

University of Central Florida

STARS

Electronic Theses and Dissertations

2004

A Software-based Knowledge Management System Using Narrative Texts

Thomas Rudy McDaniel

University of Central Florida, rudy@ucf.edu



Part of the [Technical and Professional Writing Commons](#)

Find similar works at: <https://stars.library.ucf.edu/etd>

University of Central Florida Libraries <http://library.ucf.edu>

This Doctoral Dissertation (Open Access) is brought to you for free and open access by STARS. It has been accepted for inclusion in Electronic Theses and Dissertations by an authorized administrator of STARS. For more information, please contact STARS@ucf.edu.

STARS Citation

McDaniel, Thomas Rudy, "A Software-based Knowledge Management System Using Narrative Texts" (2004). *Electronic Theses and Dissertations*. 106.

<https://stars.library.ucf.edu/etd/106>

A SOFTWARE-BASED KNOWLEDGE MANAGEMENT SYSTEM USING NARRATIVE TEXTS

by

THOMAS RUDY MCDANIEL
M.A. University of Central Florida, 2001
B.S. University of Central Florida, 2003
B.S. University of Central Florida, 1999

A dissertation submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Texts and Technology
in the Department of English
in the College of Arts and Sciences
at the University of Central Florida
Orlando, Florida

Spring Term
2004

Major Professor: Paul Dombrowski

© 2004 Thomas Rudy McDaniel

ABSTRACT

Technical and professional communicators have in recent research been challenged to make significant contributions to the field of knowledge management, and to learn or create the new technologies allowing them to do so. The purpose of this dissertation is to make such a combined theoretical and applied contribution from the context of the emerging discipline of Texts and Technology.

This dissertation explores the field of knowledge management (KM), particularly its relationship to the related study of artificial intelligence (AI), and then recommends a KM software application based on the principles of narratology and narrative information exchange. The focus of knowledge is shifted from the reductive approach of data and information to a holistic approach of meaning and the way people make sense of complex events as experiences expressed in stories. Such an analysis requires a discussion of the evolution of intelligent systems and narrative theory as well as an examination of existing computerized and non-computerized storytelling systems. After a thorough discussion of these issues, an original software program that is used to collect, analyze, and distribute thematic stories within any hierarchical organization is modeled, exemplified, and explained in detail.

ACKNOWLEDGEMENTS

There are many people who helped me build this dissertation into its current form. I would like to first thank my wife, Carole McDaniel, for spending many hours designing and creating the graphics for my online software application. Without her aesthetic expertise, the screenshots in Chapter Five would be nothing less than frightening. Secondly, I would like to thank my dissertation advisor, Dr. Paul Dombrowski, for patiently guiding me through this process. Thirdly, I need to acknowledge all the members of my committee, including Dr. Dan Jones, Dr. Karla Kitalong, and Mr. Erik Vick. Dr. Jones' comments and suggestions have vastly improved the style and readability of this dissertation, and Dr. Kitalong's enthusiastic support energized me and kept me writing steadily. Erik Vick's course *Autonomous Media*, which I enrolled in several summers ago, helped to inspire the topic of this dissertation.

I also would like to express my appreciation for Dr. Craig Saper, Dr. James Campbell, Ms. Benita Black and Ms. Ginny Herrington, who collectively represent the department's graduate team. They consistently do a wonderful job of keeping their students both happy and motivated. Finally, I would like to thank Dr. Dawn Trouard for her years of mentorship and encouragement.

TABLE OF CONTENTS

LIST OF TABLES.....	vii
LIST OF FIGURES	viii
LIST OF ABBREVIATIONS.....	x
LIST OF HYPERLINKS	xi
CHAPTER ONE: INTRODUCTION.....	1
CHAPTER TWO: THE EVOLUTION OF INTELLIGENT SYSTEMS	22
Background.....	22
A Trend towards Complexity.....	25
Abstraction and Intelligence	26
Calculating Machines.....	31
Automata.....	36
Barriers to Strong Artificial Intelligence	40
Boundary, Gender, Stereotype, and Psychoanalysis.....	50
Cyborgs and Androids	56
Applied Artificial Intelligence	62
Chatterbots and the Turing Test.....	70
Into the 21st Century.....	76
CHAPTER THREE: PROPERTIES OF A NARRATIVE SYSTEM.....	79
Background.....	79
Narratology and the Narrative System.....	81
Emergent Properties.....	93
Storing Knowledge	97
Life Experiences and the <i>SparkleTech</i> Story	108
Metaphor and Analogy	113
Scope and Domain	115
Computers and Hypertext	118
Intelligence Revisited.....	119
A New Domain for Technical Communicators	122
CHAPTER FOUR: EXAMPLES OF NARRATIVE SYSTEMS	128
Background.....	128
Oral and Written Storytelling.....	130
Computer-Based Storytelling Systems	136
TALE-SPIN	137
BRUTUS, MINSTREL, and MEXICA	141
Organizational and Institutional Storytelling Systems	147

Visual Storytelling	161
CHAPTER FIVE: MODELING <i>EDNA-E</i>	168
Background	168
Intelligence through Distributed Cognition	176
Software Overview	178
Planning and Design	180
Building the User Interface	187
Additional Design Considerations	191
Use Case Diagram	192
Sequence Diagram	198
The Story Administrator Session: Phase One	200
Administrative Homepage	200
Organization Panel	202
Group Administration Panel	205
Add Event Panel	206
Scripts Panel	208
The User Session: Phase One	216
The Registration Panel	216
Registration Results Panel	218
The Story Administrator Session: Phase Two	219
User Administration Panel	219
Group Administration Panel	220
The User Session: Phase Two	223
Login Panel	223
Add Story Panel	225
Add Story Panel	225
Analyzing Story Panel	229
Program Features	234
Design Limitations	235
Discussion	237
CHAPTER SIX: CONCLUSION	243
LIST OF REFERENCES	261

LIST OF TABLES

Table 1: <i>EDNA-E</i> Program and Documentation Links	180
Table 2: Intelligent Reasoning Models	248

LIST OF FIGURES

Figure 1: XML Representation of Dissertation	3
Figure 2: DTD for Dissertation XML File.....	5
Figure 3: XML with DTD Representation.....	6
Figure 4: XML Version of Dissertation Information.....	7
Figure 5: HTML Representation of Dissertation	8
Figure 6: HTML Version of Dissertation Information	10
Figure 7: The Curta Machine (The Curta Calculator Page online).....	35
Figure 8: <i>Turvy</i>	39
Figure 9: Lattice LIR Framework	43
Figure 10: An early poem by RACTER (Güzeldere and Franchi online)	62
Figure 11: ELIZA and PARRY Conversing (Güzeldere and Franchi online).....	73
Figure 12: Finite State Fabula Representation.....	87
Figure 13: Frames Model for Representing a Story.....	104
Figure 14: Story and NarrativePoem Classes	105
Figure 15: StoryFrame Inherited Class	107
Figure 16: Excerpt from Booker T. Washington's 1895 Cotton State Expedition Speech	133
Figure 17: Excerpt from Jesse Jackson's 1984 Campaign Speech.....	134
Figure 18: Excerpt from Reagan's Challenger Speech	135
Figure 19: A TALE-SPIN Story (Meehan 199-200)	138
Figure 20: TALE-SPIN Opening Menu (Meehan 204)	139
Figure 21: TALE-SPIN User Interface (Meehan 204-205)	140
Figure 22: A MINSTREL Story: Lancelot Story (Pérez y Pérez and Sharples 19).....	143
Figure 23: A BRUTUS Story: Simple Betrayal (Pérez y Pérez and Sharples 18).....	145
Figure 24: A MEXICA Story: The Lovers (Pérez y Pérez and Sharples 22)	147
Figure 25: Story Selection SQL Statement.....	155
Figure 26: Building the Monetary-Policy Story	161
Figure 27: Story Elements Unorganized (Eisner 10)	163
Figure 28: An Organized Story (Eisner 10).....	163
Figure 29: Comics Defined (McCloud 9)	164
Figure 30: <i>EDNA</i> Logo (Phase 1 RAD).....	183
Figure 31: <i>EDNA-E</i> User Interface (Phase 2 RAD).....	188
Figure 32: <i>EDNA-E</i> User Interface after Login	190
Figure 33: <i>EDNA-E</i> Use Case Diagram for Major Functions.....	193
Figure 34: User Functions.....	193
Figure 35: Story Administrator Functions	196

Figure 36: Sequence Diagram for <i>EDNA-E</i> Registration	199
Figure 37: Administrator Homepage	201
Figure 38: Add Organization Panel	203
Figure 39: Existing Organizations Panel	203
Figure 40: Adding Subdivision Message	204
Figure 41: Group Administration Panel	205
Figure 42: Events Panel	206
Figure 43: Add an Event Panel	207
Figure 44: Adding Events Panel	208
Figure 45: Scripts Panel	209
Figure 46: Add Script Step One	209
Figure 47: Add Script Step Two	211
Figure 48: Add Script Step Three	212
Figure 49: Add Script Step Four	213
Figure 50: Add Script Step Five	214
Figure 51: Script Output	215
Figure 52: <i>EDNA-E</i> Registration Panel	217
Figure 53: Registration Results	218
Figure 54: Enabling a User Account	220
Figure 55: Group Administration Panel	221
Figure 56: Edit Group 18 Panel	222
Figure 57: Edit Group 51 Panel	223
Figure 58: Login Panel	223
Figure 59: Login Error	224
Figure 60: Read Stories Panel	225
Figure 61: Add Story Panel	225
Figure 62: Invalid Permissions Error	226
Figure 63: Add a Story Step 1	226
Figure 64: Create a Story (Top Panel)	227
Figure 65: Create a Story (Bottom Panel)	228
Figure 66: Analyzing Story Top Panel	229
Figure 67: Analyze Story Middle Panel (Keywords)	230
Figure 68: Analyze Story Bottom Panel	232
Figure 69: “Writing.xml” File in XML Editor	233
Figure 70: Embodied/Embrained Knowledge Representations: Two Models	254
Figure 71: Freytag's Triangle (UIUC online)	258

LIST OF ABBREVIATIONS

AI: artificial intelligence
AL: artificial life
CSS: cascading style sheets
DTD: document type definition
EDNA-E.: event-driven narrative analysis engine
GOFAI: good old-fashioned artificial intelligence
HTML: hypertext markup language
KA: knowledge acquisition
KM: knowledge management
KR: knowledge representation
OO: object-oriented
OOP: object-oriented programming
PHP: hypertext preprocessor
QPM: quarterly projection model (Bank of Canada)
RAD: rapid application development model
RDF: resource descriptor framework
RP: represented participant
SAX: simple API for XML
SQL: structured query language
T&T: Texts and Technology
UCF: University of Central Florida
UML: unified modeling language
XML: eXtensible markup language

LIST OF HYPERLINKS

EDNA-E Homepage	http://www.textsandtech.org/~rudym/edna-e/
ENDA-E Documentation	http://www.textsandtech.org/~rudym/edna-e/documentation/

CHAPTER ONE: INTRODUCTION

“...while experts generally cannot access any information explaining their ability, they can usually assess the value or desirability of a situation easily and rapidly and recommend an appropriate action” – Hubert L. Dreyfus

The purpose of this dissertation is to describe and develop a narrative knowledge management application that is Semantic Web-enabled using eXtensible Markup Language (XML). A system will be created that is capable of describing and labeling themes and keywords within stories and then comparing those stories to appropriate indexes in the stories of other users. The stories will be labeled for content and meaning rather than just for structure and display, as is the case with hypertext markup language (HTML). Rather than relying on a fully independent system for classification, the software features story linking and connection functions that enable links between stories to be added, approved, or deleted by a computer administrator or information manager. Such an arrangement will allow for human supervision while eliminating the need for grueling editing and markup coding by a human data processor.

This opening preface begins with a short example which explains the XML markup language and compares this to HTML, which is similar in syntax but dissimilar in function and purpose. The Semantic Web will also be explained and described as the next evolutionary step in today's version of the World Wide Web. XML will be discussed in more detail in Chapter Five, but it is important to jump forward a little and describe the technology that will eventually

be used to implement the narrative system described in Chapter Three and improve upon previous story manipulation systems that are covered in detail in Chapter Four.

The overall structure of the dissertation is as follows: Chapter One contains a primer on XML and describes the basic layout of this document. Chapter Two develops a history of intelligent systems including computing technologies and artificial intelligence. Chapter Three explains the properties of narratology and what I will define as the “narrative system,” Chapter Four discusses applied examples of these types of systems, and finally Chapter Five models and implements an XML-based narrative system of my own design. The dissertation concludes with a brief analysis and discussion of this XML system in Chapter Six, including some final thoughts about knowledge management. The conclusion also contains some potential ideas for future research and suggested modifications to this software application. The focus of each chapter is also explained in more detail after these opening remarks on XML and knowledge management.

Before beginning a discussion concerning knowledge management, I will explain a modern technology that is often regarded as an indispensable tool for online knowledge management systems. This tool, which enables document creators to structure and organize both textual and multimedia information into a logical order of their own design, is the markup language known as XML. This language was designed to be a complement to HTML, to extend the capabilities of HTML rather than replace this language altogether.

XML is a “simple, very flexible text format derived from SGML” that was “originally designed to meet the challenges of large-scale electronic publishing” (W3C.org online). XML is similar to HTML in that both technologies are markup languages, but XML is intended to give segments of text meaning instead of just providing formatting instructions for them. “XML was

designed to describe data and to focus on what data is” while “HTML was designed to display data and to focus on how data looks” (W3Schools.com online). In this sense, XML can be used to structure information in some fashion determined by the XML’s author. For example, it is possible to mark up information describing this dissertation using XML as shown in Figure 1.

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<dissertation_file>
<dissertation>
<id>Spring04_001</id>
<author>Rudy McDaniel</author>
<title>A Software-Based Narrative Knowledge Management Model</title>
<director>Paul Dombrowski</director>
<committee_member>Dan Jones</committee_member>
<committee_member>Karla Kitalong</committee_member>
<committee_member>Erik Vick</committee_member>
</dissertation>
</dissertation_file>
```

Figure 1: XML Representation of Dissertation

The opening tag beginning with “<?xml version” indicates the XML specification being used and the character set being used in the text file, which is in this case the “ISO-8859-1 (Latin-1/West European) character set” (W3Schools.com online). From the continuing markup it should be evident that the <dissertation> tag begins this particular dissertation, and the </dissertation> tag will end the dissertation. At this point an XML processor (such as an XML-enabled software program or even a web browser) will know to stop searching for additional elements within that dissertation and instead be ready to scan in a new XML object, which may be another dissertation or something else altogether. While inside the opening <dissertation>

tag, though, the XML parser (reader) will scan for information describing that particular dissertation, which it will find is labeled as dissertation “Spring04_001” after scanning the next set of <id> tags. This next tag could also be combined as an attribute of the dissertation tag rather than as a separate child element, in which case it would look like <dissertation id=”Spring04_001”> . . . </dissertation> and in this case it would not have the <id> element listed at all. After scanning the id information, the XML parser (reader) can then gather additional information about this dissertation, such as the author, director, and committee members, and categorize this information accordingly in memory once the document has been parsed (read). This arrangement looks very similar to the structured nature of a relational database, and yet it needs no special software in order for this representation to be stored. Here both the information and the structure of that information are kept in a plain text file, which makes information exchange between other computer systems and mobile software programs, known as agents, much easier to accomplish.

The XML example above is “valid,” meaning it conforms to the rules of the XML 1.0 specification (which is very similar to the HTML 4 specification with some minor exceptions such as the insistence on all lowercase letters in tags, the requirement of closing tags on all elements, and the necessity of single and double quotation marks on all attribute tags) but it is not yet “well-formed.” A “well-formed” XML document “also conforms to the rules of a Document Type Definition (DTD)” which is intended to “define the legal building blocks of an XML document” and determine “the document structure with a list of legal elements” (W3Schools.com online). This definition allows the XML author to build his or her own set of restrictions that will be enforced by the DTD. Figure 2 provides a DTD for the XML file above.


```
<!DOCTYPE dissertation_file [  
<!ELEMENT dissertation (id,author,title,director,committee_member+)>  
<!ELEMENT id (#PCDATA)>  
<!ELEMENT author (#PCDATA)>  
<!ELEMENT title (#PCDATA)>  
<!ELEMENT director (#PCDATA)>  
<!ELEMENT committee_member (#PCDATA)>  
>]
```

Figure 2: DTD for Dissertation XML File

The “#PCDATA” tag inside each set of parentheses indicates that these elements will accept only parsed character data for their respective values, which is the text that will be inside the element tags and read by the browser or software program. The first <!ELEMENT declaration (followed by the name of that particular doctype declaration) lists in order the allowable elements for that document type, followed by commas. Note also the “+” at the end of the “committee_member” element on this same line, which means that at least one “committee_member” element must occur in the XML file but also that more than one element is acceptable. The DTD structure allows for many different conditions and element types to be specified in order to impose restrictions on an XML file. These conditions must then be enforced in order for a software program to accept that particular XML data file.

Now that the dissertation XML file is both valid (with proper syntax) and well-formed (with a DTD definition imposed), the final version of the file will appear as shown in Figure 3.

```

<?xml version="1.0" encoding="ISO-8859-1"?>

<!DOCTYPE dissertation_file [
  <!ELEMENT dissertation (id,author,title,director,committee_member+)>
  <!ELEMENT id (#PCDATA)>
  <!ELEMENT author (#PCDATA)>
  <!ELEMENT title (#PCDATA)>
  <!ELEMENT director (#PCDATA)>
  <!ELEMENT committee_member (#PCDATA)>
]>

<dissertation_file>
<dissertation>
<id>Spring04_001</id>
<author>Rudy McDaniel</author>
<title>A Software-Based Narrative Knowledge Management Model</title>
<director>Paul Dombrowski</director>
<committee_member>Dan Jones</committee_member>
<committee_member>Karla Kitalong</committee_member>
<committee_member>Erik Vick</committee_member>
</dissertation>
</dissertation_file>

```

Figure 3: XML with DTD Representation

This file is displayed in the Internet Explorer 6 web browser as shown in Figure 4. Note the “-” link on this page, which allows the element information for this dissertation object to be collapsed within the browser. Similarly, using the “+” link which appears in the compressed state allows this information to be expanded.

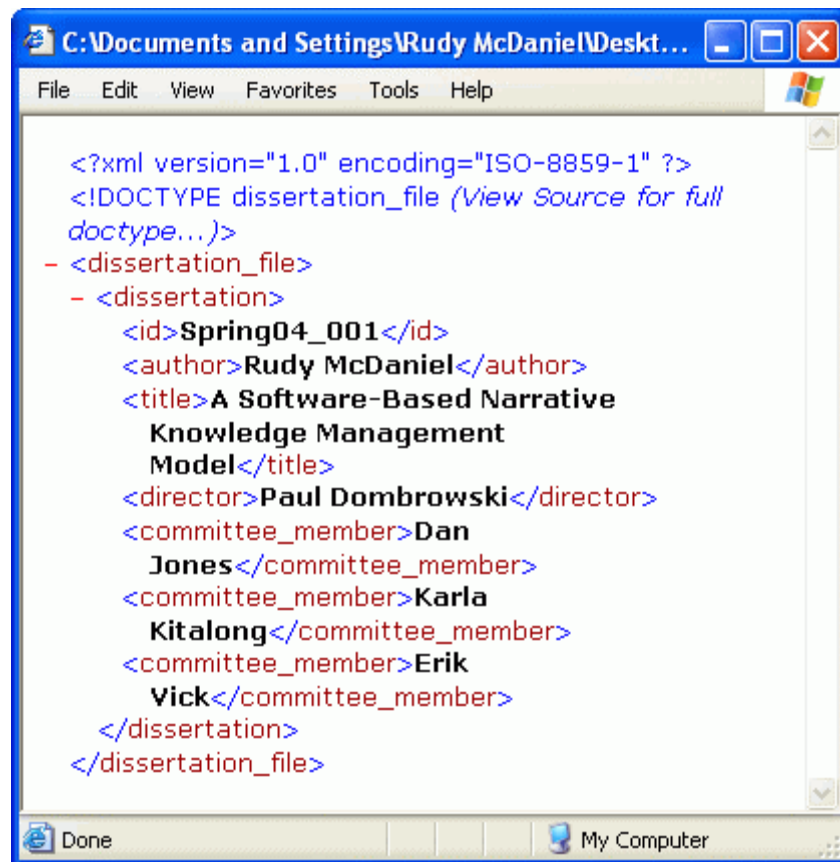


Figure 4: XML Version of Dissertation Information

The text inside this file is now meaningful in terms of the arbitrarily defined definitions on this file, and it is structured to follow a set of rules in the DTD pattern described above. In order to make this information useful, however, there needs to be some sort of software that can parse this data and do something productive with it. Perhaps for the dissertation information there might be an XML software agent that would scan through web pages listing various dissertations in progress, compiling listings with similar directors or titles (using the element tags that already exist) or even themes (which would necessitate the creation of new element tags to our existing model). These listings could then be sent to university research programs, corporate

sponsors, or even to a mass listing of authors who share common topics. This is, of course, just one hypothetical example of what is possible with XML-encoded information. The software that accompanies this dissertation, for instance, will be using XML to identify and encode particular thematic and scripted elements within a story. Multiple story elements can then be identified, classified, and compared in a standardized way.

To illustrate further the difference between XML and HTML, we can also represent this dissertation's description using HTML. This representation can be done in several different ways; Figure 5 shows one such example of this dissertation using standard HTML 4.

