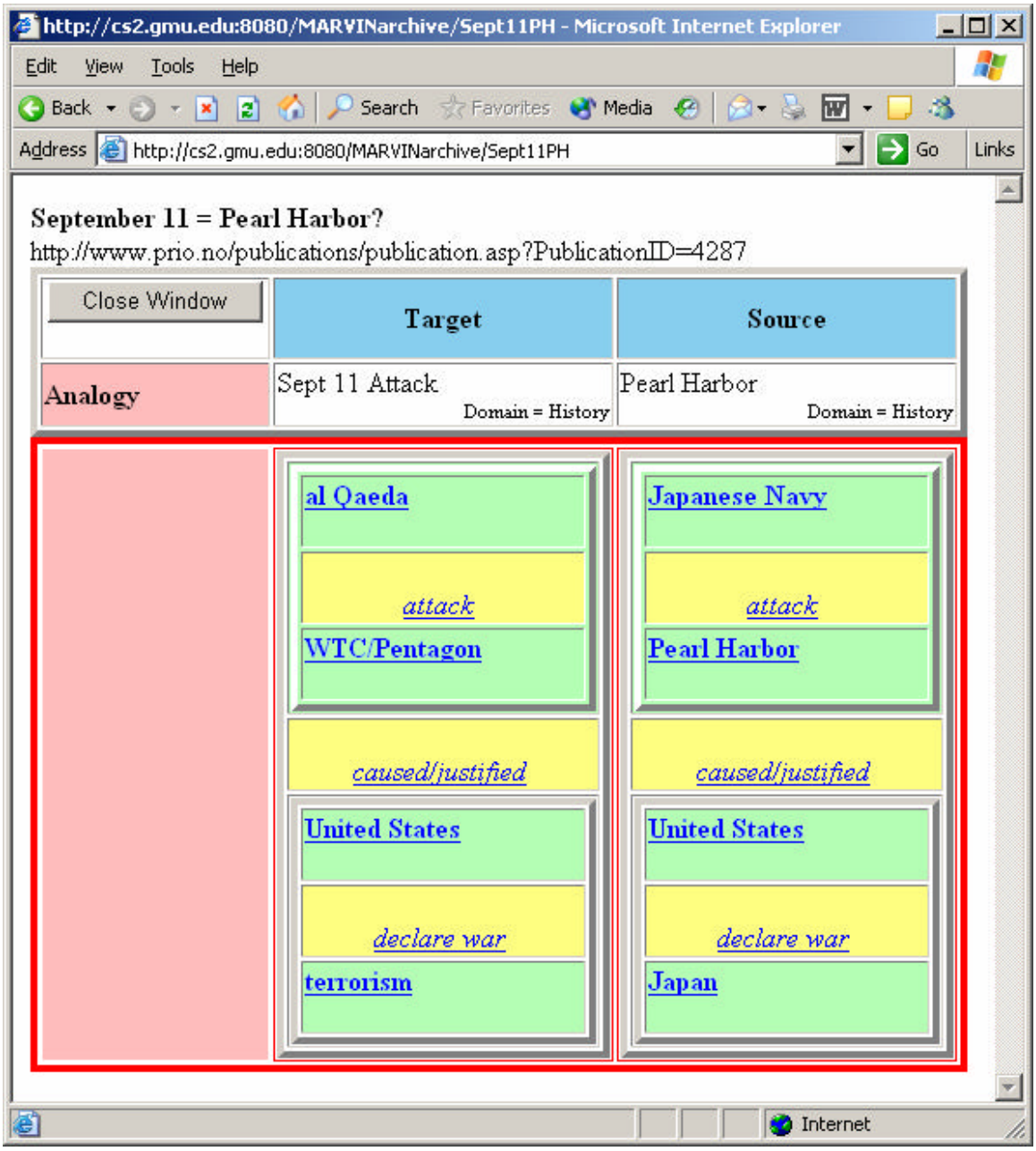


Markup for ComputerVirus.xml


```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Analogy SYSTEM
"http://cs.gmu.edu/~hfoxwell/MARVIN/Docs/AnalogyExpression.dtd"[
]>
<Analogy>
<TargetDescription Domain="Computer Science">Computer
Virus</TargetDescription>
<SourceDescription Domain="Biology/Medicine">Biological
Virus</SourceDescription>
<Map>
<RelationToConceptStructureMap MapLabel="effect">
<TargetRelationToConceptStructure>
<PrimaryRelationStructure>
<ConceptSet>
<ConceptDescription><Concept>program</Concept></ConceptDescription>
<ConceptDescription><Concept>code</Concept></ConceptDescription>
</ConceptSet>
<Relation>invades</Relation>
<ConceptSet>
<ConceptDescription><Concept>Internet</Concept></ConceptDescription>
<ConceptDescription><Concept>computer</Concept></ConceptDescription>
</ConceptSet>
</PrimaryRelationStructure>
<Relation>causes</Relation>
<ConceptSet>
<ConceptDescription><Concept>malfunction</Concept></ConceptDescription>
<ConceptDescription><Concept>crash</Concept></ConceptDescription>
</ConceptSet>
</TargetRelationToConceptStructure>
<SourceRelationToConceptStructure>
<PrimaryRelationStructure>
<ConceptSet>
<ConceptDescription><Concept>virus</Concept></ConceptDescription>
<ConceptDescription><Concept>DNA/RNA</Concept></ConceptDescription>
</ConceptSet>
<Relation>infects</Relation>
<ConceptSet>
<ConceptDescription><Concept>population/society</Concept></ConceptDescr
iption>
<ConceptDescription><Concept>body</Concept></ConceptDescription>
</ConceptSet>
</PrimaryRelationStructure>
<Relation>causes</Relation>
<ConceptSet>
<ConceptDescription><Concept>illness</Concept></ConceptDescription>
<ConceptDescription><Concept>death</Concept></ConceptDescription>
</ConceptSet>
</SourceRelationToConceptStructure>
</RelationToConceptStructureMap>
</Map>
<Map>
```

```
<ConceptSetMap MapLabel="Propagation">
<TargetConceptSet>
<ConceptDescription><Concept>copy</Concept></ConceptDescription>
</TargetConceptSet>
<SourceConceptSet>
<ConceptDescription><Concept>contact</Concept></ConceptDescription>
</SourceConceptSet>
</ConceptSetMap>
</Map>
<Map>
<ConceptSetMap MapLabel="Defense">
<TargetConceptSet>
<ConceptDescription><Concept>anti-virus
software</Concept></ConceptDescription>
<ConceptDescription><Concept>disconnect</Concept></ConceptDescription>
</TargetConceptSet>
<SourceConceptSet>
<ConceptDescription><Concept>antibodies</Concept></ConceptDescription>
<ConceptDescription><Concept>quarantine</Concept></ConceptDescription>
</SourceConceptSet>
</ConceptSetMap>
</Map>
<Map>
<ConceptSetMap MapLabel="change">
<TargetConceptSet>
<ConceptDescription><Concept>modification</Concept></ConceptDescription
>
</TargetConceptSet>
<SourceConceptSet>
<ConceptDescription><Concept>mutation</Concept></ConceptDescription>
</SourceConceptSet>
</ConceptSetMap>
</Map>
<Map>
<ConceptSetMap MapLabel="Specificity">
<TargetConceptSet>
<ConceptDescription><Concept>processor/OS</Concept></ConceptDescription
>
</TargetConceptSet>
<SourceConceptSet>
<ConceptDescription><Concept>cell type</Concept></ConceptDescription>
</SourceConceptSet>
</ConceptSetMap>
</Map>
</Analogy>
```

Historical Analogy: Sept 11: The September 11, 2001 attacks on the World Trade Center and Pentagon has been compared to the Japanese Navy's attack on Pearl Harbor.