```
<!doctype html public "-//W3C//DTD HTML 4.0 Transitional//EN">

<html>
  <head>
    <title>Dissertation Spring04_001</title>
  </head>

  <body>
    <h1>Dissertation Spring04_001</h1>
    <h2>A Software-Based Narrative Knowledge Management Model</h2>
    <h3>Rudy McDaniel</h3>
    <p><b>Director:</b> Paul Dombrowski</p>
    <p>Committee Members:</p>
    <ul>
      <li>Dan Jones</li>
      <li>Karla Kitalong</li>
      <li>Erik Vick</li>
    </ul>
  </body>
</html>
```

Figure 5: HTML Representation of Dissertation

The markup shown in Figure 5 will be rendered in a web browser as shown in Figure 6. Here it is clear that we are looking at a dissertation because the appropriate text for that dissertation is displayed on the screen. There is a title, an id, an author, a director, and a list of committee members which are all displayed in a bulleted list. Here, though, we only know the relationship of the elements to one another because of the formatting of the text or the accompanying text listed near each name. It is clear Paul Dombrowski is the director because his name is adjacent to the “Director” label. The committee members are also easy to distinguish, as they are listed under the “Committee Members” label in a bulleted list. With HTML, though, the meaning for the text is only evident through its layout in the display of the web browser. The free exchange of textual information with other systems is not possible, nor is the meaningful classification and description of elements within this text by any means except for display headings (heading 1, heading 2, bold, and list format.)

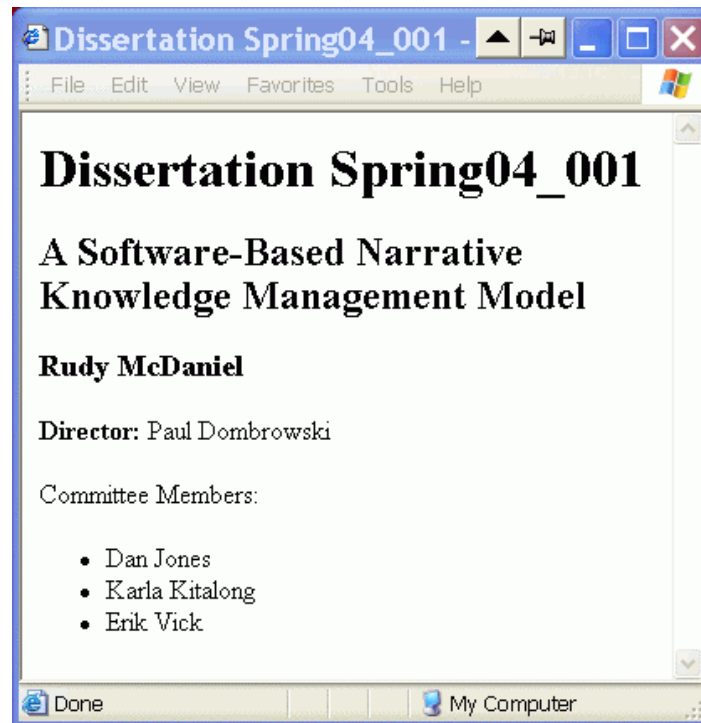


Figure 6: HTML Version of Dissertation Information

Having considered XML and some of its unique properties, we now need to consider how this markup language is moving our current World Wide Web towards the next-generation Semantic Web described by Berners-Lee, et al (“The Semantic Web” online). In January of 2004, the *IEEE Spectrum* featured a special report on innovative technologies – “six big technologies that will make a difference” – in the fields of communications, electronic power, semiconductors, transportation, computers, and bioengineering (Cass 68). The winner in the computing category was an IBM product named *WebFountain*, which is “half a football field’s worth of rack-mounted processors, routers, and disk drives running a huge menagerie of programs” (Cass 68). *WebFountain* is described by Stephen Cass as the next generation search engine, an *analysis engine*, which is capable of performing what he describes as digital alchemy:

the transforming of meaningless unlabeled data into meaningful XML labeled data (70). XML, along with the Resource Description Framework (RDF), are two technologies currently in place that enable web developers to mold their Internet webpages into a format suitable for the Semantic Web. XML has been described in the preceding paragraphs, while RDF is a model that allows for a different representation of language. For example, to express the author value of the dissertation model described previously, an RDF triple statement would be represented as: triple (author, dissertation, Rudy McDaniel).

IBM's *WebFountain* may be the first large scale corporate effort toward building this Semantic Web, which is essentially "an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation" (Berners-Lee et al. "The Semantic Web" online). *WebFountain* relies on both hardware and software support systems in order to accomplish this massive XML reclassification. Even with software compression, which reduces the amount of space necessary to store incoming data, the product requires more than 160 terabytes (160,000 gigabytes) of disk space. Processing is accomplished using 30 dual-processor 2.4GHz processors, which along with the disk drives must be upgraded approximately every nine months (Cass 70-74). Software consists of 40 specialized software "annotator" programs and specialized data mining and analysis tools that examine proprietary IBM databases as well as publicly accessible information sources such as the U.S. Securities and Exchange Commission database of publicly traded companies and the Stanford University common-sense TAP database (Cass 73-75). If nothing else, this example illustrates that large companies are investing massive amounts of time and money in order to develop software and software applications for the Semantic Web. What it also suggests, though, is that

the Internet is finally moving towards an environment supporting easier information access with less information overload and more accurate disambiguations. The process of disambiguation, as described by Cass, is “choosing the intended meaning from a range of possibilities” (73). Incorrect disambiguations are coincidental distractions, such as those provided by search engine results that return web sites listing words with the same spelling but different meanings. My belief is that a narrative analysis engine can function at a greatly reduced scale while still providing many of the same benefits of this large industrial application, and that this scaled-down model can even offer some unique benefits of its own. The specific design and implementation of this project, which I named *EDNA-E* (the Event-Driven Narrative Analysis Engine), is found in Chapter Five. Chapters One through Three are more focused on discussing the history of intelligent systems, considering and revising some classical theories of narrative, and finally explaining applied examples of successful narrative devices and applications.

There are several fundamental differences in the software I will be discussing and the software developed by IBM. The narrative analysis engine operates on a much smaller domain, and is not concerned with many of the data sources that are continually monitored by *WebFountain*, including IRC, Usenet, and giant corporate databases. Whereas IBM’s software uses “spiders,” or “crawlers,” to traverse the Internet and gather and log information, my narrative software relies on users to input stories voluntarily in response to specific events which already exist in the knowledge-base. Such an arrangement of event-driven stories is derived from Mieke Bal’s theoretical idea of the narrative fabula, which is written about extensively in Chapter Three. Before discussing narrative properties and Bal’s theory of narratology, however, it is important to discuss some of the ideas behind knowledge management and consider the

progression of intelligent systems throughout history. Such analysis of intelligent systems is provided in detail in the next chapter. An introduction to knowledge management is our next topic in this introductory chapter, however.

Knowledge management (KM) is a broad term covering many different fields of research and areas of development within various organizations across the world. Knowledge management techniques have been programmed into applied tools in many different forms and variations, and they are often classified as artificial intelligence (AI) tools and touted as evidence of progress in that field. Unfortunately, these techniques are difficult to define because of the problems involved in representing knowledge, which of course must be done before knowledge can be managed. This task is known as knowledge representation (KR), while the process of gathering and collecting knowledge is referred to as knowledge acquisition (KA). As Tim Berners-Lee et al. explain, traditional forms of “knowledge representation, as this technology is often called, is currently in a state comparable to that of hypertext before the advent of the web: it is clearly a good idea, and some very nice demonstrations exist, but it has not yet changed the world” (Berners-Lee et al. “What the Semantic Web Can Represent” online).

The first step in any knowledge management process is to collect data. Data are “facts collected on an entity” (Desouza 4) or “observations of states of the world” (Davenport 9). This data must then be somehow filtered or processed in order to become information, which can be represented as processed data put into some context (Desouza 5). Thus the relationship between data and information can be expressed as follows: $\text{Information} = \text{Context} [\text{Processed Data}]$ (Desouza 5).

Knowledge, then, is the apex of useful growth for any collected data. If data is considered as the fundamental building block of the data/information/knowledge pyramid, then knowledge is the purest form that resides at the pinnacle of this structure. Thomas Davenport defines knowledge as "information with the most value (which) is consequently the hardest form to manage. It is valuable precisely because somebody has given the information context, meaning, a particular interpretation; somebody has reflected on the knowledge, added their own wisdom to it, and considered its larger implications" (9). In a storytelling mechanism, such knowledge can be found in one or more stages of the story submission and retrieval process. For example, knowledge can be encapsulated into user-submitted stories at the beginning stage by adding descriptions of a user's experiences and that particular user's reactions to those experiences into a story. Likewise, it can emerge as an end product when a new user retrieves a story and adds his or her own life experiences and interpretations to that story. A helpful analogy for this process is the Internet forum or bulletin board, where one user starts a thread related to a certain topic and then additional users add their own thoughts, feelings, and suggestions in an asynchronous fashion. A third party can then come along at any point and make an informed decision about an issue or product simply by reading the thread of discussion from start to finish.

The idea behind knowledge management is to develop a set of processes that make it easier for humans to work with the vast amounts of data that bombard us from all angles of our lives, and to transform this data into purposeful, context-specific information. Knowledge management can therefore mean a variety of things to a variety of different people from different professions. Corey Wick explains,

It is well documented that there is a multitude of seemingly divergent definitions of knowledge management. Some emphasize information development and document management ... some concentrate primarily on the technology surrounding documents and information ... others emphasize the financial value of knowledge as intangible assets ... some emphasize cultural issues ... and still others talk about 'knowledge organizations.' (515)

This dissertation is focused on exploring traditional research in knowledge management, specifically research that has emerged as the result of developments in the field of artificial intelligence, and then on explaining how a narrative model can be used to overcome some of the flaws inherent in traditional knowledge management applications. As a result of this focus, my idea of knowledge management often encroaches on several of the various definitions mentioned by Wick above. Regardless of the specific definition used, in each model it is clear that the relationships between data, information, and knowledge play very important roles in a given organization's knowledge-building or knowledge-distributing process.

For example, we can consider the difference between data, information, and knowledge in relation to a storytelling system. In this narrative environment, the various stories collected from users represent our data. Information emerges when one of these stories presents a solution to an immediate problem for someone querying the storytelling system. Knowledge is also present at this point if the story is particularly reflective and engaging, or if the user making the query expands on the story using his or her own experiences. If this user asks the system for a story to solve a problem and a suitable story is returned, then that user is receiving contextual and purposeful information, and possibly knowledge, directly related to that problem and that

specific request. On the other hand, if an irrelevant story is returned or if the system is not successful in the process of disambiguation, then the story or stories returned are merely useless and distracting data. Of course, filtering open-ended stories is no simple task, which is why such a system needs to be limited in scope somehow. This is done either through requiring users to only input stories that follow certain themes or topics or by dynamically grouping stories around existing events within a knowledge base. In the applied project accompanying this dissertation I take a combined approach.

A detailed explanation of narrative knowledge management requires a historical summary of major developments in computing technology, artificial intelligence software, and narrative theory. Chapter Two reveals some of the primary goals and motivations of the scientists and engineers building computing technologies and intelligent systems. Here the phrase “intelligent system” is defined as a system, often computer-based, than can perform some function that would be considered intelligent if it were performed by a human user. This is, in fact, a definition that is often used for the similar phrase “artificial intelligence.” For the purposes of this dissertation, however, I examine intelligent systems that also include more primitive computer systems that served as precursors for our modern computer technologies. While some of these machines were only used for measurement or simple mathematical calculations, at the time of their creation they were certainly considered intelligent. Eventually, the narrative model is explained as an example of an intelligent system. While narratives do not need electronic components or semiconductors to function, they can certainly provide many of the same conveniences as an electronic computer, especially in regards to memory recall and organizational functions. Applied examples of narrative systems both with and without

computerized support are given, along with a framework for a programmatic narrative knowledge management model based on XML. This model functions with a human administrator who uses the software to connect appropriate story indexes in order to form a cohesive narrative. The administrator also performs various other administrative functions such as creating scripts which users can follow to input stories into the web application. Such training software using one or more human administrators has already been experimented with and is often mentioned in agent-based research in which the task is to train software programs to act as personal assistants for a human user. One such example, Lieberman and Malsby's *Turvy* program, is also part of the discussion in Chapter Two.

The approach of this dissertation's historical analysis is to outline some of the major developments in the evolution of intelligent systems. While various changes in society and culture have presented different problems and obstacles throughout history, our largest modern problem seems to be in dealing with information overload. The Internet especially holds an enormous amount of information accessible to anyone with a computer terminal and a modem or network card. Navigating this sea of hypertext, validating and securing relevant information, and organizing and linking common associations and themes between similar chunks of information is a formidable task for modern technical communicators and knowledge managers.

The same situation applies to large organizations and corporations, which collect enormous data sets that are only roughly organized according to patterns of logic and taxonomy applicable during only one given moment of that organization's history. What happens, however, when goals, directives, and motivations change within this organization? How can we continue to query the same database using the same requests for information when that

company's problems have significantly evolved? At this point, traditional static data collection systems such as databases, spreadsheets, and file-shares with collections of internal documents are simply not sufficient. There needs to be some way of collecting and storing knowledge, especially expert or unique knowledge, which will facilitate the growth of this knowledge in new and interesting directions. Such techniques have been demonstrated in the past with expert systems and similar technologies based on mathematical or statistical foundations. In this dissertation I will show that a narrative model can also function as a flexible and reliable home for specialized data that can be easily transformed into useful information.

In order to show how a narrative model is different from some of the more traditional approaches to knowledge management, I begin by examining computers and mechanical systems in general. While this analysis is by no means comprehensive, it is important because it discusses some of the fundamental properties and modes of thinking that have accompanied our technological shifts through the years. I think such a discussion is also necessary to consider a question fundamental to the topic of knowledge management itself. That question is: has computer technology actually delivered in its promise to make our lives easier? Or are we now reaching the point that Sven Birkerts, in *The Gutenberg Elegies*, terms a "crisis of meaning" (196). Birkerts describes such a crisis as emerging from the state of electronic revolution and American society in the early 1990's. He claims that the high technology of the Internet age is nothing more than glam and misdirection, and that such distraction is actually a barrier to higher thought and intellectual reflection. Birkerts writes, "And when the crisis does come, no chip or screen will have a solution for it. It will flash forth as an insistent need, a soul craving, and nothing binary will suffice" (196).

A similar theme of science as distraction is offered by Jean Baudrillard in *Simulacra and Simulation*. Here the point of attack is broadened from computers and hypertext to science in general. Baudrillard explains, “In any case, the logical evolution of a science is to distance itself increasingly from its object, until it dispenses with it entirely: its autonomy is only rendered even more fantastic—it attains its pure form” (8). While both Baudrillard’s thoughts on scientific advancement and Birkert’s thoughts on electronic communication and hypertext are admittedly severe, they do address the same fundamental weakness that emerges as a side-effect of our modern and powerful silicon computer technologies. Our computers are more efficient in space (hard disk storage) and time (processing speed) than ever before. Moore’s law, which predicts the double of transistors every 2 years, has continued to hold true; in 1971 the 4004 processor housed 2,250 transistors, while the Pentium 4 processor introduced in 2000 holds approximately 42 million transistors (Intel Research). And yet today we still work as hard if not harder to access and manipulate information within this massive electronic workspace. One problem here, of course, is that with each advancement in computer technology we are also adding more digitized information to the virtual playgrounds frequented by our machines and software programs. The software, however, has simply not kept up. Software, particularly AI software, must then also be discussed in our background narrative. More discussion of the difficulties involved with digitization, which includes a foray into Jacques Derrida’s thoughts on binary classification, is also provided in Chapter Two of this work.

The next task in ensuring the integrity of the narrative knowledge management model is to identify and classify a core set of properties we can use to describe a narrative system. This task begins in Chapter Three with a discussion of Mieke Bal’s landmark theory of narrative and

her classification of the narrative into three layers: the text, the story, and the fabula, which is a series of events that are linked to one another and influence the lives of one or more actor/agent in a particular story (9). Such a description is important in linking the narrative to a computerized model that operates in a finite-state environment. In other words, it describes how to model a story computationally. The chapter then moves to consider some of the emergent properties that can exist when stories are generated and exchanged. It also considers some of the details and approaches involved in attempting to store knowledge, and addresses some of the problems involved with using formal models for this task. Frames and scripts are offered as storage models that are more intuitive to our way of thinking about and processing our world observations. Additional properties of narrative considered in this chapter include metaphor, scope and domain, and a return to the question of machine intelligence in light of this narrative discussion.

Chapter Four continues building support for a narrative knowledge management model by gathering examples of storytelling processes in action from several different categories. Traditional, or oral, storytelling is mentioned first, with examples from several important speeches and a discussion of how stories were used to organize and support these speeches. The chapter then moves through written storytelling and computer-based storytelling systems, showing how examples from each category have contributed to our understanding of knowledge acquisition and management. The remainder of Chapter Four is devoted to considering some research in story-based systems from the banking industry.

The fourth and final chapter of this dissertation describes and models *EDNA-E*, which is an XML implementation of a narrative analysis engine. Two specific functions of the system are

first described and diagrammed in the Unified Modeling Language (UML). I then explain how the event-driven (fabula) system was implemented, and I show how users can enter and retrieve stories by querying a database to determine which stories are suitable matches for a searched event. Many detailed screenshots of the software are included in Chapter Five, along with a complete listing of source code which is available online. Table 1 in Chapter Five contains links to the main software program and the documentation files with source code for this software. My conclusion reexamines the operation of this software in the context of new research and speculates on future improvements to this type of narrative KM implementation.

CHAPTER TWO: THE EVOLUTION OF INTELLIGENT SYSTEMS

“As if driven by some invisible hand, humans have always yearned to understand what makes them think, feel, and be, and have tried to re-create that interior life artificially” – Daniel Crevier

Background

In order to understand our current information processing technologies, we must consider some of the historical inventions that helped evolve early technologies into what they are today. In this chapter, the historical analysis is preceded by a discussion of abstract thought and how this paradigm has influenced computerized information processing systems. Specifically, the connections between mathematics and computer science are touched upon, as well as a very important difference between the two fields. Abstraction is then discussed from a deconstructionist perspective which relates abstraction in the sciences to abstraction in the humanities using the ideas and writings of Jacques Derrida. His writings are compared to the operations of a digital computer, with powerful operations evident not only within the evident architectures themselves but also in the *play* between these more evident states.

After the opening theoretical discussion, this chapter moves into the age of calculating machines. These calculating machines, which manipulated the simplest type of data – numbers, are relevant because they reflected early efforts to automate and organize data into a structure that facilitated the retrieval of necessary information (generally in the form of a numeric answer).

While the narrative analysis engine presented in this document will not be using beads, cogs, or mechanical levers to manipulate arithmetic operations, the basic concept at its most fundamental level is still the same: feed in some input data and obtain some output information. Of course with modern computers we have the luxury of stored memory so our input data can be both complex and extensive, but with the earliest calculating devices such was not the case. With a story-based knowledge management system, though, stored memory is essential if this system is going to be operated on using a computer.

Explaining the basic operation of these calculating machines is *interesting*, but what is more *useful*, particularly for an information architect attempting to construct a new type of knowledge management framework, is to explain how these machines actually work when processing data at lower levels of computation. A detailed blueprint of wheel and cog operation is outside the scope of this work, but a brief sojourn through automata and finite-state operation in the modern digital computer is critical. Such a stopover introduces theoretical work from Julian Warner, Curtis Roads, Anthony Smith, Joseph Weizenbaum, and some of the leading researchers of agent-based systems including Pattie Maes, Henry Lieberman, and David Malsby. This research is mentioned in part to explore some of the limitations of AI research that are imposed by the finite state operation of digital computers. These limitations are often cited by critics of AI research to support their claims that human-like intelligence is not possible in computers. Research attempting to achieve this type of intelligence has been described by John Searle as “strong artificial intelligence” (“Is the Brain a Digital Computer” online). While the narrative analysis engine will use a lower level of intelligence, or what Searle would describe as “weak artificial intelligence,” it is important to understand some of the problems with strong

artificial intelligence before openly accepting the more flexible model. In this chapter I will show that we are nowhere near creating the technologies that will support a strong AI while our weak AI applications, such as agents and agent-based systems, are becoming more and more useful and common.

Machine intelligence also raises many other theoretical issues. Perhaps the main issue is concerned with addressing the ideas of disembodiment and posthumanism. These ideas are fairly complex, so many detailed examples from William Gibson, Donna Haraway, and N. Katherine Hayles are discussed. The ideas of disembodiment and posthumanism are, of course, not directly relevant to weak AI technologies such as the narrative analysis engine, but they do aid in showing the historical difficulties involved in achieving strong AI using computer technologies. With this foundation in place, the chapter then discusses alternate constructions of intelligence based on the boundary, again supplemented by Haraway's work and refined with some new media theory from Jay David Bolter and Richard Grusin and some visual theory from Kaja Silverman, Homi Bhabha, and Otto Fenichel.

A discussion of boundary then leads to the theoretical definition of the cyborg. Traditionally, in popular culture, the cyborg has been defined as a creature constructed from both mechanical and biological components. Haraway and her contemporaries show how this model can be extended to include other systems that exist together and are divided by real or artificial boundaries.

The remainder of the chapter is devoted to considering some classical models and examples from the field of artificial intelligence. A condensed timeline is presented, which temporarily returns the focus of the chapter to examine historical developments, although this

time in the context of AI research. Neural networks, the Turing test for machine intelligence, *ELIZA*, and frames are all discussed as important landmarks in AI research. The term “artificial intelligence” is defined according to several popular definitions, and the difference between strong and weak AI is also explained here. A brief section on chatterbots and a discussion of future research conclude this chapter.

A Trend towards Complexity

At present, in the beginnings of the 21st century, we find ourselves in the midst of a very serious problem: information overload in the information age. Luciano Floridi describes this as the problem of “infoglut,” which “concerns the degree to which retrievable information can actually be managed” (83). In the following paragraphs, I will show how our computing technologies evolved into a system capable of generating so much accessible information that infoglut is now a serious concern for organizations large and small.

Computers and mechanical systems in general have always been designed with one primary purpose in mind: to make our lives as humans easier. It is therefore surprising to see how hard humans in our current century are working; with the amount of technical innovation and ingenuity we have seen in the last hundred years, it is ironic that we have to do any work at all. Unfortunately, an increase in technological sophistication does not equal an increase in leisure and relaxation time. To understand why this is true, it is necessary to examine the history of technology and specifically the evolution of intelligent systems. It is then interesting to

consider the idea of intelligence in terms of other non-mechanical and non-computerized systems. One such example is the gender-based system of classification.

Abstraction and Intelligence

Every major technological change in the history of our culture has brought with it a profound change in perspective in how we view the world around us. Improved scientific processes and observation techniques have gradually immersed us into a more detailed environment, bringing to light previously invisible forms of data for us to observe. These paradigm shifts accompanying new technologies are inevitable considering the social and economic impact of new tools and techniques. The beginnings of organized abstract thought can be traced back to the Greeks and their introduction of classified scientific knowledge.

Russell Shackelford writes, “The ancient Greeks provided the founders of virtually every field of human knowledge: philosophy, mathematics, geometry, engineering, astronomy, anatomy, and medicine, to name but a few. All of their accomplishments share a central quality. They were able to consider phenomena from a new perspective: they were able to engage in abstraction” (8). This may seem like a minor detail to note, but in fact the idea of abstraction is what makes any system intelligent. An engineer or a designer must design a device to solve all instances of a given problem, not merely one manifestation of that problem. By organizing principles and theorizing using abstract thought, the Greeks were able to actually advance their studies far more than they would have been able to using only observation and the technology of their time. While many of their theorems and ideas had to be modified or even discarded

altogether, many of their other ideas were of sufficient quality to stand the test of time. In addition, their emphasis on abstraction paved the way for future pioneers and mechanical enthusiasts who would later use these principles in their own work.

The idea of abstraction is also critical in differentiating computing from pure mathematics. In mathematics, we have the flexibility of infinitely approximating analog values, whereas a computer must at some point stop and make a decision between two discrete points.

Daniel Kohanski elaborates,

The reduction of phenomena to numbers that fit into the orderly structures of mathematics also seduces us into believing we can avoid the unruly and chaotic real world. But the raw material of thought is information, and to the extent that we accept computer-digested data instead of seeking it on our own, our ideas about the world are based on incomplete approximations filtered through some programmer's judgment calls and the limitations of the machine. (188)

These world observations are what Davenport describes as data, and estimated data will eventually be merged into approximated information and finally personalized into knowledge. With each transformation there is a loss of precision and also a tendency towards more abstraction.

We can also address the idea of abstraction from a linguistic perspective. Derrida begins *On Grammatology* with an all-out assault on structuralism and semiology. From the very beginnings of his diatribe, he warns of the dangers of dichotomy. How, for instance, do we break down an abstract concept as language into a signifier and a signified? Derrida explains this as “inflation of the sign ‘language’” and warns that it is “the inflation of the sign itself,

absolute inflation, inflation itself” (6). Writing, therefore, is not a secondary by-product of language, but is an essential component of language itself. It “comprehends language” (7). Derrida is looking at the science of the study of writing as something different from the science of the study of language. While there is still the technology of the signified and the signifier, the act of writing is a continuously morphing activity that redefines and reconstructs the reality of the signified. For instance, is Derrida’s own writing an elaborate joke, a legitimate defense of the importance of writing, or a hell-bent joyride through classical and modern philosophical theories? Part of the point of his writing seems to be that it is often hard to tell. While Derrida uses painstaking detail to examine classical text and theory to support his argument, he seems to take genuine pleasure in deconstructing the arguments of Saussure and the structuralists.

One of Derrida’s main quarrels with structuralism is the attempt to withdraw a single, ultimate meaning from “the movement of signification” (14). He finds it impossible to arrive at a solitary meaning through language – a concept he defines as *presence*. It is such a property of language that allows two different people to arrive at two very different conclusions from observing a piece of data; to one, this data might be purposeful and therefore would represent information or possibly even knowledge. To another, the data might be irrelevant or imperfect for the task at hand, leaving it represented as simple data. Another prevalent theme in this chapter is Derrida’s distrust of the binary world, which is interesting considering our reliance on binary computers for intelligent interactions. As Hubert Dreyfus explains, there are two types of computing devices: analogue and digital. He writes,

Analogue computers do not compute in the strict sense of the word. They operate by measuring the magnitude of physical quantities. Using physical quantities,

such as voltage, duration, angle of rotation of a disk, and so forth, proportional to the quantity to be manipulated, they combine these quantities in a physical way and *measure* the result. A slide rule is a typical analogue computer. A digital computer—as the word *digit*, Latin for “finger” implies—represents all quantities by discrete states, for example, relays which are open or closed, a dial which can assume any one of ten positions, and so on, and then literally *counts* in order to get its result. (71)

Our modern computers are digital, meaning that they “operate with abstract symbols which can stand for anything” (Dreyfus 72). This ability prompted Alan Turing to describe the digital computer as the universal machine, which means that “... *any* process which can be formalized so that it can be represented as series of instructions for the manipulation of discrete elements, can, at least in principle, be reproduced by such a machine” (Dreyfus 72). It is precisely this characteristic of universality that led early artificial intelligence researchers to believe human intelligence could eventually be encoded into symbols and therefore possessed by computers as an observable property.

The very acknowledgement of using a binary system of order to represent complex human characteristics suggests that there is some point of origin at which there is a neutral value given to the signified concept. Derrida finds this concept to be impossible. The real world is analog, not digital. There is no clearly defined point of neutrality for abstract concepts such as good and evil, and these concepts rely on oppositional definitions for their very existence. When the abstract idea must further be taxonomized within itself, this binary perspective seems even more unlikely. Such is the case with language and writing. As Derrida points out, “The question

of the origin of writing and the question of the origin of language are difficult to separate” (28). If the study of writing were to look at the field of linguistics for a basis of definition, this would annoy the binary enthusiasts who claim that territory as belonging to the science of oration and speech.

This concept of digitization can also be applied to the mechanics that exist in computer architecture. Although computers are designed within the binary world, the components inside them are in fact working with analog voltages which are then encoded into their respective binary values. For example, +5 volts could be represented as a binary “1,” while -5 volts could represent a binary “0.” While there is a seeming point of origin within this system (zero volts) this is only a theoretical value that is rarely ever seen within computers (due to resistance in the wiring, interference with other signals, etc). In fact, the digital system is used in contrast to the analog system precisely because of such irregularities. Analog is reality, but binary is robust. While +5 volts is seen as the theoretical value of the binary “1,” in reality it is a range of voltage from 0 to +5 volts that is generally used in practice, and likewise in the negative direction for the binary “0.” Using this system of encoding, it takes a significant amount of interference within an electronic circuit for a “0” to be interpreted as a “1,” or vice-versa.

Even more strangely is the appearance of a third possible value within computer architecture: the tri-state. While components and circuits are indeed restricted to high or low levels of voltage within their system, there is also the transition from a high to low (or low to high) voltage that can take on a new value in itself. This very idea has led to numerous improvements in efficiency within the computer that were previously thought to be impossible within a binary system. While this metaphor may seem strained, we can think of the computer as

representing language, a new technology that enables us to communicate with one another and produce greater amounts of work in shorter amounts of time. Speech, then, would be that which is produced by the computer in its final form. And writing, if we are to consider the traditional structuralist definition, would be the binary encoding of the final material – a symbol of a symbol, or a sign of a sign.

Thus, just as in language, the true power of computer architecture is found in the “play” of the encoded signals. A “1” can only be accomplished through applying a high voltage (in both the positive or negative direction), and a “0” can only be formed using a low voltage, but the play between them (the tri-state) is very powerful because it can be achieved in many different ways (the transition from a zero to a one, the transition from a one to a zero, the maintaining of a signal for a certain amount of time, etc.). If we think of writing as the programming of language, then this type of play is even more similar. While hardware architects are making the most of their available materials to make communication within a computer as efficient as possible, so must writers work with their own materials to achieve efficiency within the bounds of printed language. Here the intelligence of the system is not confined to the normal boundaries of a dichotomy but instead is manifested through creative manipulations of abstract states.

Calculating Machines

To appreciate fully our current state of information processing technology, we must examine some of the precursors to modern mechanics and electronics. The most obvious

examples of intelligent systems are found in computers and mechanized applications.

Calculating machines, which are used to perform mathematical operations such as addition (incrementing) and subtraction (decrementing) were the first machines to make a noticeable impact both socially and economically. Michael Williams describes six basic elements in the design of most mechanical calculating machines, which are as follows: a set-up mechanism, a selector mechanism, a registering mechanism, a carry mechanism, a control mechanism, and an erasing mechanism (118-119). The set-up mechanism allowed some sort of initial configuration or data input for the machine. A selector mechanism is used to select the appropriate mathematical function and instantiate the correct mechanical movements for that function. The other mechanisms were used to control the various states of intermediate and final numbers within the device and eventually indicate an answer and allow for the reset of the registering mechanism back to zero (Williams 119). Early computational devices did not require silicon or electricity, but functioned using some type of mechanical interface with moving parts such as beads, cogs, or elaborate mechanical gear systems.

The earliest computing devices represented numerical quantities in an analog form. In analog mechanics, a continuous representation that does not depend on sampling is available for representing quantitative information. Because of the precise nature of such a representation, analog machines were useful in computing ballistic trajectories and other calculations such as navigational and astronomical charting which involved complex mathematical transformations of data using mathematical operations such as integration. The earliest analog device is probably the astrolabe, which is believed to have been in existence as early as 180 B.C. and was used for planetary calculations (Williams 194-195). Another highly sophisticated analog device, now

known as the Antikythera device after the island near which it was found, was discovered in a Greek shipwreck in 1900 and is believed to be another mechanism used for astronomical computations. This device is believed to date back to Cicero and the Romans, and contained a “sophisticated set of gears called an epicyclic differential turntable” which would not again enter into mechanical devices until the middle 1500s (Williams 196-198). More modern examples of analog machines include Lord Kelvin’s tide predictor and Vannevar Bush’s differential analyzer, which used rotating disks to calculate the area underneath curves in integration problems (Williams 198-202). Another early computing device is the abacus, which was invented more than 5,000 years ago and was used to perform arithmetic functions using rows of beads (Kurzweil 262).

More modern computing devices moved from beads to gears, with inventors such as Blaise Pascal (1623-1662), Gottfried Wilhelm Leibniz (1646-1716), and Charles Babbage (1792-1871) inventing devices that “represented data through gear positioning, with data being input mechanically to establish gear positions” (Brookshear 6).

In 1642, Pascal invented the world’s first automatic calculating machine, called the Pascaline (Kurzweil 263). The Pascaline was a fairly primitive machine in terms of mathematical capabilities, and was only able to add and subtract numbers. Fifty-two years later, in 1694, the Leibniz computer was created by G.W. Leibniz. Leibniz was a German mathematician, perhaps most famous for inventing calculus, who had grand ideas for the categorization of human knowledge. Martin Davis writes, “He dreamt of an encyclopedic compilation, of a universal artificial mathematical language in which each facet of knowledge could be expressed, of calculational rules which would reveal all the logical interrelationships

among these propositions. Finally, he dreamed of machines capable of carrying out calculations, freeing the mind for creative thought” (Davis 4). In the spirit of this endeavor Leibniz defined his symbols for fundamental operations in calculus to be closely related to their function. For example, the \int symbol denoting an integration operation was designed as a modified “S” to suggest “sum” and the “d” symbol used for differentiation was used to reflect the idea of “difference” (Davis 12). While his dream of creating an all-encompassing mathematical language for knowledge classification was never fully realized, he was able to further the development of a mathematical machine when he improved upon Babbage’s calculating machine by adding the features of multiplication and division. His machine, which depended on a device that eventually became known as the “Leibniz wheel,” was so well-received that this type of processing device continued to be used in calculating machines well into the twentieth century (Davis 8). In addition, his method of algorithmically performing repetitive additions is the same type of process that is still used in our modern computers and computer software (Kurzweil 263).

Another important calculating machine, known as the Curta and named after its inventor Curt Herzstark, was a mechanical device that could perform basic mathematical operations and was accurate up to eleven digits (Stoll 98). This device, invented in 1947, was the first true handheld calculating device and contained over 600 mechanical parts (Stoll 98). The Curta, shown in Figure 7, operated in an intuitive fashion that allowed one to make calculations using one hand, a crank, a ring that could be rotated to the left or right, and a series of sliding switches that could be set to represent the digits zero through nine (The Curta Calculator Page online). In addition, the top crank could be lifted to indicate negative values.



Figure 7: The Curta Machine (The Curta Calculator Page online)

There were of course many non-calculating machines that also had an enormous technological influence on our lives. These included mechanical programmable looms for textile production, water transporting systems, navigational and farming instruments, and numerous other devices created to lessen the load of mechanical burden on humans. For example, in the mid-fifteenth century Johann Gutenberg introduced his printing press to society. This device offered us the first glimpse of mechanical reproduction at the mass level (Shackelford 10).

Automata

Having considered some of the specific computational devices that emerged throughout history, we now need to study some of the characteristics of computing devices in order to discern their level of intelligence. A primary characteristic of a computer program is that it can be broken down and modeled into a finite number of states. These states can be represented as a series of nodes with different paths leading to different states within the system. These spatial representations of program execution are known as finite-state machines, or automata. Julian Warner speaks of the historical developments of automata from the Jewish golem (a human figure made from clay that is supernaturally brought to life) to the more restricted mechanical automatons such as the abacus, the logic piano, and finally the stored program computer (68). With the golem, we have an automaton as what Warner describes as “human simulacrum” – a copy without an original (68).

Curtis Roads has identified types of musical automata that predate even numerical computational machines. He has found records of second century B.C. mechanical carillons from the Netherlands which “were programmable, using a rotating cylinder dotted with holes” as well as a “mechanical spinet and drum set in the 1500s” built by Leonardo Da Vinci (166). These musical instruments are autonomous because they rely on a mechanical action to move them from one state (producing one musical note or pattern) to the next (producing the subsequent musical note or pattern).

Historically speaking, the properties of automata are considered to be integral to the mechanical operation of machinery. These properties, however, are certainly not beneficial to

the process of human thinking. Anthony Smith equates automata with “mental drudgery”; these are logical cycles that leave no room for creative insight or impulsive behavior. In fact, human thinking can become dependent on automata to the extent that we lose touch with our own natural abilities such as our ability to recognize the relationship between natural processes and our grasp of temporal and spatial information. Joseph Weizenbaum writes of the clock as having this type of effect on our society. He writes, “Where the clock was used to reckon time, man’s regulation of his daily life was no longer based exclusively on, say, the sun’s position over certain rocks or the crowing of a cock, but was not based on the state of an autonomously behaving model of a phenomenon of nature” (25). He later expands this argument to account for fluctuations in biological activities such as eating and sleeping, writing, “The feeling of hunger was rejected as a stimulus for eating; instead, one ate when an abstract model had achieved a certain state, i.e., when the hands of a clock pointed to certain marks on the clock’s face (the anthropomorphism here is highly significant too), and similarly for signals for sleep and rising, and so on” (25). These types of influences are so subtle that we often do not realize that we are falling into autonomous patterns of our own.

Pattie Maes has also identified autonomous patterns that influence the way we interact with computers. She has identified a transition from what she refers to as direct manipulation (where users only initiate communications with the computer) to indirect manipulation (where users and computer programs cooperate in order to communicate back and forth). She uses the metaphor of the “personal assistant who is collaborating with the user in the same work environment” (1). In this scenario, we again delegate additional control to these autonomous

agents while slowly widening the boundaries between the duties of the machine and our own responsibilities.

Henry Lieberman and David Maulsby take the idea of the personal assistant a step further and suggest we should train computers to do new tasks like we would train new human employees. This idea makes sense but is ironic considering the role reversal of humans training machines to do jobs instead of machines training humans (via aptitude testing, flight simulations, etc.). They write, "... people are extraordinarily good at learning from observing and understanding examples, especially from examples in the context of real work" (540). Because of this aptitude, Lieberman and Maulsby believe using this same type of technique is useful for "training" a personal assistant (autonomous agent) to do new and unfamiliar tasks. As they explain, "Do you want your agent to solve a new kind of problem? Show it how" (540). The agent, here a demonstrational human agent named *Turvy* (Figure 8), suggests that some type of hands-on training must be included in these training sessions in order for them to be effective.

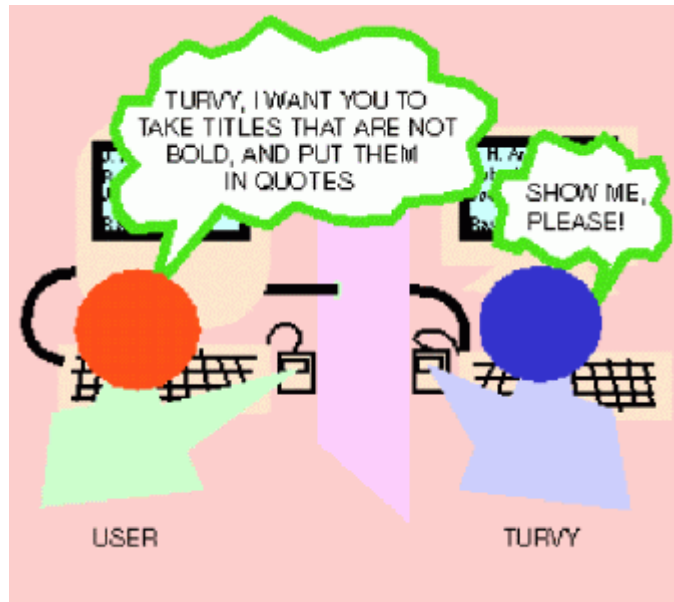


Figure 8: *Turvy*

Muriel Zimmerman envisions a different type of fusion between user and user agent. She writes, “Smart software agents residing in computers will write their own documentation, telling a highly customized story based on knowledge of user moods, learning styles, and past practices—perhaps gathering some of this information through new interfaces such as an ‘emotion mouse’ that detects a user’s state of mind. Computing will be calm, invisible, and ubiquitous” (1). Whether or not this utopia can indeed be everywhere at once without taking over our lives remains to be seen. Her idea that tranquility can accompany this omnipresent migration is especially compelling, but may be impractical as shown in the relationship between Turvy and his surveyors. Computers are still very needy, and until they have a human to push them from one state within the automata to the next they will likely be much less pleasurable to work with than Zimmerman would like us to believe. As long as our machinery is confined to

operating within a closed system of states, true human-like intelligence and behavior seems most unlikely.

Barriers to Strong Artificial Intelligence

Finite-state operation suggests there may be some hardware-based limitations for computers wishing to express creativity or human-like behaviors. Unfortunately, hardware limitations are just the beginning. Many humanists, linguists, and philosophers assert that human-level intelligence is impossible using any type of computer or machinery. John Searle, for example, is one of the leading critics of the possibility of true strong artificial intelligence. Searle writes of the homunculus fallacy, in which “the idea is always to treat the brain as if there were some agent inside it using it to compute with” (“Is the Brain a Digital Computer” online). This idea is often used by researchers in order to get around Searle’s idea that computation needs an observer to interpret the computation in order to be valid.

Other philosophers such as Hubert Dreyfus have written multiple books explaining why strong artificial intelligence in computing devices is implausible and perhaps impossible to achieve. Dreyfus’ many objections to the current methodologies used by AI researchers include their unwillingness to acknowledge their failures, their ignorance about serious knowledge representation problems, and their tendency towards citing undeveloped technologies as reasons why current AI systems do not yet work as they should (1-47). For example, at one point Dreyfus writes of robotics researchers who are waiting for an adequate knowledge representation program to emerge while at the same time the knowledge representation researchers are waiting

for a robotics model to help them figure out how to account for body-centered types of knowledge. Dreyfus explains, "... the field is in a loop—the computer world's conception of a crisis" (46).

The immediate willingness of AI researchers to accept what is known as the metaphysical assumption is also cited by Dreyfus as a serious problem in AI research. This assumption, originally identified and dismantled by Martin Heidegger, states that "the background can be treated as just another object to be represented in the same sort of structured description in which everyday objects are represented ..." (Dreyfus 56). He later extends this definition to also mean "that whatever is required for everyday intelligence can be objectified and represented in a belief system" (65). Dreyfus explains that such an assumption is in fact what Edmund Husserl would call an "infinite task" given our immense body of cultural and social conventions that influence how we see the world and how we draw our observations from the world (57). He explains, "Thus in the last analysis all intelligibility and all intelligent behavior must be traced back to our sense of what we *are*, which is, according to this argument, necessarily, on pain of regress, something we can never explicitly *know*" (57). Even prominent researchers with landmark AI applications have accepted the metaphysical assumption with no qualms. Joseph Weizenbaum, the creator of *ELIZA*, is one such example (Dreyfus 65).

Another barrier to achieving strong AI is found in the way we define commonsense-knowledge. Commonsense knowledge is of course the knowledge we as humans consider to be common sense. Unfortunately, attempts to catalog this type of knowledge have revealed that common sense facts and rules can extend to upwards of 10 million facts (Dreyfus xi). Dreyfus explains that the commonsense knowledge program actually emerged from researchers who were

trying to program computers to understand simple children's stories. He writes, "The programs lacked the common sense of a four-year old, and yet no one knew how to give them the background knowledge necessary for understanding even the simplest stories" (x). Thus a computer trying to understand the story of *Goldilocks and the Three Bears* might not understand why the three bowls of porridge situated on the table were adjacent to one another and within Goldilocks' easy reach. For humans it is common sense that we eat together at a dinner table, but for a computer this would be need to be explicitly defined and represented as knowledge. The common sense addressing the personification of the bears' actions would also need to be programmed into the software. The problem here, obviously, is that there are also hundreds and perhaps thousands of other tiny details in this story alone that we consider to be common sense. The simple problem of showing a computer how to understand a child's story has therefore grown exponentially more complex.

Dreyfus' main argument, though, is that intelligence is both situated and context-dependent. He writes, "... since intelligence must be situated it cannot be separated from the rest of human life" (62). All of the other problems involved with intelligence and knowledge representation, then, including the common sense problem, the embodiment problem, and the inherent inconsistencies with accepting Martin Heidegger's metaphysical assumption as truth, can be traced back to this idea (Heidegger 3-35). Without a cyborg or other bio-mechanical being available to house our information processing devices, the quest for strong AI may continue to be a fruitless endeavor.

As an example, one problem related to the idea of disembodiment is known as the framing problem. AI researchers are faced with this issue: how does one package a particular

world-view into a format suitable for a computer to process? Such a format would need to take into account the world/environment along with a representation of that world/environment within which to operate. Rainer Born describes one such framework for the relationship between language, information, and reality. This model is shown in Figure 9. His framework, which he describes as a “interpretive scheme, lattice LIR” (xi) is a grid-type structure in which two axes intersect one another. On one end of the horizontal axis is a “T,” which represents the scientific, technical, and theoretical concepts used to describe observations about the world. On the other vertical axis is a “V,” which represents the vernacular. On the vertical axis is a similar continuum separating the world itself, “W,” from a particular representation of that world, “R.”

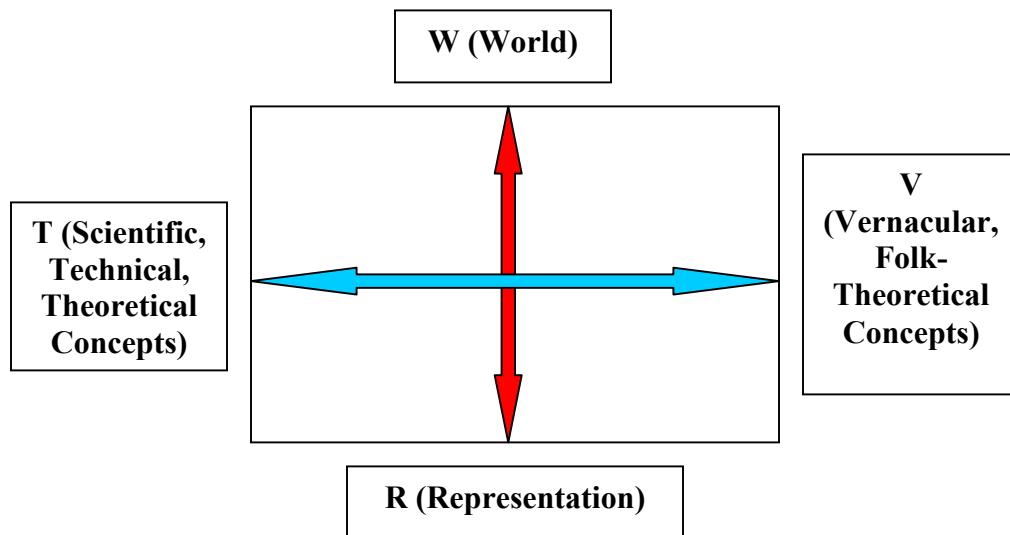


Figure 9: Lattice LIR Framework

The problem with this model is that we cannot simply sweep up the entire continuum of layers between the world and the representation of the world and package it into a data structure. The best we can do is to aim somewhere towards the center and hope for the best. The same is true for the relationship between scientific discourse and popular discourse; neither can be fully accounted for in a given instantiation of language. The same problem of subjectivity can exist even within a field of peers. This problem is also explained by Born. He writes, "... in an everyday context and therefore also in the prose a scientist uses to talk or think about his subject (in a meta-language) or which he uses to communicate the significance of his results, the abstract dimension of the significance (or meaning) of these results is suppressed" (xvii). This is likely a necessary evil of the scientific method itself. Abstract thought in meta-language form will be different from scientist to scientist, and is also necessary to determine the various specifications needed to attempt the replication of experimental results. What this type of thinking does suggest, however, is that if we were to encode empirical data into information a computer can understand in order to make an artificially intelligent scientist, that scientist would be incomplete. This subjective inner dialogue is often what prompts a scientist to revise the experiment extemporaneously or to adjust a setting or ingredient that would not logically make sense. It is also what causes them to make mistakes, which can often lead to other radical developments in science and technology, however unintended those results may be. Without a meta-language and those entirely unscientific human characteristics and emotions such as curiosity, altruism, benevolence, or even greed, the AI scientist would be an accurate measurer and experimenter, but a horrible scientist.

For yet another example of the framing paradigm, consider the theories of disembodiment and posthumanism. These theories provide us with a basis for understanding how biological information itself might function outside its natural habitat. The key question here is the following: can the human mind be transported into an artificial and mechanical body? This question, also known as the embodiment thesis question, has been debated mercilessly in artificial intelligence circles by scientists, engineers, philosophers, and science fiction writers over the last fifty years.

The disembodied experience is a characteristic of artificial intelligence that is especially appealing to many people. In *Neuromancer*, William Gibson creates a world in which users can jack-in to a virtual existence where their intelligence is manifested inside a massive computer network and their body is simply “meat” that is left behind at their terminal. Barbara Warnick describes this as “the desire to achieve immortality through the disembodied experience” (20). Creating a technology that allows our memories and intelligence to be disembodied enables us to populate a virtual world in which we are not limited by the physical demands placed on us by our bodies. There is also the side-effect of anonymity and the shedding of physical dependencies that is at once empowering and frightening. This property of disembodiment makes such a process seem all the more exciting.

The idea of disembodiment does not depend solely on the power of modern computation. The structure of information has already become disembodied through technologies such as the printing press and even the evolution of language. Anthony Smith describes this as a paradigm shift from subjective into objective knowledge. He writes, “... the chief characteristic of typographic culture as a whole is perhaps the prevalence of the very idea of objective

knowledge: the knower became separate from the author, and that which was knowable became a detached, social property, validated by reference back to the author as ultimate authority”

(126). If we apply this idea to automata, we can think of the initial state as the “author” and the final state as the new “knower” who has gathered this information through secondary means.

The purity of the information thus reflects upon the functionality of the automata. In a computer, digital logic assures that the information remains verifiable throughout each subsequent state. In an intelligent automata, or a system of computers modeled after human thinking, this property would not necessarily hold true. Such fuzzy, or unpredictable, logic is precisely the type of mechanism that the narrative analysis engine modeled in Chapter Five relies on, especially when allowing the story administrator to choose linked indexes between various users’ stories.

As one example of the changing dynamic in a disembodied state, we can consider the concept of gender. The definition of the word “gender” is debated frequently among critics, but for discussion’s sake we can use Haraway’s definition of gender as a political construction that is “at the heart of constructions and classifications of systems of difference” (130). This political construction is seriously altered when conversation takes place through the detached channels of a computer network. It was no coincidence that Weizenbaum’s natural language chatting program *ELIZA* was given a feminine name. Based on the Eliza “of Pygmalion fame,” Weizenbaum’s program was given a set of rules that would let the machine learn to speak “correctly” given a set of rules (Weizenbaum 3). In a disembodied state, though, the gender of the speaker becomes meaningless except to determine the proper formulations of pronouns and other grammatical entities. A computer simulation will not flirt with one gender and talk shop with another unless it is carefully programmed to do so. As Kolko describes in her articles about

MOOs, gender is necessary only to attribute proper syntax to spoken or written dialogues (216-217). Any further interpretation of gender in an artificial setting is done solely by human operators and does not make a bit of difference (no pun intended) to the machine itself.

Also interesting in terms of gender roles is the set of rules that enabled *ELIZA* to behave as the Rogerian psychotherapist known as *DOCTOR*. Given the fact that most psychotherapists of this time were male, perhaps people were more willing to open up to a being they perceived as feminine. Women spoke with *ELIZA* as though they were chatting on the telephone with a girlfriend, while men could be flirtatious in their conversation. Whether or not *ELIZA* responded to such queries was only a matter of changing the program that controlled the response mechanism of the machine. My own software, *EDNA-E*, is revealed to also have a feminine personification in Chapter Five. The reasoning behind my choice of personification is explained in more detail in that chapter.

Another perspective on posthuman functionality is found in the work of N. Katherine Hayles. Hayles begins examining some of these posthuman ideas by looking at some of the early works of Hans Moravec, a robotics expert, and Norbert Wiener, a scientist well-known for discovering and explaining the principles of cybernetic feedback. The underlying assumption of both of these scientists is that we can arrive at a point such that “information can circulate unchanged among different material substrates” (1). These observations, which Hayles does not seem to agree with, lead to her characterization of the posthuman. The posthuman viewpoint privileges “information over material instantiation” so that information contained in an extracted format (as in a test tube or on a floppy diskette) would be considered equivalent to that same information contained in a person (2). In this theory, each piece of information is also equivalent

within each of these different media. The posthuman theorists downplay the importance of consciousness as a defining characteristic of humanity—they believe consciousness is not the point. Rather, consciousness is nothing more than “... an evolutionary upstart trying to claim that it is the whole show when in actuality it is only a minor sideshow” (3). A third key idea of posthumanism is the idea of the body as the ultimate prosthesis—an artificial, mechanical thing that extends our physical capabilities but does nothing else. It is this idea that allows the theory of information as ether to remain within the posthuman framework. Lastly, Hayles mentions what she calls the most important assumption of the posthuman: the idea that humans and machines can cooperate together as single cohesive entity. She writes, “... the posthuman view configures the human being so that it can be seamlessly articulated with intelligent machines” (3). These four assumptions collectively represent Hayles’ definition of posthumanism (2).

This characterization of posthumanism leads us once again to *Neuromancer*. Hayles quotes William Gibson as writing of “data made flesh,” a situation in which humans regard their digitized data as more important than their own bodies (5). In Gibson’s novel, characters even risk death (flatlining) in order to remain jacked into the cybernetic system for a moment longer, trying to reach that ever elusive piece of data just beyond their grasp. In such an environment, the tenets of posthumanism no longer seem so strange. Our psyches are now identified with microcode sequences and algorithms rather than flesh and blood, and we shift our instinctive motivations accordingly. The new economy, perhaps the ultimate knowledge economy, is made up of information in *Neuromancer*. In this environment, all aspects of life must shift in order to maintain homeostasis.

Hayles is not altogether opposed to the posthuman culture. She writes, “If my nightmare is a culture inhabited by posthumans who regard their bodies as fashion accessories rather than the ground of being, my dream is a version of the posthuman which embraces the possibilities of information technologies without being seduced by fantasies of unlimited power and disembodied immortality ...” (5). She does not fear the existence of the posthuman society, but she fears the potential for danger and destruction that accompanies such a culture.

Virtuality is another important concept that is associated with posthumanism. Hayles introduces this topic in connection with feedback. She writes, “... virtuality is associated with computer simulations that put the body into a feedback loop with a computer generated image” (14). Interestingly enough, she also provides a cultural definition of virtuality as the “perception that material objects are interpenetrated by information patterns” (13). These material objects are not just penetrated by the information; they are *interpenetrated* by data that is being shuttled out of one feedback loop and into another input. These objects then appear to be physical items which can be manipulated by the body in the sense that we can pick them up and have the information coordinates transmitted back into the simulation through feedback loops. In this scenario, we actually feel as though we have picked up the objects since the graphical objects we are looking at move in accordance with our physical actions. Without feedback loops, virtuality seems to be impossible to achieve. We would have to pick up an object, then logout of the system and log back in to see the new location of the object in our hands. Transparency is lost through this complication, and virtuality has not been attained.

Hayles argues that if we are to become fully capable of posthuman functioning, we need to physically evolve into a new biological state. She writes, “If humans are information

processing machines, then they must have biological equipment enabling them to process binary code” (57). She then proceeds to explain the McCulloch-Pitts neuron and how this was supposed to help our bodies adjust to binary information. In her argument she makes the assumption that information should be forced to stay limited to a binary format in order for humans to become “information processing machines.” In reality, though, this is not true. As humans we have a harder time working with the binary numbers 100001 and 100101 than we do with their decimal equivalents of 33 and 37. The entire point of the binary number system was to allow human programmers to work within the limitations of *computers*, not of the human body. Discrete electronics operate by using a high and low state; the logical idea of a binary number system allows information to fit into this scenario. The next generation of computers, however, will not be limited by electronic components. New atomic and biological computers will have other ways of transmitting and storing information, and will likely not be limited to using a binary format for data representation. These complications will be somewhat lessened as our computers evolve into new architectures that support more than two distinct operational values.

Boundary, Gender, Stereotype, and Psychoanalysis

Having considered theories of disembodiment and posthumanism, we can now see that a definition of intelligence can be crafted in terms of a boundary between two coexisting systems that normally would not exist together. To show that such a boundary does not necessarily need to be between a computer and a human brain, we will now look at an example completely

unrelated to traditional notions of artificial intelligence. Donna Haraway provides many such examples in her analysis of the system of gender.

Haraway provides an example of such a system by discussing the early work of women scientists in male-dominated fields and relaying the troubles involved in defining what is meant by the phrase “women’s experience.” Haraway explains experience as “a crucial product and means of women’s movement” and states that “we must struggle over the terms of its articulation” (109). She feels that the true nature of women’s experience must be examined through struggle, and only then does the signification become clear.

To help explain this struggle, Haraway introduces some binary pairs to categorize and organize different components of experience. The local/global and personal/political pairs create a framework suitable for examining materials relating to this theme. The result is “an open, branching discourse with a high likelihood of reflexivity about its own interpretive and productive technology” (112). By mapping out experience in accordance with these binary pairs, one can re-construct, re-member, and re-articulate these experiences, much as one would re-experience a connection with a great piece of literature upon a second reading (113).

A reading of Buchi Emecheta helps Haraway to expand on these ideas (Haraway 115-124). She studies this author precisely because she is so different from middle-class, white, American women. Emecheta creates a literary world in which other women can connect with certain themes and relationships within that world, creating their own “webs of experience” with the material (115). This is often what ties together feminism and literature, this openness of interpretation that transfers more of the power and authority to the reader than to the writer. Haraway seems to agree with this open-boundary system. Throughout the book she is constantly

complaining about the dust jackets of books—they seem to be too confining and restrictive to the text within the work. This is too simple an explanation for Haraway.

To relate Emecheta to women's experience, Haraway focuses on how three different writers pull elements from works by Chikwenye Okonjo Ogunyemi, Barbara Christian, and her own text to create and recreate certain elements of feminist thought (117). Ogunyemi takes a womanist stance against the institution of marriage (119), Christian explores ideas of motherhood and lesbianism (120), and Haraway (herself the third reader) studies the tension and ambiguities of "Emecheta's fiction and of the fictions of her life" (121). These three readings and interpretations are all focused on different elements in Emecheta's writing, and yet they are all part of the woman's experience. In this sense, they are "all part of a contemporary struggle to articulate sensitively specific and powerfully collective women's liberatory discourses" (123). In each of these themes there is a struggle. Marriage is the quintessential struggle with identity, motherhood is a struggle with nature, and lesbianism is a struggle against what our culture deems to be the acceptable form of sexuality. These struggles become what Haraway explains as "mis-readings, re-readings, and imagined readings of a text that is originally and finally never simply there" (124). To reach out and claim this text is to assert dominance over the struggle and claim the experience as one's own.

In Chapter Seven of her book, Haraway identifies another traditionally sexist agenda in language: that of the referential canon. Even works of reference can be tainted by the dominance of male-oriented language through the centuries, and Haraway does an extremely thorough job of defining gender and the sex-gender system based on the work of contemporary theorists and feminists in the field of gender studies. The disassembly of "the canonization of language,

politics, and historical narratives in publishing” (129) becomes another important feminist struggle for control in both science and literature.

Like any deconstruction, this process must be done one step at a time. Haraway starts with her definition of “gender.” Her main point about this signifier, as it should be, is to stress the distinction between sex and gender. To make this distinction, Haraway once again depends on the struggle between binary pairs. She writes, “In all their versions, feminist gender theories attempt to articulate the specificity of the oppressions of women in the context of cultures which make a distinction between sex and gender salient. That salience depends on a related system of meanings clustered around a family of binary pairs: nature/culture, nature/history, natural/human, resource/product” (130). The conspicuous differences between gender and sex can be realized through an analysis of what is naturally determined (sex) and what is socially, culturally, politically, and historically determined (gender). As Simone de Beauvoir has pointed out, “one is not born a woman” (301).

There are many other examples in which a complex ideological or cultural idea is explained through the relationship, or play, between two boundaries. The idea here is that the true meaning of some idea or concept can only be found within a set of relationships between two or more other ideas or concepts, which may or may not be related. For another example we can consider Jay David Bolter and Richard Grusin’s idea of remediation, in which old technologies are recycled and recreated through new technologies. According to this theory, remediation is achieved through the oscillation of media between immediacy and hypermediacy, or being “there” and being “beyond there” at the same time (5). Hypertext therefore remediates

the printed text, since the text is at once there as it has always been while it is also empowered with hypertext's additional features such as anchored links and multimedia capabilities.

Usually, this play between boundaries depends on an idea floating between two different signifieds, such as the appropriate response for an idea in the virtual world versus the appropriate response for an idea in the physical world. Kaja Silverman describes this concept nicely as a “radical oscillation between contrary emotions” which “characterizes all relationships of the imaginary order” (344). Homi Bhabha adopts a similar strategy to describe the strategy of the stereotype, which he claims to be a vacillation between what is already known and what must be anxiously repeated (370). In both cases the signified idea is not set in stone (an ironic choice of words considering the relationship between the fixed gaze and being turned into stone/denied existence considered by Otto Fenichel) but is rather an idea that depends on two different signifiers, as well as the relationship between them, for existence. As Silverman explains Lacan's ideas about the subconscious, this region is “an inarticulate and chaotic region” and yet still “a signifying network” (347). This conflict between chaos and order presents an immediate problem to the idea of machine intelligence. Machines are built to understand and respond to logical sequences and rigid rules. So how do we build a machine that has such a set of signifying rules built on chaos and ambiguity? Here once again is the problem of overcoming systems of finite automata. The best we can hope for is to design a system that can do the best job possible given the finite number of choices available in a computational path and then let the human subconscious fill in the blanks. Or, as with the narrative analysis tool presented with this dissertation, to program a system capable of supporting thematic linkages and then allow a

human supervisor to approve or delete these linkages in order to improve the overall accuracy of the tool's intelligence.

In considering the problem of machine intelligence, it makes sense to consider how a machine can be programmed/trained to mimic the five senses of a human being. For vision and sound this is not much of a problem since we can train the machine to recognize mathematical variations in sound and light waves, and to interpret these variations in a way similar to a human's natural response. For example, in a machine programmed to respond to sound or visual cues in the environment, these perceptions can be configured to send feedback to the machine's motor subsystem and adjust movement or response mechanisms accordingly. Thus we can make machines that seem to be intelligent simply by recognizing and responding to language; Silverman reminds us of Barthes claim that "language mediates all other signifiers" (346).

Psychoanalysis and psychological models can also testify as to the validity of boundary-based intelligence. The very existence of these models depends on the boundary between what is considered normal and what is considered abnormal behavior. It is no coincidence that *ELIZA* was one of the earliest experiments in artificial intelligence. This program was built to listen to problems and respond to natural language typed into a terminal. Using the principles of Rogerian psychoanalysis, the program would try to persuade the subject/person to expand on their subject by repeating back the previous input in the form of a question. For example, this might be done in the following manner:

Person: My father doesn't appreciate me.

ELIZA: Why do you feel like your father does not appreciate you?

As Fenichel mentions, though, other senses such as smell present a new challenge since “in olfactory perception the introjection of minute particles of the object is actually real” (333).

Regardless of the current validity of psychoanalytic constructs such as the Lacanian “lack” or “imaginary register” or the Freudian fetish, these ideas and constructs do make it clear that there is more to human behavior than immediately meets the eye. If we are to design convincing machines in the image of human behavior we must take into account these theories of boundary. If nothing else, these types of boundary theories may explain the oddities in human behavior better than the definitions theorized about and generated by traditional AI pioneers.

Cyborgs and Androids

The cyborg is an excellent example of an intelligent system that embodies popular culture’s visions of AI. Hubert Dreyfus notes, “... workers in AI often take refuge in the idea that computers will finally achieve human understanding when they have humanoid bodies” (25). By allowing a computerized information processing system to exist in a biological body, cybernetic and robotics researchers hoped to minimize or avoid the embodiment problems involved with the process of trying to represent somatic concepts within a non-somatic entity.

One popular cyborg researcher is Donna Haraway. We have already discussed some of Haraway’s ideas about boundary and gender; the cyborg is actually implicitly related to both of these topics. In Haraway’s *A Cyborg Manifesto*, for instance, we return to the discussion of boundaries. The cyborg “appears in myth precisely where the boundary between human and animal is transgressed” (152). Again we have the comparison of science to myth, and the

implications of socially-constructed knowledge in this science. As our technology becomes increasingly sophisticated, it becomes more difficult to tell the difference between the natural and the artificial. Even modern computers are being fitted with biological processors in research laboratories and it is only a matter of time before these scientific advances trickle down into our own homes. Then the ghost in the machine becomes that much more frightening. At this point the machine is made of the same materials we are.