Markup for Sept11PH.xml


```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE Analogy SYSTEM
"http://cs.gmu.edu/~hfoxwell/MARVIN/Docs/AnalogyExpression.dtd"[]>
<Analogy AnalogyName="September 11 = Pearl Harbor?"
Reference="http://www.prio.no/publications/publication.asp?PublicationI
D=4287">
<TargetDescription Domain="History">Sept 11 Attack</TargetDescription>
<SourceDescription Domain="History">Pearl Harbor</SourceDescription>
<Map>
<RelationToRelationStructureMap>
<TargetRelationToRelationStructure>
<PrimaryRelationStructure>
<ConceptSet>
<ConceptDescription><Concept>al Qaeda</Concept></ConceptDescription>
</ConceptSet>
<Relation>attack</Relation>
<ConceptSet>
<ConceptDescription><Concept>WTC/Pentagon</Concept></ConceptDescription
>
</ConceptSet>
</PrimaryRelationStructure>
<Relation>caused/justified</Relation>
<PrimaryRelationStructure>
<ConceptSet>
<ConceptDescription><Concept>United
States</Concept></ConceptDescription>
</ConceptSet>
<Relation>declare war</Relation>
<ConceptSet>
<ConceptDescription><Concept>terrorism</Concept></ConceptDescription>
</ConceptSet>
</PrimaryRelationStructure>
</TargetRelationToRelationStructure>
<SourceRelationToRelationStructure>
<PrimaryRelationStructure>
<ConceptSet>
<ConceptDescription><Concept>Japanese
Navy</Concept></ConceptDescription>
</ConceptSet>
<Relation>attack</Relation>
<ConceptSet>
<ConceptDescription><Concept>Pearl
Harbor</Concept></ConceptDescription>
</ConceptSet>
</PrimaryRelationStructure>
<Relation>caused/justified</Relation>
<PrimaryRelationStructure>
<ConceptSet>
<ConceptDescription><Concept>United
States</Concept></ConceptDescription>
</ConceptSet>
```

```
<Relation>declare war</Relation>
<ConceptSet>
<ConceptDescription><Concept>Japan</Concept></ConceptDescription>
</ConceptSet>
</PrimaryRelationStructure>
</SourceRelationToRelationStructure>
</RelationToRelationStructureMap>
</Map>
</Analogy>
```