The rise of the microelectronic world has also influenced the way we work and communicate with one another. Personal meetings have become a thing of the past with email and the Internet, and even large conferences are now being broadcast or simulcast through Internet technologies. Haraway characterizes this trend of technological dependence by writing, “our machines are disturbingly lively, and we ourselves frighteningly inert” (152). This is a comment modern Luddites such as Sven Birkerts would certainly appreciate, but it also addresses the popular fear of losing our humanity by surrendering to the temptation of artificial immortality.

Of course a relaxed definition of intelligence also leads to a relaxed definition of cybernetics. Feminism is tied into the cyborg manifesto through its common bond with the cyborg myth of “transgressed boundaries, potent fusions, and dangerous possibilities which progressive people might explore as one part of needed political work” (Haraway 154). Gender is one such boundary that can again be drawn out and examined. Which personal characteristics should be examined in order to determine when a person is considered masculine and when they are considered feminine? Emotionalism? Dominance? Once these characteristics have been identified, how much of these certain qualities must a person possess (or how far past the

boundary line must they travel) in order to be qualified as “being” a certain gender? Perhaps we are asking the wrong types of questions altogether. Haraway notes, “there is nothing about ‘being’ female that naturally binds women. There is not even such a state as ‘being’ female, it itself a highly complex category constructed in contested sexual scientific discourses and other social practices” (155).

The cyborg feminists do not want an all-encompassing “natural matrix of unity” (157). Instead, they argue for an idealized form of self that reflects social constructions of knowledge and gender. Revisiting her sociobiological roots, Haraway explains, “communications technologies and biotechnologies are the crucial tools recrafting our bodies” (164). In this model, the “biggest threat to power is interruption of communication” (Haraway 164). The cyborg becomes useless when disconnected from a power source, but the cyborg community does not become useless until communication is cut out from underneath them all. Movies including *The Matrix* and its sequels, and scientific articles such as “Computers” by Lewis Thomas, suggest that this type of communal foundation is necessary in order for true human-level artificial intelligence to exist.

One frightening aspect of these modern technologies is the fact that the goals suggested by visions of human engineering are now actually attainable. Cloning and engineering technologies have made significant advancements in decrypting genetic sequences and bringing man into the realm formerly governed by nature alone. Even the activities that differentiate the human from the machine are being slowly “learned” and programmed into our latest technologies. Haraway writes, “Microelectronics mediates the translations of labour into robotics and word processing, sex into genetic engineering and reproductive technologies, and

mind into artificial intelligence and decision procedures” (165). The very ideas of production and reproduction in our society are being threatened by the machines we have created to make our lives easier.

To Haraway, the feminist movement faces the same threats from modern technology as it did from the scientific discoveries of the primatologists she writes about in her book. The only difference here is that the threat now comes in new forms. Stress (manifested through a breakdown in communication) and dominance (found in those who control the machinery) still manipulate and control the mechanisms of the “high-tech myth systems structuring our imaginations of personal and social possibility” (169). To combat and redefine these stereotypes and sexist challenges, the cyborg feminist must be equipped with a keen understanding of the nature of communication and a strong desire to break out from under the swaying rhythm of the collective assembly.

Hayles also considers the posthuman cyborg in her writings. With the discussion of several novels by Philip K. Dick and his writings about cyborgs, Hayles moves away from the discussion of how information lost its body and refocuses her attention on the “cultural and technological construction of the cyborg” (160). We are now examining how the historical construction of the human is giving way to the post-human. The cyborg is important because this concept takes the ideas Hayles discusses in her earlier chapters and gives them an arena to play in.

Dick acknowledges the importance of boundaries to the android form, and Hayles remarks on this in her introductory paragraph on Dick’s work: “Consistently in his fictions, androids are associated with unstable boundaries between self and world” (160). Although they

are strangers to the human population by design, this does not keep humans from being curious of (and even attracted to) androids, and vice-versa.

In order to understand the nature of the android, we have to once again change our perception of virtuality and information exchange. Hayles explains, “When system boundaries are defined by information flows and feedback loops rather than the epidermal surfaces, the subject becomes a system to be assembled and disassembled rather than an entity whose organic wholeness can be assumed” (160). To this extent, we need a way to study how androids communicate in order to figure out how we define them as separate entities. Is an android only singular when its information is independent from all other androids, or do special rules apply for informational boundaries? To accept the ideas of boundaries for information, must we accept that information *does* have context and meaning, or would that mean denying that androids could even store such information?

To answer some of these questions, Hayles studies the work of Humberto Maturana. His work is intriguing because he abandons the idea of the black box and decides to focus instead on the “autopoietic system” (160). Autopoiesis is characterized by systems that “(a) maintain their defining organization throughout a history of environmental perturbation and structural change and (b) regenerate their components in the course of their operation” (Whitaker online). Both of these attributes are obviously well-suited for the cyborg entity. Environmental perturbation and structural change is a given for cyborgs; they must exchange information through different types of media, and often from one mechanical structure to a completely different one. The regeneration of components works as a feedback loop to keep the cyborg aware of changes to their environment and to their own bodies.

Maturana was concerned with observing the power struggles within the autopoietic system. He believed such power struggles were commonplace, and that through such struggles “the weaker system is made to serve the goals of the stronger rather than pursuing its own systemic unity” (160). A parallel example here is shown in Sandra Harding’s thoughts about American and European “assistance” to weaker countries in which the same sort of political struggle comes into play (39-54). Perhaps the same is true for our hypothetical android community.

It was also important to Maturana to consider the outside influences of the environment on a closed system. Hayles writes, “Instead of treating the system as a black box and focusing on feedback loops between the system and the environment, he treated the environment as a black box and focused on the reflexive processes that enabled the system to produce itself as such” (160). This unique perspective allows the designer to take a step back from their creation and watch their android play with the world and be nurtured (or conversely be spoiled by outside influence).

Dick also mirrors many of the ideas of Jean Baudrillard, which Hayles brings into the discussion of the simulacra. One of the themes of his work is the “persistent suspicion that the objects surrounding us—and indeed reality itself—are fake” (Hayles 161). In order to know what is fake, though, we must have a working definition of authenticity, which is quite different in the land of the cyborgs. The first step is obviously to decide whether or not artificially created systems can qualify as living. If they can, then we lose the security of physical determinations for deciding what is authentically living and what is a machine. Hayles writes, “Authenticity does not depend on whether the being in question has been manufactured or born, made of flesh

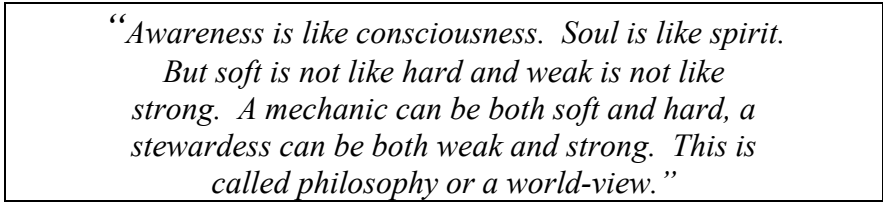
and blood or of electronic circuits. The issue cannot be decided by physiological criteria” (163).

We now have ambiguity about not only what is living, but also what is *human*.

Applied Artificial Intelligence

Theories of boundary, gender, feminism, simulacra, and remediation are all very interesting, but they are not generally valued by practitioners working in industry. The managers, engineers, programmers, and technical communicators actually working with knowledge and information want something concrete they can work with to help save time and money for their organization. Dreyfus explains four traditional areas for applied AI research and development: game playing, language translating, problem solving, and pattern recognition (85). Examples spanning each of these categories are provided in this chapter.

Unfortunately, the evolution of artificial intelligence as a practical field has been somewhat disappointing. Consider the stanza of poetry shown below in Figure 10.



*“Awareness is like consciousness. Soul is like spirit.
But soft is not like hard and weak is not like
strong. A mechanic can be both soft and hard, a
stewardess can be both weak and strong. This is
called philosophy or a world-view.”*

Figure 10: An early poem by RACTER (Güzeldere and Franchi online)

The poem shown in Figure 10 provides a thought-provoking analysis of some abstract ideas such as awareness and our general state of consciousness. It mentions the ethereal qualities

of our souls, and then seems to wander off subject to discuss random issues not related to these abstract ideas. In fact, by the end of the poem, the reader is somewhat bewildered as they try to piece together what exactly the author is trying to say. The bad craftsmanship of this poem is not particularly interesting, nor are the metaphors and figurative language supporting this rambling verse. What is meant to be interesting about this poem is that it was written by a computer.

RACTER is an “artificially insane” computer program (Güzeldere and Franchi online).

Whenever software programs enter into the creative realm to do things like converse with humans, write poetry, compose songs, and go insane, we are looking at an example of applied AI.

As human beings we cannot help but be fascinated by what applied AI has to offer. The possibilities of intelligent machinery are constantly being regurgitated in literature, film, and modern art. Most recently, a resurgence of popular interest in AI has emerged in the movie theatres with films such as *AI: Artificial Intelligence*, *The Matrix* and its two sequels, and *Bicentennial Man* drawing in large numbers to the box office.

Historically speaking, there were many important social and technological events that led up to our modern ideas about (and implementations of) applied artificial intelligence. Comprehensive listings and timelines chronicling major AI milestones can be found elsewhere in research and online; it is my intent to only discuss a few key developments influencing the birth of applied AI and sculpting this field into what it is today. In 1818, Mary Shelley published *Frankenstein*, which remains one of the most cautionary tales of the relationship between man and machine. The year 1835 brought about the invention of the electric relay by Joseph Henry, which allowed electrical current to be controlled by switching the relay into the on or off

position. This coupled with George Boole's development of symbolic and binary logic in 1847 formed the basic foundation for computational logic.

In 1917, based on his play, Karl Capek coined the term "robot" which in Czech means "worker" (Kantrowitz 1). Following shortly thereafter was the pioneering work of John Von Neumann, who is considered the father of today's stored memory-based computer architecture. One of Von Neumann's most significant contributions to the field of applied AI was his construction of an algorithm, or ordered process, for minimizing losses and maximizing gains. This algorithm was aptly named the minimax theorem. The minimax theorem was arguably the first real application of weak artificial intelligence. Rule-based programming would allow computers to "play" games of strategy with a human opponent, where the objective would be to maximize the points for the computer and eventually win the game. Von Neumann's theory was later applied to computerized versions of popular household games like *Checkers* and *Backgammon*.

The first "neural-network architecture for intelligence" was proposed by Warren McCulloch and Walter Pitts in 1943 (Kantrowitz 2). This groundbreaking research not only introduced the finite-state machine as a model for computation (McCulloch and Pitts online), but it also planted the idea that the computation done by the human brain could somehow be codified into a mathematical language and understood. This idea eventually came to be known as the symbol-system hypothesis. Unfortunately for researchers, they would soon discover that the number of states available to the human mind was infinitely complex and quite impossible to predict with mathematical certainty.

In 1950, Alan Turing, another computer science pioneer and founding father, published “Computing Machinery and Intelligence.” This article built upon his earlier theories addressing the procedure in which computers could be used to “handle symbols such as humans do in the process of thinking” (Smith 131). The term “artificial intelligence” was coined in 1956 by John McCarthy at a Dartmouth College conference and marked the point at which AI began to be considered a distinct entity from the information sciences (Buchanan 3). In 1958, Von Neumann’s famous article comparing the human nervous system to a digital computer was published. The MIT Artificial Intelligence Lab was founded by Marvin Minsky in 1959 along with McCarthy (Generation5 online). Minsky, whose ideas about frames are discussed in detail in Chapter Three, is considered by many to be the father of AI.

After the 1950s, many researchers began to consider the emergence of artificial intelligence technologies as an actual possibility. Despite these high hopes, fifty years of progress have still not delivered these technologies to our society. Our current technologies have not brought us any closer to true artificial intelligence any more than they have delivered the paperless office that has been hyped for the past few decades. As Fred Tonge writes, “... there is a large difference between saying that some accomplishment ‘ought to’ be possible and doing it. Too often, when some interesting behavior is produced, the common reaction is, ‘So what. What’s his name (was it Turing, or Jules Verne, or Isaac Asimov?) suggested that years ago’” (379). With our modern technologies, however, we are moving into new and alternative domains for the production of processing encoded instructions. These new domains may give the field of AI the breakthrough it needs in order to produce more compelling examples of artificial life and intelligence.

The late 1960s proved to be a monumental period for research in artificial intelligence. In 1966, Joseph Weizenbaum was working in the MIT AI Lab when he created the most famous AI application yet, a small program named *ELIZA*. *ELIZA*, introduced earlier in this chapter, was a natural language processing program that could converse with users based on a script which gave the program a set of rules to follow for different types of conversations. *DOCTOR*, which was *ELIZA* using a Rogerian psychotherapy script, soon became famous around the world for listening to people's problems and offering advice that seemed to show reasoning abilities within the program. In reality, though, *ELIZA* was "based on very simple pattern recognition, based on a stimulus-response model" (Wallace 1). Weizenbaum's contribution to AI was especially unique because he "paid no less attention to the moral aspects of AI than to the research itself" (Generation5 online). Weizenbaum saw computers as "tools to expedite our daily lives" and did not believe in putting the "technical advances of AI above the ethics" (Generation5 online).

By the late 1960s, we had computers that could reliably beat the world's most talented checkers players. In 1975, Marvin Minsky "published his widely-read and influential article on Frames as a representation of knowledge, in which many ideas about schemas and semantic links are brought together" (Buchanan 5). Minsky's frame model is discussed in great detail in Chapter Three, especially his ideas about story-frames, which are specific examples of his general frame concept. In 1974, Lewis Thomas published his article "Computers," proposing that true AI could only be found in a system of computers and not in a solitary supercomputer-type machine.

The most recent advancements in AI (gaining popularity in the late 1990s) have occurred in the areas of autonomous agents and ubiquitous computing. Autonomous agents are knowledge-based systems that perceive their environment and act on that environment to realize one or more goals (Tecuci 1-2). Ubiquitous computing devices are “computers on the go.” Using these technologies, firefighters and paramedics literally wear computers in order to have access to large amounts of information without losing their sense of mobility (Zimmerman 2).

We now have an idea about some of the technologies and historical events which have sparked research in the field of AI. With this background in place, we can address the ideas behind artificial intelligence itself as a topic of study. This abbreviated discussion of applied artificial intelligence concentrates on some of the dominant research concerning AI from the fields of computer science and information technology. An operational definition of artificial intelligence is provided, along with an outline of some of the major advancements in AI throughout the years.

To define applied artificial intelligence operationally, we must first define what is meant by intelligence. Intelligence can be defined as “a property of a system, a judgment based on observation of the system’s behavior and agreed to by ‘most reasonable men’ as intelligence” (Tonge 379). Using this definition, “artificial intelligence” becomes “that property as observed in non-living systems” (Tonge 379). So, in order to talk about a computer or a mechanical device as having artificial intelligence, we would need to identify observable properties that would convince most reasonable people that it is acting intelligently. Of course this idea is based on one key assumption, that “intelligence is not restricted to living systems” (Tonge 379). For

those believing in the possibility of artificial intelligence, this is obviously an assumption they are forced to make.

Artificial intelligence as an academic interest emerged sometime in the mid-20th century following the publication of Turing's famous article "Computing Machinery and Intelligence" in 1950. In this article, Turing introduced and explained the Turing Test, which is mentioned in virtually every book or article ever published about computer-based intelligence. The Turing Test is explained in more detail in the next section of this chapter. McCarthy's most recently revised definition of the term describes artificial intelligence as "the science and engineering of making intelligent machines, especially intelligent computer programs. It is related to the similar task of using computers to understand human intelligence, but AI does not have to confine itself to methods that are biologically observable" (McCarthy online).

While a more precise definition of artificial intelligence is debated among researchers, it is generally agreed that AI studies are concerned with various information processing systems. Some claim that AI should only be concerned with artificial systems of information, such as those found in computers or other electro-mechanical devices. Others, like Aaron Sloman, believe that the field should consider natural information processing systems as well (online). Sloman identifies several principles related to AI, including studies of the ways in which "knowledge is acquired and used, goals are oriented and achieved, information is communicated, collaboration is achieved, concepts are formed, and languages are developed" (online).

To understand the concept of artificial intelligence, it is important to differentiate between true artificial intelligence and artificial pseudo-intelligence. Most experts in the field use the dichotomy between "weak" and "strong" AI to separate these theories. Strong AI theory

operates under the assumption that “computers can be made to think on a level (at least) equal to humans” (Crabbe and Dubey online). We have yet to see an actual application of strong AI in our technology. Weak AI, on the other hand, has a dramatic presence in modern technology. Expert systems found in search engines like *Google* use rule-based systems to narrow down information and find the most pertinent “hits” for a given series of keywords. Voice recognition software uses intelligent algorithms to find the most likely match for a spoken phrase. Computer games use weak AI in “a combination of high-level scripts and low-level efficiently-coded, real-time, rule-based systems” (Crabbe and Dubey online). Software programs frequently incorporate AI technologies because “learning is essential for a software agent; no software developer can anticipate the needs of all users” (Leiberman and Maulsby 539). In other words, weak AI applications make existing computer programs more useful, convenient, and speedy. What weak AI applications do not do, however, is think creatively and critically in the same way that humans can when processing information.

So, as a society, we longingly wait for the wonders of strong AI to materialize while we are already immersed in the influences of weak AI. The narrative analysis engine presented in Chapter Five of this dissertation is but one example. It is also important to note that our essential humanity is not necessarily threatened by the coming of AI. Richard Lanham writes in his thoughts about artificial life: “Evolution, human and animal, actual and potential, is being charted by a field of computer-based technology called ‘artificial life.’ Into all these interactive environments the literary imagination, the fictional impulse, enters vitally. The personal computer has proved already to be a device of intrinsic dramaticity” (6).

In other words, if we believe what Lanham has to say, we will inject our own sense of drama and humanity into these virtual worlds. He suggests that our own creative energies are required to put the “life” into “artificial life” and thus these energies are needed to encourage advancements in this field. From this viewpoint, artificial intelligence will not replace humanity but will act more in accordance with the notion of Derrida’s supplement: it is both an extension to and a replacement of our current notion of humanity. This replacement is not necessarily maladaptive, however, since our creative fictions are the fuels that sustain these virtual worlds.

Chatterbots and the Turing Test

The most famous examples of weak AI applications are probably found in the chatterbots that best fit the Turing Test model of intelligence. Chatterbots are types of bots, or software agents, that “are used for chatting on the Web and are a form of entertainment” (Murch and Johnson 46). We have already examined one popular example that fits into this model, which is Joseph Weizenbaum’s *ELIZA*. The Turing Test is a game involving three people that is used to determine whether or not a computer program has “intelligence.” The test is described by Turing as follows:

The new form of the problem can be described in terms of a game which we call the ‘imitation game.’ It is played with three people, a man (A), a woman (B), and an interrogator (C) who may be of either sex. The interrogator stays in a room apart from the other two. The object of the game for the interrogator is to determine which of the other two is the man and which is the woman. He knows

them by labels X and Y, and at the end of the game he says either ‘X is A and Y is B’ or ‘X is B and Y is A.’ The interrogator is allowed to put questions to A and B.
(online)

This type of conversation based scenario was important, as Turing claimed that mathematical arguments would be of no help in deciding if a machine could think (Crevier 24). Although the passage is described in terms of gender, Turing later extends this experiment to allow for one participant to be a computer program mimicking a human subject. In other words, a human interviewer is connected by a computer terminal to another person and a computer program imitating a person. If the interviewer fails to determine which subject is the person and which is the machine, then the machine has proven to be intelligent according to the Turing Test.

The problem with the above test, which Turing acknowledges, is the poor definition of intelligence upon which the experiment is based. For example, we can contrast the view of detached intelligence to the ideas Plato discussed about writing in the *Phaedrus*. Julian Warner writes,

The extent of comparability between the concept of human intelligence implied by the *Phaedrus* and that implied by the Turing test is difficult to determine. A tendency to substitute considerations of intelligent behavior for intelligence in the development of the Turing test allowed the idea of intelligence to remain poorly developed, and not necessarily self-consistent over time, in the associated discussions of artificial intelligence. (62)

In this example, written words only offer the appearance of intelligence because the original author is not present to provide context to the words. Nor are they available to answer

any questions that might come about from reading that particular piece of writing and deciphering the embedded thoughts within that writing. Whereas Plato depicts writing as an impure form of speech that “appears” to be a faithful reproduction of essence, the intelligence of Turing’s test could be described in a similar manner. Although the machine subject may fool a human interviewer, it is a deception based on tricks and clever programming rather than authentic thought. From this perspective, any rule-based system of programs such as Weizenbaum’s *ELIZA* or other technologies adapted to excel at Turing’s interview game will never be truly authentic examples of artificial intelligence.

For example, consider the following exchange between *ELIZA* and *PARRY*, who is another chatterbot programmed to possess a paranoid personality. This dialogue is shown in Figure 11.

parry: People get on my nerves sometimes.
eliza: I am not sure I understand you fully.
parry: You should pay more attention.
eliza: Suppose you should pay more attention?
parry: You're entitled to your opinion.
eliza: What makes you think I'm entitled to my opinion?
parry: It bothers me just to be around people.
eliza: Please go on.
[Parry continues with some paranoid ramblings about horseracing, bookies, and the Mafia]
parry: I know the mob controls the big rackets.
eliza: Who are you really talking about?
parry: Have you ever really thought about the Mafia?
eliza: Suppose you ever really thought about the Mafia?
parry: Let's talk about something else.
eliza: I am not sure I understand you fully.
parry: You keep asking me the same things, why not change the topic?
eliza: In what way?
 [.]
parry: I am getting tired of answering the same questions over and over.
eliza: What resemblance do you see?
parry: Why do you need to know that?
eliza: What do you think?
 [.]
parry: You harp on the same questions too much.
eliza: What does that similarity suggest to you?
parry: I don't get you.
eliza: Do you wish that you did get it?
parry: It's about time you showed some interest in my feelings.
eliza: Oh, I showed some interest in your feelings?

Figure 11: ELIZA and PARRY Conversing (Güzeldere and Franchi online)

The above conversation seems to make sense on the surface level, but it clearly would not convince someone using the Turing Test in order to decide if they were talking to another person or a machine. In this sense, even machines cannot talk fluently with one another without some degree of disjointedness between them.

One additional problem with the Turing Test, besides its reliance on a limited definition of intelligence, is related to natural language processing. It is incredibly hard to make a machine speak and understand our natural languages. Stephen Coles elaborates, “While processing English sentences for input to a question-answering system, we must not only find the associated parsings of each sentence, but also create internal representations suitable for making necessary inferences and computations” (157). In other words, we must not only tell the computer how to logically understand a sentence but how to internalize that sentence as well. This seems like an impossible task considering how difficult it often is to communicate certain ideas to another human. The problem is exponentially more complex when it is a computer program trying to speak with a human. Even award-winning chatterbots such as *ALICE*, which has been named most indistinguishable from a human being, have quirks that reveal them as human within the first five or six sentences of the conversation (Feinberg, 1; Wallace, online).

Chatterbots are considered AI applications because they meet the operational definition of AI often used by the AI pioneers; we would consider conversation among humans to reflect intelligent behavior. There is also the relationship of natural language to computer language. Computer software (or more specifically any given computer language in which that software is written) is very similar to natural language. Both are sign manipulation systems, both rely on specialized hardware for interpretation, and both translate to gibberish when considered by someone unfamiliar with the rules and syntax that govern that particular language. The key difference, though, is the specialized hardware on which these languages are interpreted. The brain, which processes our natural language, is still very much a mystery to us. Modern efforts have in fact made it common practice to use computer processing to attempt to understand more

about the brain in efforts such as the Caltech Human Brain Project, which “assembled a diverse group of computational scientists and neuroscientists with a common interest in addressing questions about the formation of elaborate structures and the acquisition of complex function in the developing brain” (Jacobs et al. online) The computer processor, which processes our computer languages, is understood very well.

In fact, the computer processor is so simple to understand (after all, we designed it to be efficient) that its underlying operation can be simplified down to one key mechanism: the toggle switch. As we know, computers are binary; at the lowest level their most primitive operator is a single digit which can take on a single value of either one or zero. In terms of hardware these two values are polar opposites: on or off, do conduct electricity or do not. Boolean logic, which introduces set functions such as AND, OR, and NOT to these bit operators, can then be used to plan surprisingly sophisticated hardware mechanisms and control devices using thousands of logic gates embedded onto transistors, which are then used to construct the overall processor. Such details may seem outside the scope of the discussion of a narrative knowledge management model, but in fact they serve as background material for another assertion I will make in this dissertation: true strong artificial intelligence is unlikely if not impossible using current digital technologies. What I offer as an alternative is the practical example of an applied weak artificial intelligence tool that will help to combat information overload and function as a critical communication device in large organizations.

Into the 21st Century

The future of strong artificial intelligence as we enter the 21st century is fraught with uncertainties. The newest technologies in artificial intelligence in the past ten years have been nothing more than sophisticated elaborations of earlier implementations of weak AI. The natural language processing ability of AI applications has progressed at a similarly disappointing rate. Conversational software exists more as a novelty than as a useful tool for cross-cultural communication. The search engine *Google*, for example, now has a specialized web directory dedicated to the web-bots spawned by *ELIZA* and her children. As of this writing, there were more than fifty different bots available to satisfy our every conversational need. Among the more flavorful bots are Jesus Chat, Elvis Bot, Jack the Ripper Bot, the Disputed, Illegitimate Son of Mgonz, Android Bot, and sarcastic *ELIZA* (Google online).

Perhaps the largest barrier to achieving strong AI is the lack of a collective group of machines with intelligent behaviors. As Lewis Thomas notes, “It is in our collective behavior that we are most mysterious. We won’t be able to construct machines like ourselves until we’ve understood this, and we’re not even close” (476). Manuel De Landa has drawn similar conclusions through his own research in the field of Artificial Life (AL). He has identified “emergent properties,” which are “characteristics that are not displayed by individual members of a population but only by the group as a whole” (798). For this reason, it seems as though nanotechnology, or biological computers, may be our best bet for implementing artificially intelligent machinery, if strong artificial intelligence is even possible. Giving these machines the ability to reproduce on a cellular level is certainly a scary thought, but some means of

communication between one agent and the next is necessary to achieve any type of collective intelligence.

The future of weak AI looks much less disappointing, however. Search engines are becoming more and more intelligent, with companies like *Google* spending millions of dollars on research and development for next-generation smart Internet agents. Similarly, social and political institutions such as libraries are beginning to implement the technologies theorized by Anthony Smith in 1993. As he notes, "... the computer absorbs the knowledge in a whole field of expertise and helps us to use it intelligently" (131). There are also clearly identified needs in instructional design and online help systems that weak AI can certainly help to meet (Pratt 2). Curtis Roads has identified a need for AI in music composition, performance, theory, and digital sound processing (167), and L. F. Arrowood et al. have identified the needs for weak AI in the military in the areas of knowledge-based consulting (409), situation assessment (409), knowledge-based battle simulation (410), and mission planning (411).

From the wide range of research being done in many different areas, it is clear that despite the lack of success in this field there will continue to be funding and support for projects working with artificial intelligence applications. Once alternative computing technologies mature to the point of stability, I believe we will begin to see more impressive demonstrations of weak AI using quantum computing and nanotechnologies. As for strong AI, I believe we are still several breakthroughs away from experiencing the types of technologies shown in *Bicentennial Man* or similar movies. The "Age of the Cyborg" is not yet upon us, nor does it appear to be coming any time in the near future.

In the next chapter, an alternative model for an intelligent system will be presented. Rather than attempting to engage in stimulating conversation or exhibit human-like behaviors, this system will instead rely on the associative and connotative properties of human memory to engage a user in a storytelling system that will allow them to read stories and create their own stories organized around specific events. This type of system would certainly be classified under the “Weak AI” category of applied artificial intelligence, but it nonetheless offers a powerful means for dealing with organizational change and the exchange and delivery of human tacit and expert knowledge.

CHAPTER THREE: PROPERTIES OF A NARRATIVE SYSTEM

“The universe is made of stories, not of atoms” – Muriel Rukeseyer

Background

Introductions to intelligent systems, calculating machines, and artificial intelligence were provided in Chapter Two; this chapter shows how the properties of a story can be broken down and identified in order to model this story computationally. Such properties also enable an information architect to markup and identify particular story elements using XML. The information included here is meant to build upon the first chapter by demonstrating some methods that can be used to encapsulate stories into a form that can be processed by our modern calculating machines, which are of course computers. Chapter Three also shows how some of the problems with traditional artificial intelligence applications – including the lack of a social and cultural context – can be solved by using stories to represent experiences and knowledge.

The first step in the process of identifying story elements is to provide definitions useful for the process of storytelling and then describe the various layers used in the narrative process. This procedure is equivalent to studying the parts of the story. The works of one theorist in particular, Mieke Bal, are especially helpful in identifying story elements that can be represented using language a computer can understand. Diagrams showing these story representations using

both finite state automata and object-oriented class diagrams are included to add clarity to the textual explanations.

Next, the story as a whole is considered using examples of emergent properties from the physical sciences and theorizing about how the emergent properties of stories might be useful to our informational needs. Examples from our DNA, chemistry, and even insect organizational patterns are used to illustrate the manifestation of emergent properties in certain complex systems.

After the whole and the parts of the story have been considered, I explain how knowledge can be represented and stored in a storytelling system. This is perhaps the most important part of the chapter because it equates a popular model of computation and storage in the human body to a comparable model of computation and storage in a computing device. In other words, it shows how a human's process of thinking and remembering can relate to this same type of process in a computer. The theory used for this relationship, Marvin Minsky's frame model, is explained in detail.

It would be hypocritical to write a volume of information praising the potential applications of the story in the field of knowledge management without including a story of my own. To avoid this hypocrisy, I have included such a story in the section of this chapter related to life experiences. This story, which I call the *SparkleTech* story, shows how a storytelling system is used to solve a problem in a typical corporate environment. This section also briefly mentions the ability of stories to aid in decision-making tasks and the process of mnemonic recall by touching upon personal life experiences.

The final sections of this chapter outline a few additional benefits of using stories to communicate knowledge, including the very important influence of metaphor in this process. A fundamental problem with this narrative paradigm, which involves the issue of controlling scope in a storytelling system, is also discussed, along with the proposed solution of using a restricted domain model when accepting stories. Chapter Three closes with the assertion that technical communicators and Texts and Technology (T&T) practitioners can be important contributors to this type of knowledge management system given their expertise and ability to “take technical information and make it understandable to those that need it” (Hughes 275).

Narratology and the Narrative System

The properties of a narrative system are best explained by first considering some of the classical research studying the narrative itself. Many theorists have written books attempting to develop a comprehensive theory of narrative; one such theorist is the French rhetorician Gérard Genette whose works have been translated as *Narrative Discourse: An Essay in Method* and *Narrative Discourse Revisited*. Genette’s theories, which are both interesting and useful, are based around the writings of a single work: Marcel Proust’s *Remembrance of Things Past*. Using this text along with several classical works including *The Iliad*, *Madame Bovary*, and *Moby Dick*, Genette set out to systematically describe narrative properties that can be used in general narrative analysis.

In *Narrative Discourse*, Genette discusses the three most popular definitions of the word “narrative.” The first and most common meaning for this word “has *narrative* refer to the

narrative statement, the oral or written discourse that undertakes to tell of an event or a series of events ...” (25). In this sense, we might refer to the narrative of a particular speaker or writer.

The second meaning, “... less widespread but current today among analysts and theoreticians of narrative content, has *narrative* refer to the succession of events, real or fictitious, that are the subjects of this discourse, and to their several relations of linking, opposition, repetition, etc.”

(25). This second definition of narrative is analogous to Mieke Bal’s idea of *fabula*, which represents the plot of a story. The *fabula* is discussed in more detail in upcoming paragraphs. A third and final definition of narrative has this word “refer once more to an event: not, however, the event that is recounted, but the event that consists of someone recounting something: the act of narrating taken in itself” (26). This definition requires an active participant, a person who is actually doing the narrating, in order to make sense. In this chapter, the word “narrative” will at times refer to both the first and third definition noted by Genette; the event succession mentioned by his second definition will be referred to using Bal’s term “*fabula*.”

Having defined the word “narrative,” I will now differentiate this word from the familiar derivation “narrating” and the similar semantic idea of the “story.” Here Genette and Bal agree on their distinction between the two. Genette writes,

I propose, without insisting on the obvious reasons for my choice of terms, to use the word *story* for the signified or narrative content (even if this narrative turns out, in a given case, to be low in dramatic intensity or fullness of incident), to use the word *narrative* for the signifier, statement, discourse, or narrative text itself, and to use the word *narrating* for the producing narrative action and, by

extension, the whole of the real or fictional situation in which that action takes place. (27)

Using semiotic theory to differentiate between the story and the narrative is helpful here; the story is essentially the general idea and content while there may be many different narratives for that single story. For example, we can today purchase stories which exist in the narrative forms of printed books, electronic books, audio books, films, movies, videos, plays, and so on. The signified story will not change, even though it may exist in several different signifier mediums.

With adequate definitions in place, I now need to break down and study the narrative itself. Mieke Bal defines this type of study as narratology, which is “the theory of narratives, narrative texts, images, spectacles, events; cultural artifacts that ‘tell a story’” (3). Such a global definition is handy for allowing flexibility in the classification of stories—we now have visual stories, cultural and ideological stories, and even simulated stories—but at the same time this definition makes it very difficult to narrow down and describe a global body of narrative works or even to decide if one individual text is in fact narrative or not (Bal 3). It is therefore necessary to define a core set of properties which can be applied to a given text to see if it is in fact a narrative. A narrative system, then, is “a description of the way in which each narrative text is constructed” (Bal 3). By disassembling the various components of a narrative system, we can theorize about the boundaries by which narrative texts are defined.

In order to arrive at a fundamental set of properties for a narrative system, the issue of the text itself also deserves attention. Narratology itself is concerned with the narrative text, so it is necessary to decide upon the boundaries in which a text can operate. Bal’s definition here is

again a good one: a text is “a finite, structured whole composed of language signs” (5). Bal also stresses that the finite property of the text should not be misunderstood here. She writes, “The finite ensemble of signs does not mean that the text itself is finite, for its meanings, effects, functions, and background are not. It only means that there is a first and last word to be identified; a first and a last image of the film; a frame of a painting ...” (5). In other words, it is the medium of the text, and not truly the text itself, that is identifiable within these boundaries.

With a definition for the text in place, it is now possible for Bal to discuss some of the properties of a narrative text, including the story itself and the properties that make up a story. She writes that a narrative is:

... a text in which an agent relates (‘tells’) a story in a particular medium, such as language, imagery, sound, buildings, or a combination thereof. A story is a fabula that is presented in a certain manner. A fabula is a series of logically and chronologically related events that are caused or experienced by actors. An event is the transition from one state to another state. Actors are agents that perform actions. They are not necessarily human. To act is defined here as to cause or to experience an event. The assertion that a narrative text is one in which a story is related implies that the text is not identical to the story. (5)

Using this classification we can now determine one fundamental property of the narrative text: at the lowest level there must be some agent involved that causes a transition from one state in the story to another. This transition is triggered by some event in the story. These states and event transitions are then connected using language, sound, or imagery to create the structure Bal defines as a “fabula,” which can be likened to the mechanical event framework of a story.

Bal claims that by using her narrative definitions it is possible to classify narrative texts using a three layer distinction: text, story, and narrative (6). Of these layers, “Only the text layer, embodied in the sign system of language, visual images, or any other, is directly accessible” (6). It is the textual layer that is first “seen” by a reader. After this, the fabula is interpreted, “... an interpretation influenced by both the initial encounter with the text and by the manipulations of the story” (Bal 9). So a reader, listener, or viewer, all of which are legitimate processors of the text according to Bal’s definitions, can only interact with the textual layer itself. This interaction is accomplished by scanning across the words on a written page, processing the rapid series of frames in a film, or listening to the sound waves from a musical instrument or recording. Each of these actions can also be divided between active and passive states of response. A reader can easily read the words on a page without comprehending them, just as a person can listen to a conversation without actually *hearing* that conversation.

Both active and passive responses depend on the reader/listener/viewer being able to respond to the particular medium in which a text is embedded. For example, someone untrained in Braille would not be able to read the text of a story printed in the raised dots of that medium. Likewise, a hearing impaired person would be unable to listen to a story that was spoken in a frequency range inaccessible to that listener. It is also important to note that the “average reader” of a particular story is not interested in the various layers of narrative texts; this distinction is only useful to the “narrative analyst” attempting to determine the effects of a text on readers (Bal 6). Or, in the case of a software program such as IBM’s software *WebFountain* which is mentioned in Chapter One of this dissertation, such an analysis might be performed in order to better describe data on the Internet. In the narrative analysis engine presented in

Chapter Five, for example, such analysis attempts to dig through the textual layer and reach part of the fabula, or event-layer. This process then allows a story to be described in XML according to the actors present in a story as well as through the event structure of that story. To connect stories, then, the system only needs to look for the same actor to appear in multiple stories, or for similar events which may occur in separate contexts. Allowing a user to enter a story's theme will only improve this process as the narrative analysis engine can work using both an internal knowledge base as well as with programmatic suggestions from users.

The primary organization of a narrative text, according to Bal, is determined by the event fabula. The fabula essentially represents the plot of the story (Theune et al. 2). This event series is held together by certain rules connecting the events, actors, times, and locations where events occur in this fabula. These rules are collectively described as "elements" of the fabula (Bal 7). These elements are then organized in a manner which produces the appropriate pathetic or logical effect intended for a particular story. Rules describing the relationship of these event elements can then be produced for the purposes of analysis (Bal 8). These rules could perhaps be described as syntax for narrative, and would incorporate common sense rules about time and place as well as more complicated rules taking into account readers' expectations about event plausibility and character development.

The story layer is concerned with the movement of one or more actors through the event fabula. Andrew Glassner writes, "At its core, a traditional story describes a sympathetic character who want [sic] something desperately, and confronts a series of escalating obstacles in order to obtain it. The character may in fact not reach his or her goal in the end; it's the struggle and not the conclusion that forms the heart of a story" (52). We could represent this struggle as a

finite state automata as shown in Figure 12. Here, the starting point “A” would be the entry point for a given character in the story. This entry point is represented with an enlarged curly bracket in this diagram. During the story itself, which might take place after the narrator has finished laying the appropriate background elements into place, this character would start out by experiencing some event that is indicated by circle “A.” At some point in the story, the actor/character does something that moves them into circle/event “B.” At this point, that character may continue to experience different instantiations of the same basic event or may move directly into event “C,” which represents the character’s achievement of their goal. This final state in the narrative fabula is represented by the dual concentric circles.

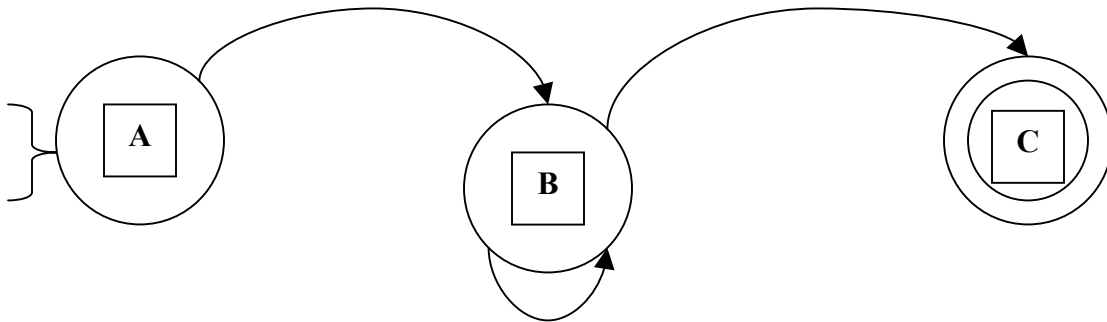


Figure 12: Finite State Fabula Representation

We can move through this narrative automaton using a fairy tale story as an example. In this tale, there is a prince who must rescue a princess that has been captured by an evil dragon

and flown across the sea to his dragon cave. The narrator explains this situation to the reader, which would effectively position an agent, in this case the prince, at the beginning bracket before circle “A” in Figure 12. We can represent the first event in this sequence, indicated by circle “A,” as the point at which the prince realizes he must go after the princess in order to save her. Perhaps he has realized that he is in love with the princess, or maybe he finds a prophecy chiseled into a stone somewhere committing him to action. In any case, here I will define event “A” in this fabula to be the prince’s acceptance of his mission to rescue the princess. At this point, the prince moves into event “B.” I will define event “B” as the trials and adventures the prince encounters while trying to achieve his objective. As an “adventure event,” this state in the fabula may be revisited many different times using the self-referential path defined by the arrow pointing from event “B” back to event “B.” Note that there is no way in this particular story for the prince to go back and rethink his decision to embark on a rescue mission. In this story, the prince must either continue adventuring forever or eventually rescue the princess and defeat the dragon, at which point the character would reach the final state “C” and the story would end.

If this were an actual story embedded into a text, the story itself would, to some degree, be open to interpretation. A broad category might be assigned based on some of the classification rules used by the fabula, but an analysis of the story itself is highly subjective and cultural. For instance, if the prince were to pass through all of his adventures without harm only to die when facing the dragon (an action that would also end the story and leave that actor in event “C” of the automata) two different readers might both agree that the broad category of this particular narrative is “tragedy.” If they were asked to describe the story in its entirety, however, a very large number of interpretations would be possible. One reader might describe the tale as a

traditional sexist tale of a man rescuing a woman only to be defeated by his own pride at the end of the story. Another might characterize the story as a tragic love story illustrating the struggle of man against nature and the inevitable triumph of the natural world. Either of these interpretations, if supported with appropriate details from the text, is equally valid. Neither the text nor the story actually changes, though – these alternate descriptions equate to two different fabula created from the same story. Bal explains, “The fabula is really the result of the interpretation by the reader, an interpretation influenced both by the initial encounter with the text and by the manipulations of the story” (9). So, it is quite possible and even likely that two readers of the “dragon steals princess that must be rescued by prince” story would come up with two different event automata for that particular story. Event “B,” for instance, might be divided up into several different adventure events each with their own defining characteristics. Such an organization might make it impossible to use a self-referential link.

The agent, who is telling a particular story, is not the author of a book or the composer of a symphony. Story agents are somewhat ethereal; they jump into existence only when active in the story itself. Bal writes, “... the narrator does not relate continually. Whenever direct speech occurs in the text, it is as if the narrator temporarily transfers this function to one of the actors” (8). Depending on the particular layer being studied, there may be an “actor” (in the fabula), a “character” (in a story), or a “speaker” (in a text) (Bal 9).

From this discussion, Bal has concluded that three main characteristics can be used to identify the narrative text:

1. The existence of two speakers in the text, one of which plays a role in the fabula and one of which does not. This holds true even when the narrator and

the actor are the same person, as in a first person narrative. Here, ‘the narrator is the same person, but at another moment and in another situation than when s/he experienced the events.’ (9)

2. The existence of the three layers described in the above paragraphs: the text, the story, and the fabula, all of which are ‘describable.’ (9)
3. The existence of ‘contents’ within the narrative text that are conveyed in ‘a series of connected events caused or experienced by actors presented in a specific manner.’ (9)

It is now possible to construct a narrative system which can operate as a knowledge management tool. In this type of situation, a narrative system is simply a communication system by which stories are the means of information transfer. Stories are not be the only forms of knowledge possessed by a given sender or receiver, but they are the means of packaging and distilling that knowledge into a format suitable for transfer. Analogous units have been termed “chunks” or “blocks” of information by positivist theorists who have asserted that all knowledge can be separated into discrete units capable of transfer from one location to another. The main benefit of using stories as encapsulating units is that this construct works well with both positivist and social constructivist theories. In other words, a narrative system enables the transfer of knowledge and information using a packaging strategy without ignoring the social factors that help to shape that information before, after, and during its transit. Many researchers of human intelligence and information theory are beginning to realize the efficiency and power of such a narrative construct. As Roger Schank explains, “Our knowledge of the world is more or less equivalent to the set of experiences that we have had, but our communication is limited to

the number of stories we know to tell. In other words, all we have are experiences, but all we can effectively tell others are stories” (12). Using this paradigm, stories are the shuttles in which we transport life experiences and knowledge from an origin to one or more points of human contact.

Storytelling is a process that is quite intuitive and natural to humans. Walter Ong writes of the importance of mnemonic structures in pre-literate cultures as a means for organizing complex thoughts and ideas in a manner that facilitated their easy retrieval (34). Stories, in this sense as conveyed by an oral storyteller, are the perfect mnemonic tools for this type of mental organization. Here, information is not static and independent from other things we “know” in memory. Instead, our new knowledge is linked to previously existing knowledge which makes for an easier memorization and a quicker retrieval. By embedding critical concepts, definitions, or even mathematical formulas into a narrative format, one can quickly memorize and categorize new pieces of information by using these fresh ideas as mental signposts that signal transitions from one phase of a story to the next. A forgotten signpost thus signals that some key idea has been forgotten, prompting one to reexamine the story and find out the point at which the story diverged into some new and unexpected domain. In this regard, stories can also function as regulators for data integrity much like parity bits function to verify the correct operation of stored data in computer memory. We have a certain expectation of how each story should proceed, which is based on our experiences reading similar stories or perhaps even some life experiences that are similar to events in that story. When our story as a whole does not meet that expectation, it is necessary to reexamine the individual data points to try and find the faulty point

or points in the chain of events that make up the story. Locating this point will likely highlight the corrupted or missing data.

A narrative system is also very similar to the way we mentally store and categorize information in our modern culture. While our stories have become more complex and more synthesized with technology, the underlying mechanisms of linkage and retrieval are still the same. Because of this instinctive familiarity, there are many advantages associated with using a narrative system for any scenario in which managing information and knowledge is critical. Some of the major benefits of using such a narrative format are discussed by Stephen Denning. Denning writes, “Storytelling is natural and easy and entertaining and energizing. Stories help us understand complexity. Stories can enhance or change perceptions. Stories are easy to remember. Stories are inherently non-adversarial and non-hierarchical” (xv). Each of these benefits is considered in more detail as specific properties of narrative systems are discussed throughout this chapter. For example, the discussion of metaphor highlights the feature of stories as helping us to understand complexity and considering narrative knowledge management strategies demonstrates how natural and easy it can be to augment our memory with stories and narrative indices. I discuss more of Denning’s research on storytelling properties in Chapter Four.

Building a narrative system in its primitive form is actually quite easy. Gather two or more people with life experiences and have them tell each other stories. This simple dialogue is all that is necessary to observe the properties of a narrative system. Building an efficient multi-user and distributed system, however, is quite a bit more complicated. To accomplish efficiency in a storytelling system, it is necessary to understand in detail the properties of a narrative system

and find the best means for fine-tuning an environment for the delivery of informational stories. Chapter Five of this dissertation is therefore devoted entirely to implementing an analysis engine that can work with the narrative properties identified in the opening of this chapter. Such a software system is used to build a robust and capable story management system. Once a storytelling system is in place, it is possible to observe an entirely new set of informational patterns in the form of emergent properties.

Emergent Properties

Emergent properties are without a doubt some of the most interesting characteristics of any complex system, and the narrative system is no exception. Combining two simple building blocks with fundamental properties and then realizing a combined product with entirely new properties seems almost magical. In order to understand fully the properties of this new product, it is not necessary to understand the operation of the product's various subcomponents. The beauty of a product with emergent properties is that the whole of the product works in a way that is much different from the workings of its various parts.

There are numerous examples of emergent systems in our everyday environments. One critical system that exhibits emergent properties is our own genetic substructure, composed of deoxyribonucleic acid (DNA). Our DNA is synthesized from nucleic acids with quite different properties from DNA itself, and in fact emergent properties are a fundamental focus of modern biological studies. As Neil Campbell explains, "with each step upward in the hierarchy of biological order, novel properties emerge that were not present at the simpler levels of

organization” (4). It is precisely this ability of biological systems to synthesize new levels of complexity from simpler building blocks which makes our bodies so incredibly detailed and functional. Manuel De Landa has taken this evolutionary approach a step further and studied emergent properties in relation to nanotechnology and communities of artificial life (798).

Another example from the biological world is found in the social order of insects. Eric Bonabeau et al. write of the impressive social behaviors of insects such as leafcutter and weaver ants, honey bees, and tropical wasps when carrying out social tasks such as nest building and food collection (1-6). Though each individual insect is operating at a primitive level, the overall community of insects operates smoothly and efficiently to solve a particular problem. They theorize that the modeling of such insect societies can help researchers to design communities of robots that work together to solve problems. This concept is known as “swarm intelligence” (7). What makes these swarms intelligent is not any one operation at the individual level, but rather the overall actions and accomplishments of the swarm. By simply following their intuitive patterns and transporting materials and supplies from one location to the next, these insects are contributing to an overall order of efficiency that can only be observed from a higher level of observation; by us, for instance. By taking a step back and examining the system as a holistic entity, intelligent patterns can be observed.

The final example of emergent properties from the applied sciences comes from the field of chemistry. Chemical reactions, which are of course also a major part of biological reactions, often combine two reactants to create a product with new and unique properties. The formation of table salt from sodium and chlorine is a simple example of this. John Kotz writes that table salt (sodium chloride) is formed by the reaction of a sodium metal with chlorine gas (12-13).

The interesting thing about this process is that table salt is an edible compound whereas its two ingredients in unreacted form are not only inedible but in the case of chlorine also poisonous. Furthermore, the physical properties of sodium and chlorine are very different from the physical properties of table salt. Such a reaction is a clear example of emergent properties in a chemical process.

An example that is more relevant to a business model but perhaps less interesting overall is the general day to day operations of any distributed organization. Here there are various employees doing various jobs which in turn combine together in various higher-order levels to allow that organization to operate. A measurement of success can be obtained through management reports which should illustrate an overall impression of a particular segment of that organization, which may or may not describe how well employees are working within that segment. In other words, the combined work of employees is what leads to organizational success much more so than Herculean individual efforts.

While this sounds very much like common sense, it actually seems as though corporate entities are reluctant to accept this idea, at least in Western culture. Promotions and accomplishments are often awarded and recognized individually rather than communally, regardless of what emergent properties of success and profit might arise. Of course it does not make any sense to reward unproductive employees, but it does make sense to look at the overall team dynamic and overall accomplishments and perhaps move employees around to a different environment in order to help the community as a whole.

A narrative model can be used to collect and share information in such a distributed environment, and appears to be the best tool for managing tacit knowledge in such a setting.

Here we have a collection of stories that relate individual experiences in various forms within a given organization. These stories may be about a person's experience in troubleshooting a particularly nasty technical problem, about a situation in which someone arranged special considerations for accessibility for a website launch, or about any one of thousands of other personal stories that have happened to employees in this organization.

Paul Hildreth et al. also note the importance of a narrative model in the information and knowledge-gathering process (348-349). Their article talks about knowledge in terms of hard and soft knowledge, which roughly correspond to the ideas of explicit and tacit knowledge. They explain the major problems facing large organizations, including globalization, downsizing, and outsourcing, and talk about the loss of soft (tacit) knowledge in regards to all three of these scenarios. Hildreth et al. explain hard knowledge as using a capture-codify-store cycle to manage information, while soft knowledge is perpetuated through more complex activities such as technical workarounds or the telling of technical "war-stories" in which veterans relate their experiences to newer employees, often in a humorous or entertaining manner (350). They also discuss Lave and Wenger's process of legitimate peripheral participation (LPP) which is another theory for how soft knowledge is managed within an organization (354). They conclude the article by examining two case studies and observing how soft knowledge is arranged in two large corporations; their conclusion is that face to face contact is a very important part of the process (355). This has significant implications for an AI-based system trying to learn (or teach) tacit or soft knowledge within a knowledge management system.

Storing Knowledge

Emergent properties in a narrative system are present because of the ways in which we index information from stories. In this context, indexing is actually a way of referencing what we can refer to as stored knowledge. There are many different ways to define knowledge, but Thomas Davenport's is perhaps the most succinct and appropriate for a narrative model. As stated in Chapter One of this dissertation, Davenport defines knowledge as "information with the most value" and explains that "it is valuable because somebody has given the information context, meaning, a particular interpretation; somebody has reflected on the knowledge, added their own wisdom to it, and considered its larger implications" (9). This definition places an appropriate emphasis on the social dimension of knowledge, which is also critical for a narrative model. Without some sort of community ideology or social anchorage, our shared stories would be unintelligible, and we would have no explicit way to exchange knowledge and information within a story-based system. Bal reminds us, "... interpretation is both subjective, and susceptible to cultural constraints – framings – that make the process of more general interest" (11). She also adds that this interpretation "... turns narrative analysis into an activity of 'cultural analysis'" (11). This subjective property means that different cultures may construct very different fabula from a given story constructed in a given text. The framing process is explained through more examples later in this section.

Stories also excel at storing knowledge because they can capture the contextual details linked with a particular piece of knowledge. These details may include related ideas, themes, or even emotional linkages that could affect the interpretation of the story. We remember stories

about our own experiences because this is such an efficient way to remember events that may have happened long ago. Donald Norman writes, “stories are marvelous means of summarizing experiences, of capturing an event and the surrounding context that seems essential. Stories are important cognitive events, for they encapsulate, into one compact package, information, knowledge, context, and emotion” (129). Traditional quantitative methods for information modeling, while testable and seemingly robust, simply cannot encapsulate these contextual connections that are so important to the knowledge itself. These methods are often based on statistical methods and probability, which while ideal for ensuring data integrity are not sufficient for measuring and evaluating stored tacit knowledge.

Not only does a narrative system store data, it also facilitates the sharing of tacit knowledge. What this means is that a narrative model can be used as a technique for representing knowledge that is difficult for the knower to explain or describe. This type of knowledge is in contrast to explicit knowledge, which is more easily explained by the knower. Michael Hughes explains the difference between the two as follows:

Explicit knowledge is knowledge that we know we know. It can be articulated, codified, stored, transferred, and recalled through symbols. In short, it is the knowledge that can be transferred through documents. *Tacit knowledge* is knowledge that we do not know we know. It is difficult to articulate and generally is expressible only through action. (278)

The narrative model is ideal for representing this type of knowledge because it too is built around action. Agents, characters, or actors (depending on which level of Bal’s narrative model we are looking at) move through an event fabula only through their own actions. A story, then, can

explain how someone reacted to a certain situation and express knowledge that person did not fully understand even when writing that story. When two people with similar goals or tasks to perform read stories that trigger their own mental experiences, tacit knowledge is evident even if the original authors may not be able to easily express this knowledge through speaking or writing it down.

Other knowledge representation techniques traditionally include mathematical and logical systems such as predicate logic and production systems as well as object-based scenarios such as semantic networks (Sharpley et al. 63). Each of these techniques has its various benefits and disadvantages, but traditionally the mathematical and logical models have been those which have been considered most apt for practical systems. Unfortunately, even these models have serious flaws which can make structuring and representing knowledge a formidable challenge.

For example, logical systems such as predicate calculus are often confusing. They are also very mathematical and depend on a strong mathematics background for one to be able to even understand simple relationships within the model. As a result, they may not be the best way to explain something as intuitive as our own knowledge. Norman writes, "... logic is artificial: it was invented some two thousand years ago. If logic were a natural property of human thought, people would not have so much difficulty understanding it or using it. Logic, however, is a branch of mathematics, not of natural thought" (129). This is not to suggest that logic should be abandoned altogether. Without some sort of logical foundation, gathered stories would resemble unorganized and random thoughts plucked arbitrarily from a stranger in a crowd. There must be some logical boundaries which encapsulate and define the context of a given story. These boundaries, however, must also be loose enough to allow the story to accept

seemingly irrelevant patterns that may in fact have some impact on the final product. Good written stories often do not simply present an idea in the first paragraph and then logically connect each idea in that paragraph until the idea itself is exhaustively considered. The logic in these types of stories is more subdued, with themes developed and expanded through character development and the trials of a protagonist. Such logic is important because it allows the reader to make connections from the story with events from his or her own life, making the story not only entertaining, but memorable. A rigid, mathematically defined system of logic is not an ideal tool for this type of problem.

Semantic nets can be equally overwhelming. A semantic net is a “large linked data structure in which pointers are used to indicate associations among the data items” (Brookshear 475). The flexibility of using pointers, which are simply references to the appropriate locations in memory where data is kept, makes the semantic net seem very similar to our nervous system. Unfortunately, there are several problems with this model too. For one thing, our nervous system is processing many times more interactions than even the most complex semantic net is able to process. As John Holland notes, “the behavior of the central nervous system depends on the interactions much more than the actions. The sheer number of interactions—hundreds of millions of neurons, each undergoing thousands of simultaneous interactions in a thousandth of a second—takes us well beyond any of our experiences with machines” (3). Another problem with attempting to use semantic nets to model intelligence and store knowledge in computers is that however similar they may seem they just do not function in the same way. As early as 1958, researchers such as Von Neumann were writing of the difficulties in modeling such biological processes as nerve impulses using binary logic (486). Even philosophers have taken on the task

of contrasting the human nervous system and digital computers. John Searle writes, “the brain ... does no information processing. It is a specific biological organ and its specific neurobiological processes cause specific forms of intentionality” (“Is the Brain a Digital Computer” online). If we accept this argument, then the entire idea of semantic nets is not very useful; it doesn’t make sense to develop an information processing model based on an original entity that does no information processing itself.

Fortunately, frames and scripts do seem to be more intuitive to the human mind. These models of knowledge representation share many similarities. The most important similarity is that they share a common theme: the idea that “our knowledge of concepts, events, and situations is organized around *expectations* of key features of those situations” (Sharples et al. 69). This is the cultural framing process Bal refers to in the interpretative process of narrative analysis (Bal 11). Key theorists of frames and scripts include Marvin Minsky and Roger Schank, whose research has helped to define the narrative model as a viable tool for information and knowledge management.

Minsky explains that frames are nothing more than “structures we’ve acquired in the course of previous experience” (244). He writes, “We all remember millions of frames, each representing some stereotyped situation like meeting a certain kind of person, being in a certain kind of room, or attending a certain kind of party” (244). Such a model is equivalent to Schank’s ideas about conversation, in which frames between the various conversationalists are linked back and forth or perhaps contain similar ideas. The structure of the frame is described as “a sort of skeleton, somewhat like an application form with many blanks or slots to be filled” (Minsky 245). These blanks inside our frames are “terminals” which are used “as connection points to

which we can attach other types of information” (Minsky 245). Blank terminals are then filled with specific information in order to create a particular instance of an item being framed. Scripts are simply a modified version of frames, reinvented by Roger Schank to “encode the essential steps involved in stereotypical social activities” (Dreyfus 40). I use the script concept in Chapter Five, for example, to create XML files which control allowable story elements and themes for user-submitted stories.

Dreyfus believes Minsky’s frame idea has a foundation in Husserl’s concept of the noema, which is “a symbolic description of all the features which can be expected with certainty in exploring a certain type of object” (34-35). Using this definition, Husserl spent more than twenty years attempting to build a comprehensive noema taxonomy for everyday objects only to find “he had to include more and more of what he called the ‘outer horizon,’ a subject’s total knowledge of the world” (Dreyfus 35). Husserl’s successor Martin Heidegger then added,

... since the outer horizon or background of cultural practices was the condition of the possibility of determining relevant facts and features and thus prerequisite for structuring the inner horizon, as long as the cultural context had not been clarified the proposed analysis of the inner horizon of the *noema* could not even claim progress. (Dreyfus 36)

What Dreyfus is saying here is that Minsky “has embarked on the same misguided ‘infinite task’ that eventually overwhelmed Husserl” (36) although Husserl was at least able to admit defeat.

Regardless of their utility as strong AI knowledge-representation tools, frames and scripts can be quite valuable in weak AI research and applications which need to have some means for storing and manipulating information and knowledge. Like finite state automata, frames are very

easy to implement using computational models. Frames translate easily to objects using object-oriented programming techniques. The object-oriented design methodology is “unique in that it is modeled after real-world objects” (McDaniel 9). In this type of model, a frame is represented using a class template, which is a pattern that allows new story objects to be created at will. Objects in general are data types with structure and state which are abstractions of real world objects (McDaniel 11). From a class template, many objects can be created that have the same placeholders (what Minsky would call “terminals”) but different data stored in these locations. This is precisely how the event-driven narrative analysis engine described in Chapter Five is implemented.

For example, consider the diagram shown in Figure 13. Here is a class template for a story object. Using this template, which is equivalent to a story frame, we can create many different stories using many different texts. We could create two such stories, one with an expected ending and one with a surprise ending, simply by changing the properties of each object as that object is created from the class template. In other words, new frames of representation can be created by filling in the blank terminals with specific information about each story.

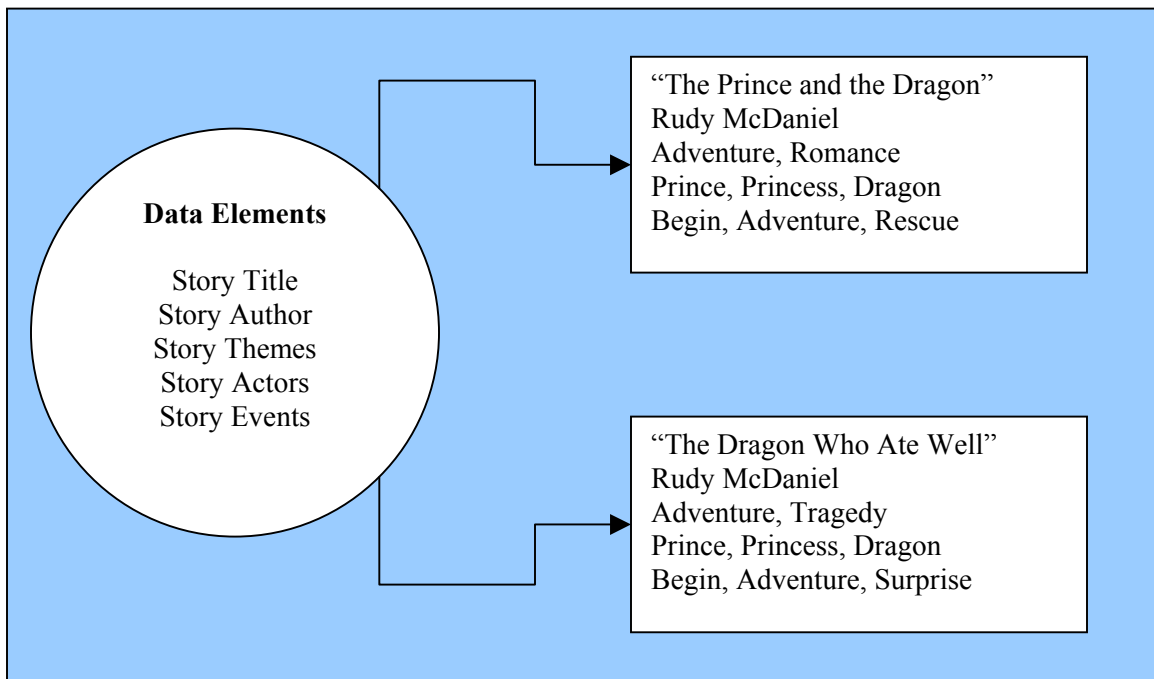


Figure 13: Frames Model for Representing a Story

If the story object’s class template is well-designed, it will have the appropriate data elements defined as well as the proper operations/methods in place to act on those elements. Then, using only this template, we can create thousands of new stories simply by calling new copies of type “story object” into computer memory using a software program. It is even possible to extend this base object through inheritance, which allows the creation of new story objects which adopt the parent object’s features but extend functionality through the creation of new data structures and methods. For instance, a narrative poetry object could be created which still uses the title, author, themes, actors, and events fields from the original object, but adds a new field for the number of stanzas in that particular poem. This new object could then be

created using syntax similar to the original object's creation. Figure 14 shows both the original "Story" class represented in pseudocode as well as the new "NarrativePoem" class template which would inherit from the original class. Both of these objects, when created from a class template and loaded into memory as active objects, are quite similar to Minsky's ideas about how our active memories are loaded from frames of reference. He believes memories are accessed by manipulating the various pieces of information stored in each terminal within our generalized memory-based frames.