XSL Stylesheet for Tabular Visualization

```
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="html" indent="yes"/>

<xsl:template match="/">
<html>
<head>
<B><xsl:apply-templates select="Analogy/@AnalogyName"/></B><br/>
   <xsl:apply-templates select="Analogy/@Reference"/><br/>
</head>
<body>
    <xsl:apply-templates/>
</body>
</html>
</xsl:template>

<xsl:template match="Analogy">
<table border="4">
<tr><th width="128" style="font-size:70%">
<form><input type="button" value="Close Window"
onClick="window.close()"/></form>
</th><th bgcolor="#87CEEE" width="192">Target</th><th bgcolor="#87CEEE"
width="192">Source</th></tr>
   <tr>
   <td width="128" bgcolor="#FFBCBC"><B>Analogy</B></td>
   <td width="192"><xsl:apply-templates
select="TargetDescription"/><br/>
      <div align="right" style="font-size:70%">
       Domain = <xsl:apply-templates
select="TargetDescription/@Domain"/></div></td>
   <td width="192"><xsl:apply-templates
select="SourceDescription"/><br/>
      <div align="right" style="font-size:70%">
       Domain = <xsl:apply-templates
select="SourceDescription/@Domain"/></div></td>
   </tr>
</table>

<xsl:apply-templates select="Map"/>
</xsl:template>

<xsl:template match="Map">
<xsl:apply-templates select="ConceptSetMap"/>
<xsl:apply-templates select="PrimaryRelationStructureMap"/>
<xsl:apply-templates select="ConceptToRelationStructureMap"/>
<xsl:apply-templates select="RelationToConceptStructureMap"/>
<xsl:apply-templates select="RelationToRelationStructureMap"/>
</xsl:template>

<xsl:template match="ConceptSetMap">
<table border="4" bordercolor="red">
```

```
<tr><td width="128" bgcolor="#FFBCBC"><xsl:apply-templates
select="@MapLabel"/></td>
<td width="192" bgcolor="#B4FFB4"><xsl:apply-templates
select="TargetConceptSet"/></td>
<td width="192" bgcolor="#B4FFB4"><xsl:apply-templates
select="SourceConceptSet"/></td>
</tr></table>
</xsl:template>

<xsl:template match="TargetConceptSet">
<xsl:apply-templates select="ConceptDescription"/>
</xsl:template>

<xsl:template match="SourceConceptSet">
<xsl:apply-templates select="ConceptDescription"/>
</xsl:template>

<xsl:template match="ConceptDescription">
<xsl:apply-templates select="Concept"/>
<ul><xsl:apply-templates select="Property"/></ul>
</xsl:template>

<xsl:template match="Concept">
<B><A TARGET="new2">
    <xsl:attribute name="HREF">

<xsl:text>http://meganesia.int.gu.edu.au/~phmartin/WebKB/bin/termDispla
y.cgi?term=</xsl:text>
      <xsl:apply-templates/>
    </xsl:attribute>
    <xsl:apply-templates/>
</A></B><p/>
</xsl:template>

<xsl:template match="Property">
<li><xsl:apply-templates/></li>
</xsl:template>

<xsl:template match="PrimaryRelationStructureMap">
<table border="4" bordercolor="red">
<tr><td width="128" bgcolor="#FFBCBC"><xsl:apply-templates
select="@MapLabel"/></td>
<td width="192"><xsl:apply-templates
select="TargetPrimaryRelationStructure"/></td>
<td width="192"><xsl:apply-templates
select="SourcePrimaryRelationStructure"/></td>
</tr></table>
</xsl:template>

<xsl:template match="TargetPrimaryRelationStructure">
<table border="4" width="100%">
<tr><td bgcolor="#B4FFB4">
<xsl:apply-templates select="ConceptSet[1]"/>
</td></tr>
```

```
<tr><td align="center" bgcolor="#FFFF80">
<xsl:apply-templates select="Relation"/>
</td></tr>
<tr><td bgcolor="#B4FFB4">
<xsl:apply-templates select="ConceptSet[2]"/>
</td></tr>
</table>
</xsl:template>

<xsl:template match="SourcePrimaryRelationStructure">
<table border="4" width="100%">
<tr><td bgcolor="#B4FFB4">
<xsl:apply-templates select="ConceptSet[1]"/>
</td></tr>
<tr><td align="center" bgcolor="#FFFF80">
<xsl:apply-templates select="Relation"/>
</td></tr>
<tr><td bgcolor="#B4FFB4">
<xsl:apply-templates select="ConceptSet[2]"/>
</td></tr>
</table>
</xsl:template>

<xsl:template match="ConceptSet">
<xsl:apply-templates select="ConceptDescription"/>
</xsl:template>

<xsl:template match="Relation">
<br/><i>
<A TARGET="new2">
    <xsl:attribute name="HREF">