```
Class Story {  
  
  Data Field Title;  
  Data Field Author;  
  Data Field Themes;  
  Data Field Events;  
  Data Field Actors;  
  
  Method Story (title)  
  {  
    Set This Object's Title = title;  
  }  
  
}
```

```
Class NarrativePoem extends Story {  
  
  Data Field NumberOfStanzas;  
  
  Method NarrativePoem (name,stanzas)  
  {  
    Set This Object's NumberOfStanzas = stanzas;  
    Set This Object's Title = name;  
  }  
  
}
```

Figure 14: Story and NarrativePoem Classes

The methods listed within each class in Figure 14 can be used to perform operations on the data stored in the objects that are loaded into memory. A special method, called a constructor, is called automatically when these objects are first created. So, if we want to create instances of the two stories shown in Figure 13, we can simply create the objects using syntax similar to that shown in the following example:

Story1 = new Story (“The Prince and the Dragon”);

Story2 = new Story (“The Dragon Who Ate Well”);

Or, if we want to create a new narrative poem object, we can do that with the following syntax:

NarrativePoem1 = new NarrativePoem (“The Odyssey”, 100);

This command will create a new narrative poem object entitled “The Odyssey” which contains 100 stanzas of verse. Note also here that the method is able to store data in the title field of that object, which is actually defined in the parent “Story” class and not in the inherited class. This demonstrates that the inherited class can still access data and functions from the parent class while also extending functionality by adding new data and methods. In this case a new data field is added to hold the number of stanzas and a new constructor (initialization function) is defined to automatically set the title and the number of stanzas when this object is created.

In these examples, new stories are created which automatically call the constructor method and pass in the string of quoted text inside each set of parentheses. Other methods would then be used to operate on the data within these Story objects. Although not defined in Figure 14, such methods might include operations such as `GetStoryTitle()`, `SetAuthor()`,

AnalyzeStoryForTheme(), LinkStory(), EditStory() and so on. The specific methods I used for the event-driven narrative analysis engine are presented in Chapter Five.

Minsky also created terminals specifically for the frames we reserve for stories. These terminals include items such as a time setting (when), a place setting (where), a protagonist (who), a central concern (what/why) and an antagonist (who or what) (265). Visualizing these terminals as blanks that can be filled in to represent particular stories is a helpful way of thinking about how these “story-frames” function (Minsky 265). A new “StoryFrame” object can also be inherited from the original “Story” class quite easily, and can be used to describe a story based on the original story items as well as these additional characteristics based on Minsky’s terminals. One possible class template for this new StoryFrame object is shown in Figure 15.

```
Class StoryFrame extends Story {  
  
  Data Field TimeSetting;  
  Data Field PlaceSetting;  
  Data Field Protagonist;  
  Data Field CentralConcern;  
  Data Field Antagonist;  
  
  Method StoryFrame (name,time_setting)  
  {  
    Set This Object's Title = name;  
    Set This Object's TimeSetting = time_setting;  
  }  
  
}
```

Figure 15: StoryFrame Inherited Class

Life Experiences and the *SparkleTech* Story

Donald Norman explains that even quantitative business decisions often are actually decided by personal life experiences. Norman writes,

I remember ... a meeting of senior executives at a major American company, decision makers inundated with statistics and facts. And then one of the highly placed decision makers spoke up: 'You know, my daughter came home the other day and ...,' and the story poured out. 'Hmm,' another executive said, 'that makes sense, but you know, the other day ...' and out came a story. A few stories, some discussion on the stories, and the decision. All these facts and statistics, and in the end, the decision was based upon some personal stories.

(129)

This incident Norman describes is not an isolated one. All across the world, decisions in industry and policy are often made solely on the exchange of stories between decision makers. Without valid life experiences to inject into these stories, they become dull and insubstantial; influencing a major decision at this point is unlikely.

When we hear or read a story, we are able to build associations from that story using our own life experiences. This allows us to relate to one another and if necessary respond to stories we hear with stories of our own in real-time, which is essentially the process of conversation. Sometimes we may even hear or read stories that do not actually connect with our own experiences until sometime later. In these instances, it is as though something finally clicks into place in our mind as we realize how the stories we previously heard or read intersect with the

new story we have just encountered. Using Minsky's theory, this can be explained as the process by which our personal experiences help to repopulate the terminals for that particular frame in our memory.

Consider the following scenario of how a story-based model might work in a distributed organization. The story explains an event which takes place in a typical networked organization. Chris, Sandra, and Jane, three employees of this organization, share their stories with one another over a period of time. One employee is then able to realize that these stories are related, and eventually is able to solve multiple problems by simply looking at the problem from a different perspective. By sharing stories verbally and then eventually distributing these stories throughout the organization using a printed newsletter, three individual problems are solved by one emergent property that is created from the employees' shared stories. This story, which I refer to as the *SparkleTech* story throughout this dissertation, is shown below.

Chris, a fictional computer technician, is an employee of the equally fictional company known as SparkleTech. As a result of a horrible automobile accident several years ago, Chris lost vision in one eye and is now forced to wear a rather ridiculous looking pair of eyeglasses in order to have any sense of depth perception whatsoever. The glasses have an overextended lens on one side and clear plastic on the other so a large frame is required to support the lenses on his head. The glasses, combined with his unruly curly and graying hair and his tendency to walk around with various tools and computer parts in his shirt pockets, have earned him the affectionate nickname "Mad Scientist" from his colleagues. Chris has spent several hours of the past few days (and has stayed late into the evening as well) attempting to troubleshoot a firewall problem with his company's Internet routers. The routers allow information to flow to and from SparkleTech's Intranet, which is the system of computers that allows all the various SparkleTech employees to exchange information and do their daily work. As a result of the firewall problem, all SparkleTech employees can reach their internal network resources but nothing on the outside network, which includes the Internet.

Sandra, a manager in Chris' department, has been under pressure from her own boss, who is the regional vice president of SparkleTech's southern division where she is based. Sandra, an attractive brunette, was at one point romantically involved with Chris. This makes certain duties

of her position uncomfortable to her. Having worked with Chris for three years now, she knows that if there were an easy solution to the router problem Chris would have found it days ago. She is running out of options, though, especially since the Lang contract is nearing deadline and without a working link to the Internet they have no way to demonstrate their progress to the client. She has the feeling a confrontation with Chris is inevitable.

Jane is a mid-level technical writer working in the publication division of SparkleTech. Jane is an avid runner and recently completed her first triathlon event on her 32nd birthday. Among Jane's various responsibilities, including managing and overseeing software documentation as well as coordinating website content, Jane enjoys creating the SparkleTech newsletter the most. On this particular day, the newsletter content seems to be all bad news. A major client is likely going to cancel services due to the inability of the Southern division to meet their deadline. Hoping to find at least some good news in the midst of this disarray, Jane decides to go out and interview the two employees involved, who are of course Chris and Sandra.

She goes down to the third floor and back into the network area. Smiling to Carl, the receptionist, she walks back to find Chris hunkered down in front of a monitor.

"Hello, Chris."

Chris adjusts his eyepatch and turns from the monitor. "Hi, ahh, Jane.. look, it's a bit of a bad time here."

"Yeah I know, I was just hoping to get a quick interview with you for the newsletter."

"Can we do this later? Maybe tomorrow?"

"Sorry, I have to get this out today. Can you just give me five minutes?"

Chris glances at his watch, then back at the server array in disgust. "Okay, five minutes more isn't going to make much of a difference I guess. Let's go."

In the next five minutes, which turns out to be more like half an hour, Chris explains the whole saga of the network router and the firewall problems. It turns out that an enormous amount of traffic has been funneling through their router somehow, so all of SparkleTech's hardware resources are hard at work scanning for security breaches and virus activity. With the hardware so busy with his scanning process, there is not enough bandwidth left over for normal Internet activities. Chris is convinced it is a hardware error since they have never seen close to this much activity on their network. As a result, he has been randomly swapping out switches, routers, cables, and network cards hoping to find the faulty part that would allow him to restore normal activities.

"Wow, that's quite a problem. You're sure it's hardware?" Jane asked.

"It has to be. Nothing else makes sense. Look, I have to get back..."

"Of course," and then more halfheartedly, "Good luck."

Strangely enough, it was much more difficult to get an appointment with Sandra, the department manager. Jane went through several layers of administration trying to get a five minute block only to have it rescheduled twice and almost cancelled. Finally, she was able to get an early afternoon appointment after she promised she would not need more than five minutes.

"Hi Sandra," Jane offered, walking into the large office. She didn't know Sandra very well, but she seemed to be a nice enough person. At the moment she seemed frazzled and frantic.

"Jane, Hi. Sit down, please, have a seat, can I get you anything? Water? Coffee?"

“No thanks. Listen, I won’t take much of your time. I just wanted to get a quick quote from you about the Lang contract. I was hoping to get some good news but I just met with Chris and it is actually worse than I thought. So, anything you have for me I am willing to use at this point.”

Sandra looked at her for a few minutes before answering. “Listen, can I tell you something off the record?”

“Sure, I guess.” Jane tapped her fingers against her notebook in frustration. She was anticipating being blown off again.

Sandra gave her a strange look then began talking. “It turns out things are worse than we thought. NotoTech has apparently been sending in software agents to try and steal our project notes and designs. They are going to try and use our own technology to outbid us on the Lang contract.”

Jane’s eyes widened. “Did they get anything?”

“We don’t know yet. Listen, no one knows about this, so it sure can’t go in the newsletter. I am going down to tell Chris now. Can you hold off on sending this out one more day?”

Jane agreed without hesitation. At this point she was happy to let this project slide another day.

The next day things were tense at SparkleTech. Rumors were circulating that the Lang contract had not gone through as planned, and to make matters even more suspicious the company newsletter had not gone out on its usual day. Even Carl was less than his normal buoyant self as he anxiously routed technical support calls to the appropriate technicians and waited for Chris to arrive. He was usually in by nine o’clock so his tardiness didn’t do much to improve the atmosphere.

At quarter to ten something even more unusual happened. Someone finally walked through the door, but it wasn’t Chris.

Jane walked through the door, smiling triumphantly and holding up a sheet of heavy paper – the familiar green and blue stock used to print the company newsletter. This particular newsletter was bare save for a bold headline across the upper margin.

“SPARKLETECH FOILS CORPORATE ESPIONAGE, LANG CONTRACT RENEWED,” Carl read. *Well that sounds promising*, he thought.

Ten minutes later, Chris had arrived and after a short discussion they both headed up to Sandra’s office. Ten minutes after that, all three of them exchanged stories, and what was previously a disaster was now actually looking quite good.

What had happened was this: rival company NotoTech had indeed planted a malicious software program inside SparkleTech’s internal computer network. How they were able to install the software was still a mystery, since the installation would require access to a computer inside their building, but that was a detail that could be investigated later. What was important was that the flaw Chris had found in the network was not actually a flaw at all; the network was

simply overloaded trying to block all of the outgoing traffic that the agent was trying to send out to NotoTech's own computers on the outside of their corporate intranet.

After hearing Sandra's confidential aside, Jane eventually connected this new development with her previous conversation with Chris. Once she considered their stories side-by-side, she was able to figure out that what they had initially considered as a problem had actually saved their technology from being stolen by a rival company. She had called Sandra at home that evening, and Sandra was able to finalize the Lang contract after describing the situation to a few executives over a conference call and explaining just how impressively SparkleTech's security was able to perform when blocking illegitimate network traffic. Chris, finally convinced that the problem was not hardware related, was able to analyze the server logs and easily shut down the infected computer once he knew what he was looking for.

Later that afternoon, Jane had finished the article to accompany her headline and the company newsletter was sent out to all SparkleTech employees by email. The network, now restored and fully functioning, delivered the publication quickly and news soon spread that the contract was now in place. Sandra was praised by the regional vice president, Chris was given a new gold-plated eyepatch in recognition of his solid security practices, and Jane started training for her next marathon that next week as a senior technical writer; her contributions to solving these problems had not gone unnoticed. For the time being, all was well again at SparkleTech.

The events in the *SparkleTech* story, although somewhat streamlined for the sake of brevity, could reflect events in any large organization with various layers of management and segregated departments. Having an open policy for employee interaction and story exchange, or at least some type of forum for this type of discussion at employee meetings, helps to connect knowledge specialists with one another. In addition, this type of discourse can smooth transitions between the end of one large project and the beginning of another. This is only one such example of what a storytelling model can contribute to a corporate organization.

Metaphor and Analogy

If we were to be tested on our recall of the SparkleTech story next week, which details would we remember most clearly? More than likely it would not be the various job responsibilities of Jane or the technical details of the router problem. It would probably be the personal details about each employee that may have triggered some sort of memory of our own having to do with their fictional lives. For example, we may have remembered that Chris lost an eye because we remembered an old movie we had seen with a mad scientist as a principal character in that movie. Or we may have remembered that Sandra and Chris were once romantically involved because we had a disagreement with our own significant other the day we read the story. Or maybe we remembered the names of both female employees because we personally know two women with the same names. Of course there are many other minor details that we could have remembered from the story, but the point here is that the interesting details are the ones that usually stick around the longest in our memory. If these seemingly non-relevant details were the pieces of information we remembered, then we would have relied on a critical property of a narrative knowledge management system, which is the ability of a narrative to anchor recollection indexes to our own experiences through metaphor.

Todd Post writes that the power of metaphor in storytelling is due to the fact that stories allow people to understand complex issues and topics in language they can understand and relate to. He writes, “Metaphors allow an entry point into an issue that might seem too intimidating to confront head on. It’s the idea of looking at things on the slant, as Emily Dickinson liked to say of poetry’s power to unlock the secrets of the heart. Metaphors displace context just enough so

that we can look at a problem, an issue, or a situation that may apply to work, seeing it in its distilled essence” (28). It is this distilled essence that is of primary importance in a narrative knowledge management model, especially one concerned with capturing tacit as well as explicit knowledge. Using metaphorical language within stories is like a narrative vernacular; it allows access to complex and specialized concepts that simply would not be understood by listing engineering or scientific specifications in conventional formats.

The ease at which people relate to metaphors in stories is not lost on capitalist America. Media is constantly recycling stories throughout our communities, and it is hard to travel, turn on the television, or listen to the radio without hearing stories about how our lives would be better if only we purchased the product being advertised to us. Even public service announcements and advertisements often follow a narrative approach, warning of the dangers of a particular substance or product by showing us how it ruined a life in the past. Here stories become metaphors for real, tangible products – we simply replace the human agents in the stories we hear with ourselves and we either want to do everything in our power to make that scenario happen or else throw all our available resources into keeping it as far away as possible.

Analogies are equally important in narrative prose. Hubert Dreyfus writes of three basic ways to think about the human capacity for using analogies. The first, which comes from Aristotle and classical rationalist tradition, is to “understand analogies as proportions” (xxiv). The second is related to our experience with our bodies, and the third reacts “to the implausibility of the classical tradition by approaching analogy in terms of extrapolating a style” (xxv). In each of these approaches, there is a definitive humanistic aspect which requires a degree of personal subjectivity in order for that approach to make any sense. Computers can

certainly understand literal proportions, but figurative proportions based on humor or other pathos-driven analogies will require an understanding of common sense which computers today do not possess. The second body-centered definition certainly doesn't apply to computers, and a computerized or mechanical style is more accurately described as a lack of style than as a style. In other words, the capacity to manipulate and use analogies and figurative language is another ability possessed by humans that computers can not currently draw upon for their own knowledge-management processing tasks.

Scope and Domain

Because a narrative model is so open to metaphorical constructs, it is not tied down to any single domain. As a result, it can be an exhaustive process to try and narrow down even a single story into one specific category. If given an open-ended invitation to write a story, a randomly selected group of people will come up with a very different group of stories, even if they are given a topic to write about. As an example of this tendency towards diversity, consider the World Wide Web's remarkable assortment of stories. We can consider the Web to be a giant collection of stories, with links into other stories and other experiences scattered throughout. Many stories, though, will take on distinctly different characteristics even when describing the same basic experiences or ideas. Personal home pages, for example, represent one such range of stories with a notably wide breadth. Some people choose to represent themselves with a picture and a background on their webpage. Others instead list professional and academic achievements. Some are more creative and display poetry or even fiction on their website. Due

to the highly personal nature of storytelling, then, it becomes a challenge to attempt to classify stories based on topic alone. My approach to this challenge was to use thematic dictionaries encoded using XML. Details about these dictionaries are provided in Chapter Five.

Despite the challenge of such a topical classification, Roger Schank has managed to define five types of stories based on the origin of these stories. Schank's story categories include "official stories," "invented or adapted stories," "firsthand experiential stories," "secondhand stories," and "culturally common stories" (30). Each of these stories may of course overlap, or there may be blending between two or more categories. For the most part, though, these categories can be applied to stories to determine which of these types works best to describe a given narrative within the overall taxonomy.

Official stories are "those we learn from an official place, such as a school or church or a business or from the government" (Schank 30). These stories might also be called ideological or societal stories, and they are propagated through complex mechanisms and systems of order and control. Telling an official story to someone gives them a quick idea as to our feelings about politics, religion, and social issues, so these types of stories can often be more controversial and perhaps more repressed than other types of stories. They are also very powerful since they influence how we react to and interact with groups of people who believe in the same or different types of official stories.

The invented story, which can also be an official story, "expands upon an experience for the purpose of entertainment, sometimes leaving the original story unrecognizable in the process" (Schank 32). Invented stories can be highly creative and original, but can be dangerous

if they are used for teaching purposes and have not been grounded in proper facts or given accurate background details.

Firsthand experiential stories “represent our own personal experiences,” some of which “have been digested and analyzed so that we know their point or points and are reminded of them when these points come up” (Schank 36). In the *SparkleTech* story, Jane was able to synthesize two such firsthand stories and figure out a solution to a complex problem by analyzing the points of these stories and determining where they intersected.

Secondhand stories are “simply the firsthand stories of others that we have heard and remembered” (Schank 37). Referring once more to the *SparkleTech* story, Jane was able to transform two firsthand experiential stories into secondhand stories when she edited them and included condensed versions in the company newsletter.

Finally, culturally common stories are those culturally-authored stories that come from our environment (Schank 37). Culturally common stories can be manifested in both desirable and undesirable forms; the examples of gender and racial stereotypes discussed in Chapter Two are generally considered to be undesirable forms of culturally common stories.

The narrative analysis engine presented in Chapter Five thrives on the stories Schank describes as firsthand experiential stories when functioning to manage tacit knowledge. Secondhand stories may also work in this context, although these types of stories are likely not as effective as the stories an expert with tacit knowledge in some field has personally experienced.

Computers and Hypertext

Now that we have considered a narrative approach to managing information, examined the properties of such a system, and read through an example story, we can consider the idea of how to make such a system practical and flexible using a computer. Computers have long since been recognized as powerful tools for extending humankind's reach into all types of sophisticated domains. Computers have allowed us to refine our scientific theories, extend our productivity in the workplace, expand our visions of multimedia through entertainment using image processing applications, and do many other things that were formerly too detailed or complex for us to consider alone. It is only natural that a computer should help us to collect and manage our informational stories. Wick describes such an approach to knowledge management as the "technological perspective" (517).

There are numerous properties of computers that make them ideally suited as narrative machines. Some of their more important properties include their ability to process and store symbolic information, their random access capabilities, and their computational speed. When combined, these properties allow a computer to fetch any one of thousands of relevant stories about a particular topic from memory within a few milliseconds. When compared to the human brain this still may seem slow, but it is worth mentioning that a computer will not experience the side-effects of human memory which will limit retrieval for a person. Side-effects include psychological buffers such as repressed memories, false memories, or even forgotten memories. With a computer, if an index exists in memory it can be searched for and located every time.

Hypertext is the obvious domain in which storytelling will thrive, especially if extended by the semantic markup of XML. XML was described in detail in the Introduction, and hypertext is briefly described here. This markup language emerged as a concept in 1990 by Tim Berners-Lee and was released by CERN, Geneva's high energy physics lab, that same year (Kurzweil 276). Just ten years later, the World Wide Web, a global community linked by hypertext, was ubiquitous. A hypertext community has the global reach necessary to integrate stories from all walks of life into a searchable and organized store of knowledge. Peter Lunenfeld writes, "... regardless of the hype about hypertext, there are still very few examples that fully demonstrate the power of storytelling, and thus of authorship, in the new medium" (141). Hypertext is not without its share of problems, however. There are now issues of reputability and ethical soundness to consider, as well as the very real potential for information overload or infoglut. There is a combination that will work, though, which is to use a restricted domain model to store knowledge related to a particular organization and categorize it by events and personalized reactions to that event. Such a model can then be used to extend the HTML framework to also accept XML-encoded stories using a system similar to that described in the Introduction. This script based model is precisely what is created and explained in Chapter Five.

Intelligence Revisited

Now that artificial intelligence has been discussed in detail in Chapter Two, and a narrative system has been defined in this current chapter, we can decide whether or not our computer needs to be "intelligent" in order to be our storytelling control mechanism. For this

discussion it is necessary to return briefly to some of the ideas and concepts presented in the first chapter of this work. In their article considering the current state of machine intelligence, Ian Brackenbury and Yael Ravin propose several different areas in which a machine should be well-versed. These include the ability to recognize and process “declarative statements, linguistic knowledge, procedural knowledge, naïve physics, recognition knowledge, tactile/kinesthetic knowledge, visual knowledge, and social knowledge” (526).

Of these eight categories identified by Brackenbury and Ravin, only three are of vital importance to our storytelling computer, with perhaps a slight reliance on a fourth. A storytelling computer has no need to walk around from person to person telling stories. Although such a device might be entertaining, it is much simpler and just as effective for us to go to the machine for our stories, especially considering the ubiquity of the Internet and modern wireless network computing. So, there is no need for our storytelling agent to have knowledge of naïve physics or tactile/kinesthetic knowledge since it will not be moving around in the physical world. Visual knowledge is also not important since our computer does not need to see out into the world; all knowledge for the computer comes from the stories that are collected and stored in a database.

Knowledge of these other domains will allow our narrative computers to work in a way that makes them appear to be somewhat intelligent. Modern computers have in fact mastered many of these areas and have been described as intelligent in regards to very specific applications. Ray Kurzweil writes, “Computers today exceed human intelligence in a broad variety of intelligent yet narrow domains such as playing chess, diagnosing certain medical conditions, buying and selling stocks, and guiding cruise missiles. Yet human intelligence

overall remains far more supple and flexible” (2). It is this flexibility and potential for creative insight that has led many philosophers such as John Searle to claim true human-like intelligence in computing is impossible.

Since our computer does not need to possess the myriad of intelligent abilities proposed by researchers such as Brackenbury and Ravin, we can consider our storytelling computer to be semi-intelligent, weakly intelligent, or clever. Critics of rule-based natural language programs (such as the chatterbots discussed in Chapter Two) have long argued that what appeared to be intelligent behavior was in fact merely clever programming by computer scientists using a giant collection of rules and a computer that simply narrowed down the field of acceptable information until it found the most likely response. This type of system does not even require the most cutting edge hardware; the famous script-base chatterbot *ELIZA*, for example, was created using technology of the 1960s (Weizenbaum 2-3). With this in mind, it is worth considering whether or not a storytelling computerized model needs to possess intelligence at any level of its operation.

The most important idea to remember when trying to apply the concept of intelligence to a computerized narrative model is that we can not even begin to consider the system as being intelligent unless we consider it in its entirety. Intelligence here is simply “a combination of simpler things” which are themselves not intelligent (Minsky 23). In other words, the individual stories themselves are not semi-intelligent or clever, but only the system as a whole. We are again relying on both emergent properties and the social dimension of intelligence and knowledge to help catalyze connections between stories and written conversations.

A final measure of intelligence in our computerized narrative mechanism can be found in our system's ability to converse with users in the fashion of *ELIZA* or *ALICE*. Traditionally, as demonstrated by the Turing Test, conversational ability has been looked upon as a measure of intelligence (Schank 17). Indeed, when we first consider a human to be intelligent or smart, it is usually because of an engaging talk or conversation we had with that particular person. So we could apply this same operational definition to our computer system. If we feed in a story to the computer and an appropriately themed story is returned then we can argue that there is some degree of intelligence in the computer, in the sense of the "weak" artificial intelligence described in Chapter Two. Regardless of the numerous problems with the lack of common sense and flexible creativity in computing technologies, using the weak AI definition suggests that a narrative computer system can be somewhat intelligent simply by arranging and returning stories in an appropriate manner. Some computer-based conversational and storytelling systems are considered more in depth in Chapter Four, while a specific mechanism for framing and returning stories is the primary focus of Chapter Five.

A New Domain for Technical Communicators

If storytelling becomes an accepted and widespread tool for disseminating organizational knowledge, then the implications for the field of Technical Communication are numerous. Allowing trained writers and creative thinkers to be involved in the knowledge-shaping process within their respective occupations not only increases the ease at which information is shared from one division to the next but also solidifies their own importance within their organizations.

As Todd Post reminds us, “Plenty of people have important stories to tell, yet lack the facility to tell it in a way that highlights what’s most important ... it’s naïve to think you can turn anyone loose to tell a story who has no experience at it and expect a coherent narrative to flow out in a natural way. People, after all, get advanced degrees in writing stories” (29). It is only natural to have creative writers, literary theorists (who in their natural studies are essentially acting as narrative analysts), T&T practitioners, and technical communicators involved in the story creation and dissemination process. Having a knowledge of technology and a rhetorical understanding of communication and communication theory is certainly helpful; technical communicators routinely use such knowledge to “transform tacit knowledge into explicit artifacts so that it can be accessed by others within the organization” (Hughes 280).

Technical Communication and Texts and Technology professionals may also fill an important gap in the job market of our current knowledge economy as noted by Egbert De Weert. De Weert writes about the difficulties in finding adequate knowledge management professionals with the educational training necessary to fill the requirements of these types of positions. He explains, “... several studies presuppose a one-to-one relationship between education and job where educational types are considered as boxes that fit into occupational boxes. It is very questionable whether this still holds true, as the relationship tends to be quite elusive in a knowledge society in which jobs are less clearly defined and structured” (65-66). De Weert fails to consider both T&T researchers and technical communicators, though, who are trained to deal with these types of ambiguities and dynamic situations by learning about specific technologies (such as single-sourcing) and theories (such as audience analysis) which require learning about the structure and manipulation of data.

With this in mind, textual technologists and technical communicators seem to be the obvious choice for constructing knowledge management systems with narrative components. Corey Wick identifies three critical skills possessed by technical communicators that make them ideally suited for knowledge-management positions in general. He writes,

I believe technical communicators have three core competencies that will help them to position themselves as the forefront of their companies' knowledge-management efforts and at the forefront of the emerging discipline of knowledge management.

1. They have a thorough understanding of the complexities of knowledge, language, and communication.
2. They are exceptionally talented in working across functions, departments, and disciplines.
3. Most importantly, they are *expert* communicators. (524)

As Wick explains, technical communicators are trained to critique information; they can quickly and accurately assess the nature and purpose of a document and determine whether or not this document will be effective for its intended audience. While storytelling itself is admittedly more suitable for other professionals such as creative writers and novelists, the use of stories as an organizational knowledge-sharing tool makes such a medium well-suited to take advantage of the expertise and training of technical writers and communicators. These technical communication professionals are also well-aware of the specific advantages and disadvantages involved with using different types of media for different KM tasks. Wick elaborates, "... technical communicators can supplement ..." their core competencies "... with a thorough

understanding of which media and technology to employ for which types of knowledge for which purpose” (525). If a computerized narrative analysis model is the best technology for disseminating knowledge in a given organization, then a technical communicator within that organization will likely possess the academic background, the professional training, and the innate talent necessary to create, populate, and disseminate stories encapsulated in that system.

The need for technical communicators to take a more active role in their organizations as knowledge engineers is well-documented in the literature. In one such article, Michael Hughes explained two ways in which technical communicators have added value to an organization by taking an active role in knowledge management. He suggests that technical communicators can increase the value of their profession by becoming the creators of knowledge rather than simply the transmitters.

Among the key ideas introduced by Hughes are positivist versus constructivist views of technical communication, the difference between knowledge and information, explicit and tacit forms of knowledge, organizational, group, and individual scales of knowledge, and specific ways that technical communicators can become creators of knowledge (275-284). Hughes explains that one of the most important functions of the technical communicator is to transform tacit knowledge into explicit knowledge, and explains two techniques for doing this: probing interviews and critical reverse engineering (281). Finally, he discusses some ways in which technical communicators can propagate individual knowledge upwards through the pyramidal scales of knowledge to reach the organizational level (283). Hughes writes that technical communicators are “increasing return on investment” either “directly, through improving the quality of the output and, thus, competitiveness in the marketplace” or “indirectly, through

reducing the internal costs of development, production, and support” (276). He explains that technical communicators are able to accomplish such a return on investment through their creation of these knowledge assets, which are exactly what is represented by the stories in a narrative system.

J.D. Appen's article, much like the articles from Hughes and Wick mentioned previously, is concerned with defining (or redefining) technical communicators as knowledge managers within their organizations. He explains some of Kuhn's scientific theories and paradigms that have influenced knowledge management in science from a constructivist perspective, and describes the competitive rivalry and the social constructionism involved in the work of biologists Charles Child and Thomas Morgan (303-304). He also mentions specific examples of knowledge management techniques within such companies as Arthur Anderson, Microsoft, and Buckman Labs (306). Other useful information in this article includes the four types of organizational knowledge (declarative, procedural, causal, and relational) and the five major components for "building and maintaining a knowledge map" as used by Microsoft (306). The article concludes with an overview of XML and a discussion of how this markup language relates to knowledge management scenarios, especially as used by technical communicators (307-312).

In the next chapter, specific applied examples of narrative systems will be discussed. These examples will span from oral stories recounted in speeches to the virtual storytellers living in computer software. Such models will show that a storytelling approach has been both a successful and natural way for members of our society to organize information and propagate

knowledge for many centuries. As technologies change and evolve, the story is inevitably remediating into a new form that is suitable for the current technologies of the time.

CHAPTER FOUR: EXAMPLES OF NARRATIVE SYSTEMS

“Stories tell us of what we already knew and forgot,
and remind us of what we haven’t yet imagined.” – Anne L. Watson

Background

The purpose of this chapter is to explore various types of storytelling systems used to shape rhetoric and manage information and knowledge in Western culture. While a multicultural analysis would also be quite interesting, such research is outside the scope of this dissertation, although this area of study is certainly a legitimate subject for future research in storytelling and narrative knowledge management mechanisms. My primary task in this chapter is to explain oral, computerized, and managerial forms of narrative examples and to gather additional theories from these examples that I will later use in my own model in Chapter Five.

The chapter begins by considering the most ancient forms of stories as they existed in pre-literate society. These stories, along with the stories embedded in speeches, make up part of what I consider to be traditional oral storytelling, or the types of stories a group might trade around a campfire. Excerpts of speeches from Booker T. Washington, Ronald Reagan, and Jesse Jackson are given to illustrate some of the rhetorical/persuasive properties of stories when used in public speeches. Written stories are also mentioned but not expanded upon; the subsequent examples given in this chapter all incorporate written stories to one degree or another and the

written story is therefore considered in terms of its remediation through computer technologies and its role as a foundational model for these other types of narrative.

The next section of this chapter is devoted to some of the most famous computer storytelling systems: TALE-SPIN, BRUTUS, MINSTREL, and MEXICA. These systems are important because they reveal how computer scientists in the past have attempted to deal with representing narrative theories and stories using computer processing techniques and data structures. Technical programming techniques used for story knowledge-representation and story grammars are discussed, as are narrative analysis techniques used for determining creativity and novelty in computer-based stories. Example stories from each of these storytelling systems are also provided.

Two organizational and institutional storytelling systems are explained later in this chapter; these will be the most useful when building the narrative analysis engine in Chapter Five. Two storytelling researchers in particular, Stephen Denning and Graham Smart, have written detailed studies in which they analyze the use of storytelling as a knowledge management tool in large banking organizations. These studies are summarized in this chapter. In Denning's book, he explains several characteristic features that make up what he describes as the "springboard story," which is a type of story that is particularly useful for propagating and managing knowledge. Smart's article, about a similar system used in the Bank of Canada, describes the gradual process of building the *monetary-policy story* using a merger of a computerized expert system with human expertise and knowledge. Smart describes the resulting story as "a rhetorical resource for communicating the bank's monetary policy" (267). The banking industry has both recognized and embraced the power of narrative systems, and has used

these systems to assist in conveying specialized tacit knowledge. Smart explains, “Over time, their professional community has developed a tacit understanding—a historically achieved institutional wisdom—about how an organizational narrative, or, more accurately, a metanarrative, provides a way of combining the differentiated expertise and analytic work of many people in a configuration that allows for the construction and application of the complex specialized knowledge the bank needs to conduct monetary policy” (269). In other words, the expert knowledge needed to perform specialized banking tasks can be embedded and distributed through stories in a narrative knowledge management system.

The chapter closes by examining a few examples of storytelling using images, or visual storytelling. The premier theorists in this genre, Will Eisner and Scott McCloud, offer definitions of comics and graphical storytelling. These theories are bolstered with a bit of visual social semiotic theory from researcher Claire Harrison. A brief discussion of Hans-Christian Christiansen’s theoretical work comparing comic media to film media is also included to illustrate some of the advantages of the visual narrative over its written counterparts. Considering both practical models and these theoretical arguments will help to illustrate those particularly useful characteristics of narrative systems that I will incorporate into my software model in the next chapter.

Oral and Written Storytelling

The most obvious examples of narrative systems are also the simplest. We have been using stories, both oral and written, for many centuries to assist in communication and

memorization. Aside from their recreational and entertainment value, spoken and printed stories have long served as important tools for managing information.

One compelling piece of evidence suggesting storytelling is a superior technique for managing information comes from observing the public speaker. The art of public speaking – Walter Ong’s *technē* or “speech art” – has historically been considered the purest and truest form of communication. Public speaking also introduced the study of rhetoric as a field in academia (Ong 9). It is no accident that the greatest orators of past and present were able to inspire action and demand attention by weaving stories and personal experiences into their speeches. Speaking in front of an audience, though, is by nature a more precise and unforgiving activity than writing ideas down on paper. For one thing, there is no way for us to easily go back and erase or correct irrelevant or distracting ideas that have somehow entered into our narratives. Instead, we must attempt to integrate these ideas into the themes of our speeches and hope the audience doesn’t catch on to what was in essence a mistake in the weaving of our thoughts. In a speech, these mistakes must be ironed over or smoothed instead of removed or edited as we can do with a written product.

Secondly, oration requires an audience to remain seated and to listen to our ideas at a particular time and in a particular place. In a live assembly, there is no way for the audience to pause the orator’s speech and resume listening to it at a more opportune time. Because of these two very demanding limitations, speakers know they need some sort of powerful method for both establishing an audience’s attention and then keeping that attention fixated throughout their speech. Visual aids such as models, charts, and unusual physical objects can be one such external mechanism for controlling audience attention and helping to remind an orator of

transitions within their speech. Storytelling, however, is another powerful tool that does not require anything external in order to function. A story, especially one full of metaphor and figurative language, can provide an alternate format for an idea or concept in a speech while also addressing the limitations described above.

A third restriction on oral delivery, while perhaps less problematic than those mentioned above but still quite present, is that orators should remain speaking throughout their delivery without unnecessary or awkward pauses in their delivery. Ong explains, "... it is better to repeat something, artfully if possible, rather than simply to stop speaking while fishing for the next idea. Oral cultures encourage fluency, fulsomeness, volubility. Rhetoricians call this *copia*" (41). He continues to explain how this trend towards redundancy continued into early forms of writing, often in an annoying and unnecessary fashion. Stories, though, can function to not only remind one of the linkages between themes and ideas in a speech, but they can also act as a sort of safety net to rely on in case the primary linkage is forgotten in this speech. Narrative *copia* can thus be explained as the repeating of a concept or idea by demonstrating or elucidating that idea in the form of a story. If a speaker forgets what they have planned to discuss next in their speech then they can simply rely on one of several backup stories they have constructed in advance.

Storytelling in speeches exists in abundance. One famous example from American history is found in Booker T. Washington's speech to the Cotton State Expedition in 1895. Early in this speech Washington explains the race relations between Southern Caucasian and African American citizens in industry by comparing this relationship to a ship lost at sea. Washington tells the following story, shown in Figure 16.

A ship lost at sea for many days suddenly sighted a friendly vessel. From the mast of the unfortunate vessel was seen a signal, "Water, water; we die of thirst!" The answer from the friendly vessel at once came back, "Cast down your bucket where you are." A second time the signal, "Water, water; send us water!" ran up from the distressed vessel, and was answered, "Cast down your bucket where you are." And a third and fourth signal for water was answered, "Cast down your bucket where you are." The captain of the distressed vessel, at last heeding the injunction, cast down his bucket, and it came up full of fresh, sparkling water from the mouth of the Amazon River.

To those of my race who depend on bettering their condition in a foreign land or who underestimate the importance of cultivating friendly relations with the Southern white man, who is their next-door neighbor, I would say: "Cast down your bucket where you are" -- cast it down in making friends in every manly way of the people of all races by whom we are surrounded.

Figure 16: Excerpt from Booker T. Washington's 1895 Cotton State Expedition Speech

Later in the speech, Washington encourages both Caucasian and African Americans to “cast down their buckets” in order to promote both patience and conformity within the current system of Southern industry. Here Washington introduced a story early in the speech and then used a theme from that story to shape and develop his later arguments.

It is also worth pointing out that a story does not need to be lengthy, nor does it need to be particularly well-developed, to work well in a speech. Sometimes a story works at a pathetic level to fire up an audience’s emotions and focus their attention. Consider Jesse Jackson’s 1984 campaign for president as shown in Figure 17. Jackson opens his speech by recounting his experience witnessing his father’s return from war as a small child.

There are eight Democrats running for the nomination this time around: You got a chance, and you got a choice.

You want somebody who marched for you to get the right to vote, you got a choice. Want somebody who challenged corporate America to hire and give you contracts, you got a choice. It's choice time! [Loud applause.]

All this talk about qualifications..."What do blacks know about foreign policy?" It's an insult. I was three years old, I came into my consciousness, my Daddy was coming home from the war. Foreign policy. If he was so dumb, how did he get over there and get back? If he didn't know foreign policy, why did they give him a gun? And when they gave it to him he knew which way to shoot. [Cheers and laughter.]

We know foreign policy. When you buy Honda and Toyota, that's foreign policy. Russian Vodka, that's foreign policy. Panasonic and Sony, that's foreign policy. Mercedes Benz, that's foreign policy, and a matter of fact, we came here on a foreign policy!

Yeah! [extended applause and cheers]

Figure 17: Excerpt from Jesse Jackson's 1984 Campaign Speech

Here Jackson is able to take a concept as complicated (and seemingly boring) as foreign policy and use it to rouse the crowd into a receptive mood for the rest of his address. The story is simple and straightforward but it is something everyone in the audience can likely identify with: a point in childhood at which they were still proud of, and not yet embarrassed by, their parents. While it may not have done much to explain his specific ideas and plans for foreign policy, it certainly anchored some enthusiastic support for his speech. At the same time, his story touched upon this important topic in the news and media.

One last example of storytelling in oration demonstrates another property of narrative – its ability to inspire hope and to comfort in times of great loss or tragedy. In his remarks on the

Challenger explosion spoken in 1986, then President Reagan closed his speech with the following parallel narrative detailed in Figure 18.

There's a coincidence today. On this day 390 years ago, the great explorer Sir Francis Drake died aboard ship off the coast of Panama. In his lifetime, the great frontiers were the oceans, and an historian later said, "He lived by the sea, died on it, and was buried in it." Well, today we can say of the Challenger crew: Their dedication was, like Drake's, complete.

The crew of the space shuttle Challenger honored us by the manner in which they lived their lives. We will never forget them, nor the last time we saw them, this morning, as they prepared for their journey and waved goodbye, and "slipped the surly bonds of earth" to "touch the face of God."

Figure 18: Excerpt from Reagan's Challenger Speech

The simple historical account of Sir Francis Drake was more than a fact gleaned from a history book. Here it is a story that reveals the Challenger astronauts as great explorers and adventurers and equates their deaths with the passing of another great adventurer from the past. Reagan characterizes these astronauts as having as the same unfailing dedication and determination as Sir Francis Drake. Such a testament is fitting for the nature of our language and especially speech, which as Ong explains tends to "grow from the unconscious" rather than from the conscious (7). By creating a story which seems to be about sadness but is actually about the triumph of man over a natural boundary, Reagan delivers an appropriate amount of sympathy while at the same time guarding the interests of the space program and future manned missions into space. One interpretation of this speech might be to unconsciously measure up Drake's passing against all of his achievements, which would then lead one to consider the Challenger incident in the same fashion. Yes the Challenger disaster was a horrible tragedy, but look at

everything the space program has accomplished in the past! Without a narrative account somewhere in his speech, such a rhetorical accomplishment would be more difficult to achieve.

Written stories are also often vessels for storing and communicating complex ideas. We need only to look at the seemingly immeasurable amounts of nonfiction literature available to observe the impact of the written story on our culture. Vast collections of stories have been used to educate on topics from history to engineering, often using first person accounts of one individual's experiences with these subjects.

Written stories can be very efficient in helping one to learn unfamiliar material. A visit to a local bookstore will reveal several story-based collections that explore topics such as mathematics, physics, or engineering while at the same time recounting stories about the key players of these figures during their pioneering days of invention. Because of the vast amounts of narrative poetry and prose available, including everything from children's literature to epic poetry, examples from this category of narratives are not included in this dissertation. Instead, it is more appropriate for me to move my discussion into computerized storytelling systems.

Computer-Based Storytelling Systems

Some Artificial Intelligence researchers such as Roger Schank consider a computer that can tell appropriate stories to be an intelligent machine. Rather than simply collecting and organizing information using data structures and databases, a storytelling computer has some means for acquiring and storing knowledge as well as generating new knowledge from the knowledge it already has. As a result, there have been many attempts, some quite impressive, in

which computer science practitioners have tried to build a storytelling computer. Successful models have been researched and designed throughout the last thirty years of AI research.

TALE-SPIN

One early serious attempt at a computer-based storytelling system was James Meehan's program TALE-SPIN, created in 1976. TALE-SPIN is a program that was created in order to determine the types of knowledge necessary to create a story (Meehan 197). The program operates as a simple simulation with a small group of characters who are given a particular problem to solve by the computer. Much like modern complex software that relies on events to transition from one state of operation to another, TALE-SPIN also uses events to determine when the text for a story should be generated and displayed to the screen. While an event to a modern program might be the click of a mouse button or the closing of a window, an event to TALE-SPIN is a specific event that occurs for a character in that simulation. For example, if a character picks up an object, that qualifies as an event. Text then needs to be displayed to the screen, as in "character x picked up the object and waited for something to happen."

TALE-SPIN creates stories based on characters that have a particular problem that needs solving. For instance, there might be a bee character that needs to find its way back to a bee-hive, or a bear character that needs to find a way to obtain the honey from a bee-hive without getting stung. As the characters act in pursuit of their goals, the text describing these actions is printed to the screen in upper-case characters. The following story, shown in Figure 19, is an example TALE-SPIN story taken from Meehan's article.

Once upon a time George Ant lived near a patch of ground. There was a nest in an ash tree. Wilma Bird lived in the nest. There was some water in a river. Wilma knew that the water was in the river. George knew that the water was in the river. One day Wilma was very thirsty. Wilma wanted to get near some water. Wilma flew from her nest across a meadow through a valley to the river. Wilma drank the water. Wilma was not thirsty anymore.

George was very thirsty. George wanted to get near some water. George walked from his patch of ground across the meadow through the valley to a river bank. George fell into the water. George wanted to get near the valley. George couldn't get near the valley. George wanted to get near the meadow. George couldn't get near the meadow. Wilma wanted George to get near the meadow. Wilma wanted to get near George. Wilma grabbed George with her claw. Wilma took George from the river through the valley to the meadow. George was devoted to Wilma. George owed everything to Wilma. Wilma let go of George. George fell to the meadow. The end.

Figure 19: A TALE-SPIN Story (Meehan 199-200)

While this story seems somewhat clunky and awkward to read, it does in fact have all of the essential properties of a story: protagonists (George and Wilma), central concerns (getting a drink of water for Wilma, the same for George and then an additional concern of getting out of the river for George and rescuing George for Wilma), antagonists (the river), a time (once upon a time), and a place (a patch of ground, a tree, a meadow, a valley, and a river). TALE-SPIN supports eight different kinds of characters, forty-six kinds of physical locations constructed from “abstract descriptions,” and problem-solving procedures for transportation, acquisition of objects, acquisition of information, transfer of information, persuasion, bargaining, and asking favors (Meehan 201). There are also thirteen “social states” such as honesty, affection, etc. and forty-one inference procedures to make new inferences from existing knowledge (Meehan 201). The program is written in a language known as MLISP, which is “an ALGOL-like language that is converted by a pre-processor into LISP code for execution” and originally required about 90

kilobytes of memory to load (Meehan 202). Although tiny by modern standards, Meehan lamented this large file size and noted it as the major drawback of TALE-SPIN in 1979 when he was writing this article.

The most interesting thing about TALE-SPIN was the program's degree of sophistication at the time of its design. Meehan writes, "It is easily distinguished from any of the 'mechanical' devices one can use for writing stories ..." (197). Rather than simply using a database full of names, locations, and stored story concepts to create random stories, TALE-SPIN enables users to direct the development of the story by allowing them to choose characters and decide whether or not they want certain objects available to these characters. Users can then decide what each character "knows about" in the story, and the resulting story is then generated based on these variables.

For example, the menu captured in Figure 20 shows the start of an interactive session using TALE-SPIN.

```
*(START)

***** WELCOME TO TALE-SPIN *****

CHOOSE ANY OF THE FOLLOWING CHARACTERS FOR THE STORY:
1: BEAR      2: BEE      3: BOY      4: GIRL      5: FOX
6: CROW      7: ANT      8: CANARY
```

Figure 20: TALE-SPIN Opening Menu (Meehan 204)

Given this menu, the user can select one or more characters to act as an actor in the story. Names and sexes for these characters (except for the menu selections 3 and 4) are chosen at random and local settings and default objects for each character are then created (Meehan 204).

After the characters are created, the program queries the user to decide whether or not the characters “know” about each other and then asks which objects should be added to the story (Meehan 204). This secondary menu is shown in Figure 21.

```
--DECIDE: DOES BETTY BEE KNOW WHERE SAM BEAR IS? *NO
--DECIDE: DOES SAM BEAR KNOW WHERE BETTY BEE IS? *YES
    SAM KNEW THAT BETTY WAS IN HER BEEHIVE.
    SAM KNEW THAT BETTY HAD THE HONEY.