<xsl:text>http://meganesia.int.gu.edu.au/~phmartin/WebKB/bin/termDispla
y.cgi?term=</xsl:text>
      <xsl:apply-templates/>
    </xsl:attribute>
    <xsl:apply-templates/>
</A>
</i>
</xsl:template>

<xsl:template match="ConceptToRelationStructureMap">
<table border="4" bordercolor="red">
<tr><td width="128" bgcolor="#FFBCBC"><xsl:apply-templates
select="@MapLabel"/></td>
<td width="192"><xsl:apply-templates
select="TargetConceptToRelationStructure"/></td>
<td width="192"><xsl:apply-templates
select="SourceConceptToRelationStructure"/></td>
</tr></table>
</xsl:template>

<xsl:template match="TargetConceptToRelationStructure">
<table border="4" width="100%">
```

```
<tr><td bgcolor="#B4FFB4">
<xsl:apply-templates select="ConceptSet"/>
</td></tr>
<tr><td align="center" bgcolor="#FFFF80">
<xsl:apply-templates select="Relation"/>
</td></tr>
<tr><td>
<xsl:apply-templates select="PrimaryRelationStructure"/>
</td></tr>
</table>
</xsl:template>

<xsl:template match="SourceConceptToRelationStructure">
<table border="4" width="100%">
<tr><td bgcolor="#B4FFB4">
<xsl:apply-templates select="ConceptSet"/>
</td></tr>
<tr><td align="center" bgcolor="#FFFF80">
<xsl:apply-templates select="Relation"/>
</td></tr>
<tr><td>
<xsl:apply-templates select="PrimaryRelationStructure"/>
</td></tr>
</table>
</xsl:template>

<xsl:template match="RelationToConceptStructureMap">
<table border="4" bordercolor="red">
<tr><td width="128" bgcolor="#FFBCBC"><xsl:apply-templates
select="@MapLabel"/></td>
<td width="192"><xsl:apply-templates
select="TargetRelationToConceptStructure"/></td>
<td width="192"><xsl:apply-templates
select="SourceRelationToConceptStructure"/></td>
</tr></table>
</xsl:template>

<xsl:template match="TargetRelationToConceptStructure">
<table border="4" width="100%">
<tr><td>
<xsl:apply-templates select="PrimaryRelationStructure"/>
</td></tr>
<tr><td align="center" bgcolor="#FFFF80">
<xsl:apply-templates select="Relation"/>
</td></tr>
<tr><td bgcolor="#B4FFB4">
<xsl:apply-templates select="ConceptSet"/>
</td></tr>
</table>
</xsl:template>

<xsl:template match="SourceRelationToConceptStructure">
<table border="4" width="100%">
<tr><td>
```

```
<xsl:apply-templates select="PrimaryRelationStructure"/>
</td></tr>
<tr><td align="center" bgcolor="#FFFF80">
<xsl:apply-templates select="Relation"/>
</td></tr>
<tr><td bgcolor="#B4FFB4">
<xsl:apply-templates select="ConceptSet"/>
</td></tr>
</table>
</xsl:template>

<xsl:template match="RelationToRelationStructureMap">
<table border="4" bordercolor="red">
<tr><td width="128" bgcolor="#FFBCBC"><xsl:apply-templates
select="@MapLabel"/></td>
<td width="192"><xsl:apply-templates
select="TargetRelationToRelationStructure"/></td>
<td width="192"><xsl:apply-templates
select="SourceRelationToRelationStructure"/></td>
</tr></table>
</xsl:template>

<xsl:template match="TargetRelationToRelationStructure">
<table border="4" width="100%">
<tr><td bgcolor="#B4FFB4">
<xsl:apply-templates select="PrimaryRelationStructure[1]"/>
</td></tr>
<tr><td align="center" bgcolor="#FFFF80">
<xsl:apply-templates select="Relation"/>
</td></tr>
<tr><td>
<xsl:apply-templates select="PrimaryRelationStructure[2]"/>
</td></tr>
</table>
</xsl:template>

<xsl:template match="SourceRelationToRelationStructure">
<table border="4" width="100%">
<tr><td bgcolor="#B4FFB4">
<xsl:apply-templates select="PrimaryRelationStructure[1]"/>
</td></tr>
<tr><td align="center" bgcolor="#FFFF80">
<xsl:apply-templates select="Relation"/>
</td></tr>
<tr><td>
<xsl:apply-templates select="PrimaryRelationStructure[2]"/>
</td></tr>
</table>
</xsl:template>

<xsl:template match="PrimaryRelationStructure">
<table border="4" width="100%">
<tr><td bgcolor="#B4FFB4">
<xsl:apply-templates select="ConceptSet[1]"/>
```

```
</td></tr>
<tr><td align="center" bgcolor="#FFFF80">
<xsl:apply-templates select="Relation"/>
</td></tr>
<tr><td bgcolor="#B4FFB4">
<xsl:apply-templates select="ConceptSet[2]"/>
</td></tr>
</table>
</xsl:template>

</xsl:stylesheet>
```

XSL Stylesheet for Graphical Visualization

```
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"

xmlns:xlink="http://www.w3.org/2000/xlink/namespace">
<xsl:output method="xml" indent="yes"
     doctype-public="-//W3C//DTD SVG 1.0//EN"
     doctype-system="http://www.w3.org/TR/2001/REC-
SVG=20010904/DTD/svg1.0.dtd"/>

<xsl:variable name="TargX" select="100"/>
<xsl:variable name="TargY" select=" 30"/>
<xsl:variable name="SrcX"  select="360"/>
<xsl:variable name="SrcY"  select=" 30"/>

<xsl:variable name="MapSize" select="160"/>

<xsl:variable name="BoxH"  select=" 20"/>
<xsl:variable name="BoxW"  select="160"/>

<xsl:template match="/">
<svg width="640" height="640" viewBox="0 0 640 640">

<xsl:apply-templates/>

</svg>
</xsl:template>

<xsl:template match="Analogy">
<xsl:apply-templates select="TargetDescription"/>
<xsl:apply-templates select="SourceDescription"/>
<xsl:apply-templates select="Map"/>
</xsl:template>

<xsl:template match="TargetDescription">
<text x="{$TargX}" y="{$TargY}"><xsl:apply-templates/></text>
</xsl:template>

<xsl:template match="SourceDescription">
<text x="{$SrcX}" y="{$SrcY}"><xsl:apply-templates/></text>
</xsl:template>

<xsl:template match="Map">
<xsl:apply-templates select="PrimaryRelationStructureMap"/>
</xsl:template>

<xsl:template match="PrimaryRelationStructureMap">

<xsl:apply-templates select="TargetPrimaryRelationStructure"/>
<xsl:apply-templates select="SourcePrimaryRelationStructure"/>
</xsl:template>
```

```
<xsl:template match="TargetPrimaryRelationStructure">
<xsl:variable name="Yoffset"
select="(count(preceding::PrimaryRelationStructureMap))"/>
<text x="20" y="{140 + $MapSize * $Yoffset}" style="stroke:
blue"><xsl:apply-templates select="../@MapLabel"/></text>

<rect x="100" y="{ 80 + $MapSize * $Yoffset}" rx="5" width="{$BoxW}"
height="{$BoxH}" style="fill: none; stroke: black;"/>
<rect x="100" y="{160 + $MapSize * $Yoffset}" rx="5" width="{$BoxW}"
height="{$BoxH}" style="fill: none; stroke: black;"/>

<line x1="160" y1="{100 + $MapSize * $Yoffset}" x2="160" y2="{160 +
$MapSize * $Yoffset}" style="stroke:black;"/>
<line x1="240" y1="{140 + $MapSize * $Yoffset}" x2="380" y2="{140 +
$MapSize * $Yoffset}" style="stroke: blue; stroke-dasharray: 9 5"/>

<text x="110" y="{95 + $MapSize * $Yoffset}"><xsl:apply-templates
select="ConceptSet[1]/ConceptDescription/Concept"/></text>
<text x="170" y="{130 + $MapSize * $Yoffset}"><xsl:apply-templates
select="Relation"/></text>
<text x="110" y="{175 + $MapSize * $Yoffset}"><xsl:apply-templates
select="ConceptSet[2]/ConceptDescription/Concept"/></text>
</xsl:template>

<xsl:template match="SourcePrimaryRelationStructure">
<xsl:variable name="Yoffset"
select="(count(preceding::PrimaryRelationStructureMap))"/>

<rect x="360" y="{ 80 + $MapSize * $Yoffset}" rx="5" width="{$BoxW}"
height="{$BoxH}" style="fill: none; stroke: black;"/>
<rect x="360" y="{160 + $MapSize * $Yoffset}" rx="5" width="{$BoxW}"
height="{$BoxH}" style="fill: none; stroke: black;"/>

<line x1="420" y1="{100+ $MapSize * $Yoffset}" x2="420" y2="{160+
$MapSize * $Yoffset}" style="stroke:black;"/>

<text x="370" y="{95 + $MapSize * $Yoffset}"><xsl:apply-templates
select="ConceptSet[1]/ConceptDescription/Concept"/></text>
<text x="430" y="{130 + $MapSize * $Yoffset}"><xsl:apply-templates
select="Relation"/></text>
<text x="370" y="{175 + $MapSize * $Yoffset}"><xsl:apply-templates
select="ConceptSet[2]/ConceptDescription/Concept"/></text>
</xsl:template>

</xsl:stylesheet>
```

CURRICULUM VITAE

Harry J. Foxwell was born in Brooklyn, NY on August 15, 1947; he is an American citizen. His undergraduate studies were interrupted by a tour of duty in the US Army during 1968-1969 as an Infantry Platoon Sergeant in Viet Nam. He completed his Bachelor of Arts in Mathematics at Franklin & Marshall College in 1973, and completed his Master of Science in Applied Statistics at Villanova University in 1978. He has worked as a high school mathematics teacher and as a survey statistician; he is currently employed in McLean, Virginia as a System Engineer for Sun Microsystems. He has been a member of the Association for Computing Machinery (ACM) since 1997.

He has published:

Foxwell, H., and Menascé, D., *Web-based Representation and Visualization of Analogies*, Proceedings of the IADIS International Conference, WWW/Internet 2002, Lisbon, Portugal, http://cs.gmu.edu/~hfoxwell/MARVIN/Papers/iadispaper.pdf, November 2002

Foxwell, H., *A Framework for Evaluating Mobile Agent Systems*, JavaOne'99 Software Developer's Conference, http://mason.gmu.edu/~hfoxwell/J1MABOF/, June 1999

Foxwell, H., and Menascé, D., *Prefetching Results of Web Searches*, CMG98: Computer Measurement Group's 1998 International Conference, December 1998

Foxwell, H., *Java for Linux*, Java Developer's Journal, Vol. 4, Issue 12, December 1999