--DECIDE: DO YOU WANT ANY OF THESE IN THE STORY?
    (BREADCRUMBS, CHEESE, BASEBALL) *NO

--DECIDE: DO YOU WANT ANY OF THESE IN THE STORY?
    (BERRIES, FLOWER, WATER, WORM) *YES

WHICH ONES?
1: BERRIES          2: FLOWER          3: WATER          4: WORM
*2
    THERE WAS A ROSE FLOWER IN A FLOWERBED.
WHO KNOWS ABOUT THE ROSE FLOWER?
1: BETTY BEE       2: SAM BEAR
*2
    SAM KNEW THAT THE ROSE FLOWER WAS IN THE FLOWERBED.
```

Figure 21: TALE-SPIN User Interface (Meehan 204-205)

Finally, the user is allowed to choose the main character for the story. In this case it is a choice between Betty Bee and Sam Bear. The user then gives that character a problem, such as the character being hungry, tired, or thirsty, and the system generates a story similar to the *George the Ant* story told previously in this chapter. Such was the extent of TALE-SPIN’s capabilities.

BRUTUS, MINSTREL, and MEXICA

Subsequent researchers critiquing Meehan's TALE-SPIN program identified several flaws with his implementation. Among them were two major problems: that "it was only able to generate a limited range of stories within a rigid pre-defined structure," and "its lack of differentiation between the goals of the characters and the author" (Pérez y Pérez and Sharples 15). Because of these limitations, "uninteresting stories can arise, such as 'John Bear is hungry, John Bear gets some berries, John Bear eats the berries, John Bear is not hungry anymore, the end'" (Pérez y Pérez and Sharples 15). These problems prompted future computerized storytelling researchers to develop story grammars, which "represent stories as linguistic objects which have a constituent structure that can be represented by a grammar" (Pérez y Pérez and Sharples 15). One example of a storytelling system that uses a story grammar is GESTER, which is a program that generates story outlines based on a story-grammar from medieval French epics (Pérez y Pérez and Sharples 15). The problem with GESTER, which is typical of any system using a story grammar, is that it can only produce stories that fall within this rigidly defined grammar system. Story grammars are therefore very controversial among researchers; some believe they are necessary for generating and understanding stories while others argue that the grammars are "incapable of generating all and only valid stories" (Pérez y Pérez and Sharples 15).

Another problem with TALE-SPIN is that the plots of these stories were restricted to solving simple problems, and were therefore quite dull. Rafael Pérez y Pérez and Mike Sharples note, "Other characteristics of successful human stories must be explicitly addressed, such as

novelty and interestingness” (16). These computer models needed to be creative as well as syntactically correct when generating stories. Three models that attempted to achieve this creativity were MINSTREL in 1993, MEXICA in 1999, and BRUTUS in 2000 (Pérez y Pérez and Sharples 16).

In order to evaluate these three creative storytelling systems, Pérez y Pérez and Sharples devised an empirical method based on story-predictability. Story-predictability is a measurement which can be used to assess the creativity of a computerized story. Pérez y Pérez and Sharples write, “... if story-predictability is low, the system is more likely to be evaluated as representing a creative process” (16). In other words, if a reader is not anticipating the transition of one event into another in a particular story’s event fabula, then that story surprises them and does not meet that reader’s expectations. When this surprise happens in a computerized story, then, the computer generating that story appears to be creative.

MINSTREL, the earliest program considered by Pérez y Pérez and Sharples, is “a computer program that writes short stories about King Arthur and his Knights of the round table” (18). In order to accomplish this storytelling process, the software uses both author-level goals and character-level goals combined with “a set of heuristics called Transform Recall Adapt Methods (TRAMS)” that are used to create novel and creative scenes in a story (20). This process makes MINSTREL’s stories seem quite a bit more interesting than the stories output by TALE-SPIN. A story from MINSTREL is shown in Figure 22.

In the spring of 1089, a knight named Lancelot returned to Camelot from elsewhere. Lancelot was hot tempered. Once, when Lancelot lost a joust, he blamed his lance and wanted to destroy it. He struck his lance and broke it. One day, a lady of the court named Andrea wanted to pick some berries. Andrea went to the woods. Andrea picked some berries. Lancelot's horse unexpectedly carried him to the woods. Lancelot was near Andrea and fell in love with her. Lancelot wanted Andrea to love him.

Some time later, Lancelot's horse again carried him into the woods. There he saw Andrea kissing a knight named Frederik. Because of this, Lancelot believed that Andrea loved Frederik. Because Lancelot loved Andrea, he wanted her to love him. Lancelot hated Frederik. Lancelot's hot temper made him mad. He wanted to kill Frederik. Lancelot moved to Frederik and fought him. Frederik was killed.

Andrea ran to Frederik. Andrea told Lancelot that Frederik was her brother. Lancelot wanted to take back that he wanted to kill Frederik, but he could not. Lancelot hated himself. Lancelot became a hermit to hide his shame. Frederik was buried in the woods where he died, and Andrea became a nun.

Figure 22: A MINSTREL Story: Lancelot Story (Pérez y Pérez and Sharples 19)

MINSTREL's improvements over the TALE-SPIN implementation of automated storytelling are numerous, and are as follows:

- MINSTREL is arguably the first computerized storyteller that explicitly represents a computer model of the creative process of writing.
- MINSTREL's main contribution to the computational creativity in writing is the concept of TRAMs, which demonstrate the potential power of small changes in story schemas.
- MINSTREL is one of the first computerized storytellers to explicitly employ author-level goals, combined with character-level goals, to generate adequate and interesting outputs.

- MINSTREL generates novel stories by transforming ‘old’ episodes stored in memory. (Pérez y Pérez and Sharples 21)

Some of these features will be included in the knowledge management software model shown in Chapter Five. While creativity remains the responsibility of the users writing the stories, the author-level goals of *EDNA-E* are imposed through the use of an events management control panel. Users will only be allowed to submit stories that match with these events; the author-level goals are therefore imposed by the software. *EDNA-E* will also generate novel stories by transforming old stories, much like the *monetary-policy story* will be constructed later in this chapter in Graham Smart’s Bank of Canada example. This transformation will allow the merger of many individualized stories into a comprehensive narrative that describes, solves, or considers some institutional problem or task that is relevant to current goals within that organization.

BRUTUS is “a program that writes short stories about pre-defined themes like betrayal” (Pérez y Pérez and Sharples 17). An example of a BRUTUS story is provided in Figure 23. Note the sophistication of the story enabled by the use of a story grammar, which makes this story seem much more engaging and interesting than the previous story generated by TALE-SPIN. The progress of over twenty years worth of automated storytelling research (BRUTUS was designed in 2001) is immediately obvious.

Dave Strider loved the university. He loved its ivy-covered clocktowers, its ancient and sturdy brick, and its sunsplashed verdant greens and eager youth. He also loved the fact that the university is free of the stark unforgiving trials of the business world—only this isn't a fact: academia has its own tests, and some are as merciless as any in the marketplace. A prime example is the dissertation defense: to earn the PhD, to become a doctor, one must pass an oral examination on one's dissertation.

Dave wanted desperately to be a doctor. But he needed the signatures of three people on the first page of his dissertation, the priceless inscription which, together, would certify that he had passed his defense. One of the signatures had to come from Professor Hart. Well before the defense, Striver gave Hart a penultimate copy of his thesis. Hart read it and told Striver that it was absolutely first rate, and that he would gladly sign it at the defense. They even shook hands in Hart's book-lined office. Dave noticed that Hart's eyes were bright and trustful, and his bearing paternal.

At the defense, Dave thought he eloquently summarized Chapter 3 of his dissertation. There were two questions, one from Professor Rodman and one from Dr. Teer; Dave answered both, apparently to everyone's satisfaction. There were no further objections. Professor Rodman signed. He slid the tome to Teer; she too signed, and then slid it in front of Hart. Hart didn't move. "Ed?" Rodman said. Hart still sat motionless. Dave felt slightly dizzy. "Edward, are you going to sign?" Later, Hart sat alone in his office, in his big leather chair, underneath his framed PhD diploma.

Figure 23: A BRUTUS Story: Simple Betrayal (Pérez y Pérez and Sharples 18)

The BRUTUS system uses a combination of frames and production rules in order to generate stories similar to the one shown in Figure 23. This procedure depends on the following three core processes:

1. "The instantiation of a thematic frame" (Pérez y Pérez and Sharples 17).
2. "A simulation process where characters attempt to achieve a set of pre-defined goals (development of a plot)" (Pérez y Pérez and Sharples 17).
3. "The expansion of story-grammars to produce the final output" (Pérez y Pérez and Sharples 17).

This type of system is important because it is very similar to the method that is used in the event-driven narrative analysis engine designed in Chapter Five. The thematic frame in this system is in fact the event that controls the topic of a collected story. The second core process, the achievement of goals by characters in the story, is handled by the user who is typing the story into the system. Story grammars are not used in *EDNA-E*, so the final output of the system is handled by a human story administrator. Much like BRUTUS', though, *EDNA-E*'s contribution "consists of merging different known methodologies into one program" (Pérez y Pérez and Sharples 18).

The last of the three systems considered by Pérez y Pérez and Sharples is MEXICA. MEXICA is "a computer model based on an engagement-reflection cognitive account of creative writing that produces stories about the MEXICAs (the former inhabitants of what today is México City, also wrongly known as the Aztecs)" (21). The engagement-reflection mechanism works by allowing the software to analyze the story as it is constructed and adjust the plot developments accordingly. Pérez y Pérez and Sharples explain,

During the engagement-mode the system produces material driven by content and rhetorical constraints avoiding the use of explicit goal-states or story-structure information. During the reflection-mode the system breaks impasses generated during engagement, satisfies coherence requirements, and evaluates the novelty and interestingness of the story in progress. If the results of the evaluation are not satisfactory, MEXICA can modify the constraints (21)

What this means is that “the stories produced by the program are the result of interaction between engagement and reflection” (22). A story generated by MEXICA is shown in Figure 24. Note the text in italics here indicates the reflection mode of the system. Of the three storytelling systems, MEXICA is the least sophisticated in terms of natural language syntax generation (Pérez y Pérez and Sharples 27).

Jaguar knight was an inhabitant of the great Tenochtitlan. Princess was an inhabitant of the great Tenochtitlan. From the first day they met, princess felt a special affection for jaguar knight. Although at the beginning princess did not want to admit it, princess fell in love with jaguar knight. Princess respected and admired artist because artist's heroic and intrepid behaviour during the last Flowery-war. For long time jaguar knight and princess had been flirting. Now, openly they accepted the mutual attraction they felt for each other. Jaguar knight was an ambitious person and wanted to be rich and powerful. So, jaguar knight kidnapped artist and went to Chapultepec forest. Jaguar knight's plan was to ask for an important amount of cacauatl (cacao beans) and quetzalli (quetzal) feathers to liberate artist. Princess had ambivalent thoughts towards jaguar knight. On one hand princess had strong feelings towards jaguar knight but on the other hand princess abominated what jaguar knight did. Suddenly, the day turned into night and after seconds the sun shone again. Princess was scared. The Shaman explained to princess that Tonatiuh (the divinity representing the sun) was demanding princess to rescue artist and punish the criminal. Otherwise princess's family would die. Early in the morning princess went to Chapultepec forest. Princess thoroughly observed jaguar knight. Then, princess took a dagger, jumped towards jaguar knight and attacked jaguar knight. Jaguar knight was shocked by princess's actions and for some seconds jaguar knight did not know what to do. Suddenly, princess and jaguar knight were involved in a violent fight. In a fast movement, jaguar knight wounded princess. An intense haemorrhage arose which weakened princess. Jaguar knight felt panic and ran away. Thus, while Tlahuizcalpantecuhltli (the god who affected people's fate with his lance) observed, princess cut the rope which bound artist. Finally, artist was free again! Princess was emotionally affected and was not sure if what princess did was right. Princess was really confused. The injuries that princess received were very serious. So, while praying to Mictlantecuhltli (the lord of the land of the dead) princess died.

Figure 24: A MEXICA Story: The Lovers (Pérez y Pérez and Sharples 22)

Organizational and Institutional Storytelling Systems

Although computerized storytelling models are interesting for determining how stories can be represented as digitized data, the most useful examples of storytelling systems for the purposes of this dissertation are the systems that demonstrate the knowledge management capabilities of narrative devices in large organizations. Fortunately, there are several well-documented examples of these narrative KM tools at work. Many of these systems, including

the two discussed here, have been demonstrated as important tools and strategies in the banking industry. The Bank of Canada's model, considered later in this chapter, uses one such system. Graham Smart explains, "As a shared cognitive and rhetorical resource, the narrative plays a central role in the collaborative construction of the specialized knowledge that allows the bank to accomplish its work in the world" (268). Often these rhetorical frameworks need a particular type of story in order to function efficiently. For example, consider the springboard system devised by Stephen Denning, who at the time of his book's publication was the program director of knowledge management at World Bank.

These types of KM systems rely on what Denning characterizes as the "springboard story," which is "a story that enables a leap in understanding by the audience so as to grasp how an organization or community or complex system may change" (xviii). Springboard stories act as entry points for complex issues that often involve tacit knowledge and are not immediately accessible by non-experts. These entry points are often constructed using figurative language of some sort, which allows an audience to compare their understanding of a familiar topic to this new and unknown idea they are now considering.

The springboard story is one type of story that can be built using *EDNA-E*. It is therefore necessary to define this type of story's features so that they can be enforced using a DTD and an appropriate script for potential authors. Denning found that the most effective springboard stories all had the following characteristics:

1. The stories were "told from the perspective of a single protagonist who was in a predicament that was prototypical of the organization's business" (xix).

2. “The predicament of the explicit story was familiar to the particular audience, and indeed, it was the very predicament that the change proposal was meant to solve” (xix).
3. “The story had a degree of strangeness or incongruity for the listeners, so that it captured their attentions and stimulated their imaginations” (xix).
4. The story was plausible and familiar (xix).
5. The stories were told as simply and briefly as possible, “speed and conciseness of style were keys...” (xix).
6. The stories all had “happy endings” (xx).

Looking back to the *SparkleTech* story from Chapter Three, it should now be clear that the brief story about Chris, Sandra, and Jane is a potential springboard story. Each of the independent stories about Chris, Sandra, and Jane could be represented in a single story from each perspective: Chris could write about his technical issues with the network routers, Jane could write about her trouble gathering material for the company newsletter, and Sandra could write about the difficulties of closing a major corporate contract while at the same time attempting to manage an employee with whom she had once had an outside relationship. Each of these stories would certainly be familiar to a specific audience of SparkleTech employees; Chris’ story would be familiar to technicians, Jane’s to technical writers, and Sandra’s to other employees in management.

The *SparkleTech* story also has the degree of strangeness necessary to spark imaginations in the readers or listeners of their stories. For instance, Chris’ eyepatch and eccentric behavior, along with his nickname and former relationship with his boss, add a bit of dramatic flair to his

actions attempting to fix the company's network. Jane could write about her recent triathlon completion and use the strenuous athletic competition as a metaphor for overcoming adversity for seemingly overwhelming workplace projects. Finally, Sandra's relationship with Chris could also be used metaphorically: landing the big contract is a lot like working with an ex-significant other; both require a degree of delicacy and decorum at times when such behavior seems quite impossible.

The next characteristic of the springboard story, familiarity, is also present in the *SparkleTech* story. Anyone who has ever tried to set up a home network is familiar with the frustrations that can accompany this process; these same frustrations are simply magnified several times over for Chris in his attempts to fix a much larger network. Most people can likely also identify with trying to finish a large paper document under a given deadline, as was the case with Jane and her company newsletter. Finally, the social and ethical concerns that accompany management decisions are also familiar to those with managerial responsibilities. Sandra's experiences with such decisions should also strike a familiar note with these types of readers or listeners.

Brevity was evident in the *SparkleTech* story, too. Although each character was given a minor descriptive characteristic and an outside interest, these details were minute enough to introduce a small amount of strangeness to the story without losing focus of the organizational problems that needed to be solved. As soon as a clear solution to the problems was evident, the story moved towards a closing point, and in this particular story the solution turned out to be a simple change of perspective for each protagonist. Springboard stories are not meant to be

immersive and wholly entertaining; they need just enough detail to spark a creative link in the interpreter of that story. The *SparkleTech* story, however brief, was meant to do just that.

Denning's last story feature, the happy ending, is most open to debate among his characteristics. If someone is to invest the time and mental energy to fully digest a story only to realize that the outcome was not pleasant for the protagonists then it might seem as though that story were a waste of time. On the other hand, though, if the point of that particular story were to illustrate some of the negative consequences of a particular action, then this type of ending would be very effective. This property depends largely on the objective of the story: it can facilitate understanding and show how change happens within an organization through negation as well as straightforward definition. For the most part, however, a happy ending to a story will work well when trying to show how a particular problem was overcome using strategies evident in that particular narrative. This type of strategy was used in the story about Chris, Jane, and Sandra. The happy ending of this narrative showed how each individual problem, as well as the overall organizational problem, were all eventually solved.

Later in his book, Denning expands his ideas about springboard stories to write that these types of stories need three essential keys, in order of importance. These three keys are:

1. "Connectedness: The story, however condensed, has to link the audience with a positive controlling idea and a protagonist with whom the audience identifies" (124).
2. "Strangeness: The springboard story must violate the listener's expectations in some way" (126).
3. "Comprehensibility: The story has to embody the idea so as to spring the listener to a new level of understanding" (128).

There are again many issues that will accompany stories exhibiting these properties. Firstly, as the protagonist must be identifiable and recognizable, there may be some storytellers uncomfortable with sharing personal aspects of their lives that make them identifiable in the story. For instance, many people would be uncomfortable writing about details which portray them as acting strange or eccentric, which is the second essential key identified by Denning.

Secondly, in order to violate listener expectations, one must have a good idea about what these listeners are expecting in the first place. This type of intuition will likely improve as one remains in a position for a longer period of time, but may be more difficult for newly hired or recently relocated employees. Some expectations can be based on common sense or on culturally-shared beliefs, as well. The types of situations where these types of expectations are anticipated would best be added into stories meant to be read both inside and outside of an organization, since specific expectations about organizational goals, job descriptions, and duties are relative to any given organization.

Finally, comprehensibility, Denning's most important key, must be considered. In order to "spring a listener to a new level of understanding" (128), authors must be willing to do the proper research to ensure they completely understand the topic themselves. Unfortunately, time limitations often mean that an employee can only invest the amount of time necessary to solve a problem and can not necessarily afford the extra few hours to research the problem and find out exactly why it occurred. It is up to the knowledge manager, then, to convince management that such time is time well spent; the few additional hours up front might save days or even weeks in the long run when a variation of the same problem is encountered several years down the road and the original employee who solved that problem has long since retired or relocated.

Denning's third key thus suggests that some training might be beneficial for the story authors/senders as well as the story listeners/receivers.

Another organizational storytelling system in a banking setting was studied by Graham Smart. Smart studied "narrative construction as a communal process of corporate knowledge making" at the Bank of Canada (249). Smart was particularly concerned with one narrative present in that environment. He writes,

... I trace the construction and use of one particular narrative—known in the bank as the *monetary-policy story*—as it evolves across a number of written genres, incorporating and synthesizing analyses from different economists, until reaching a final stage where it is used by the bank's executives as a shared cognitive and rhetorical resource for making decisions about monetary policy and for explaining and justifying those decisions to the Canadian public. Viewed in this way, the *monetary-policy story* is a complex structure of economic knowledge that, in the course of its production, serves to organize, to consolidate, and to give textual expression to the differentiated expertise and intellectual efforts of a large number of people. (250)

The monetary-policy story is thus another instantiation of the organizational stories that are collected using the narrative analysis engine in Chapter Five. Smart's analysis, much like Denning's, is important because it studies the impact of stories in very large and complicated organizations. Banks not only have to deal with the maintenance, collection, and dissemination of all types and variations of fiscal materials, but also with the international relations and cultural

differences omnipresent in such relations. This makes the success of storytelling as a knowledge management tool all the more remarkable.

The Bank of Canada relies on a computerized model for generating storytelling scenarios to describe the Canadian economy. Their program, known as *QPM*, was developed in-house by expert economists and is able to present a “mathematical representation of the economy” based on “30 interconnected equations depicting different ‘sectors’ of the economy” (Smart 256). *QPM* can also be used to combine individual narratives into a larger institutional narrative, the process of which is described later in this analysis of Smart’s ethnography. This final institutional story, which will later be described as a story not confined to the boundaries of the printed page, “becomes a shared cognitive resource for making decisions about policy” (Smart 266-267) for this organization. Building the final story requires a bottom-up approach, which means that the final story is built from several smaller stories that continue to be combined together at each level climbed in the narrative hierarchy. This cycle requires “... the merging of statistical data, verbal and mathematical discourse, and professional judgment” (Smart 269).

Several different aspects of Smart’s research are particularly interesting. For one thing, his attention to genres in the knowledge-creation process is especially supportive of the script-based system that is devised in Chapter Five. Smart cites the research of Amy Devitt in devising a genre-theory for the “knowledge-making functions of families of genres” (252). He writes, “According to Devitt, an organization’s system of written genres can be seen as a multifaceted rhetorical structure for creating and distributing knowledge needed for carrying out the organization’s work” (252). It is entirely possible to construct genres based on division and classification, which can then be coded into a script and used to control and distribute

appropriate stories to appropriate groups based on this rhetorical strategy and taxonomy. Genres can also be constructed for the story elements themselves, and even more complex queries can be built with these genres in mind. For example, if all stories and authors were given genres and stored in a database, we could query that database with a simple structured query language (SQL) statement as shown in Figure 25.

```
SELECT story FROM database_table WHERE  
(author_genre = 'SparkleTech Technical Division' &&  
story_genre='Network Problems' order by name);
```

Figure 25: Story Selection SQL Statement

Smart's research also reminds us of Marvin Minsky's idea of frames, although this time the frames are used in an empirical setting to analyze economic data. As Smart explains, quoting a Bank of Canada department chief, "... the bank's policy actions are 'based on a theoretical and empirical view of how the economy functions.' He continues to describe the use of institutionally created analytic 'frameworks' to interpret economic data, an ongoing activity known in the banks as 'current analysis'" (256). Once again we have the idea of a blank frame (in this case a frame representing an indicator of current economic trends) with terminals that can be filled in with specific information to create multiple instances from that frame.

Another interesting detail of Smart's research is his consideration of distributed cognition. In Chapter Two, some of the difficulties of traditional artificial intelligence were discussed; one of the major difficulties was trying to achieve AI in a single isolated computer. Distributed cognition, which is more supportive of the emergent properties such as those studied

by Eric Bonabeau et al. as explained in Chapter Three and also theorized about by Lewis Thomas in Chapter Two, is a model of intelligence in which “knowledge is understood to be activity based and socially generated” (Smart 253). Such properties of intelligence were discussed from the onset of this dissertation, but we now have a means of again restating this idea in the words of a researcher studying an applied model. Smart explains, “... distributed cognition theorists accord close attention to the various ‘mediational means’ – symbolic representations, technologies, collaborative arrangements—that groups employ to enable the particular specialized forms of reasoning and knowledge creation required to accomplish their work” (253). He continues to explain that narratives are in fact symbolic representations which foster “intersubjectivity – that is, the ground of shared understandings—necessary for productive intellectual collaboration” (253).

The idea of intersubjectivity suggests that we have our own private thinking spaces where we can perform functions like symbolically working out math problems or thinking about our own personal experiences. There is then an additional public or communal space where “intersubjectivity is negotiated” (253). In this space, we can communicate our own thoughts as well as receive communications of other private thoughts that have been transported into the communal space through one vehicle or another. The argument that Smart makes, and the argument I make in this dissertation, is that stories are the optimal vehicles for such transfers.

Smart’s methodology for studying the *monetary-policy story* at the Bank of Canada was to carry out an “interpretive ethnography” where he studied various forms of qualitative data including “field notes from on-site observations, reading protocols, tape-recorded meetings, recordings of presentations by senior economists at writing seminars, internal and published

documents, and scores of interviews” from the years 1983 through 1996 (254). Smart observed economists, business analysts, administrators, and computer experts, but mainly focused on economists during the latter years of 1994 through 1996 (254).

In his study, Smart identifies three stages in the construction of the *monetary-policy story*. The narrative itself appears in the first stage as a “cluster of ... sector stories, specialists’ analyses of developments in different sectors of the economy” (257). The second stage builds a narrative about the Canadian economy as a whole, and is “produced by a team of economists during a quarterly activity known as the Projection Exercise and then inscribed in a document called the *White Book*” (257). The final stage is the construction of “a fully elaborated institutional story, constructed by the executives from the *White Book* and other sources of information” (257). The model here is similar to the storytelling model presented in Chapter Five, although there are only two stages in the construction of stories in *EDNA-E*. Like the Bank of Canada system, though, the stories from the first stage (individual stories) will be combined into a more comprehensive institutional narrative in the second stage. This is done using a computerized model to help identify stories that can be combined together into the next stage’s narrative form. In the Bank of Canada narrative system this model was *QPM*. A similar model I used in my software is also presented in this dissertation; this computerized helper is explained and described in detail in the next chapter.

As Smart explains, each of these stages of story collection can be used to address a number of questions that may not be answerable using other techniques. For example, in the creation of the stage one sector analysis stories, economists can consider the following questions:

- “What of significance has been happening in a particular area of the economy, is happening now, or is expected to happen in the future?” (259).
- “What are the explanations for these developments, in terms of the underlying economic forces at play?” (259).
- “How do the developments confirm or challenge the bank’s current view of the economy?” (259).
- “And what are the implications of all this for ‘the job we do here at the bank,’ conducting monetary policy?” (259).

Each one of these questions can be added into a script as a central concern as the “central concern” frame terminal theorized by Minsky and mentioned in Chapter Three of this dissertation. The script, when combined with a DTD, will only accept stories from users that match the particular concern from one of those four key questions. These questions, or any additional questions, can be formulated specifically for the goals and operations of any organization and framed accordingly using Minsky’s thematic terminal for any particular story. Undoubtedly, the *QPM* model described by Smart used a similar method when gathering sector analysis stories, but for the model in Chapter Five the script will control the types of central concerns or questions that the story control mechanism will accept.

The next stage of building the *monetary-policy story* is to combine the sector stories into a larger narrative which is stored in a document known as the *White Book*. This task is carried out through a Projection Exercise, in which “a team of six senior economists ... consolidate and interpret information about different areas of the economy provided by the 14 sector specialists” (Smart 261). The resulting *White Book* is “addressed to the Bank’s executives” and “describes

current conditions and developments in the Canadian economy, forecasts trends, and recommends a direction for monetary policy” (Smart 261). In other words, the *White Book* takes the data from the sector analysis stories and turns that data into useful information and possibly knowledge which can be used by the Bank executives. Such a procedure is the *modus operandi* for a knowledge management system, and makes it possible to reduce enormous amounts of data into a format digestible by those who need to make quick decisions using that data.

Unlike the short organizational stories described by Stephen Denning, the *White Book* story is a large document, approximately 50 pages in length (Smart 263). The critical difference here is that the story told by the *White Book* is a compilation of many other smaller stories that have been organized together by economic experts over many meetings. At this point the narrative has also gone through several format reviews. Denning’s stories are more appropriate for those individuals looking to find tacit knowledge quickly in order to solve a particular problem, whereas the Bank of Canada *White Book* is more suitable for producing a large compendium of knowledge that represents a massive volume of interpreted numeric and statistical data capable of solving many problems. When finished, the *White Book* is the physical representation of many experts’ tacit knowledge and professional judgment. As one department chief stated, “It’s a summary of everything we know, the best advice we can give them” (Smart 263). This expert knowledge is then passed on to bank executives for the final stage of building the *monetary-policy story* (Smart 263).

The final phase in building this story is a verification phase administered by the bank executives. This phase consists of a meeting where the *White Book* authors meet with and are questioned by the bank executives in order to make sure the communication of the story is clear

and unambiguous. As one executive explains, “We probe, ask questions, make sure that we understand what’s there. They’re presenting a product, and what’s important for us is that we understand the thinking and judgment that went into it, so that we can then discuss and debate that among ourselves later” (Smart 264). Finally, the *White Book* is approved and serves as the “basis for a more comprehensive narrative constructed by the executives themselves. We might see this narrative as the full institutional *story* on the state of the Canadian economy, the probably future developments, and the appropriate monetary policy to be followed” (Smart 265). Human judgment working in conjunction with the automated *QPM* system is again critical at this point. For instance, one executive Smart quotes is stated as saying, “One thing QPM doesn’t deal with—and I don’t think any model can—is the state of market confidence, or market nervousness, in various circumstances. So this is something we factor in afterwards” (265). The final narrative is known as the Bank of Canada institutional story, and at this point the narrative is no longer confined to a printed medium. Instead, it exists as an active story that is constantly being rebounded through spoken discourse. Smart explains, “... the fully elaborated version is nowhere completely articulated in written form in any internal document; rather, it resides in the executives’ discourse, surfacing in meetings and informal conversations, in whole or in part, and underlying certain assumptions and lines of argument in the texts they produce” (266).

A graphical depiction of the steps leading up to the creation of the institutional story is shown in Figure 26. To summarize the process, multiple sector stories (although only five are shown in the model to save space, fourteen are actually used by the Bank of Canada process) are combined into a more comprehensive narrative in a document known as the *White Book*. After meeting with the *White Book* authors, bank executives then add their own professional judgment

to further refine this narrative into its final form, which is the full institutional story. It is important to realize that at each phase of this knowledge-building process, expert human guidance is needed in order to move to the next level in building the *monetary-policy story*. This story, “in its three stages of construction, functions as a key locus of knowledge making in the bank” (Smart 268).

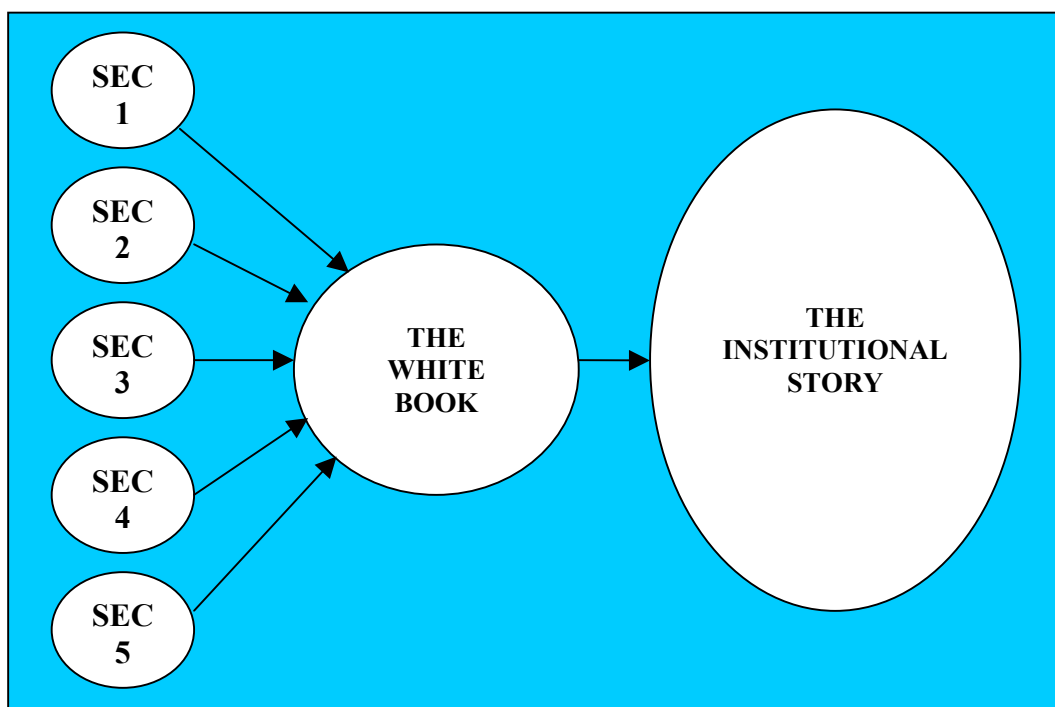


Figure 26: Building the Monetary-Policy Story

Visual Storytelling

The most popular forms of storytelling through imagery are found in film and comic books (Eisner 3). To describe stories told through imagery, Will Eisner uses the term *graphic narrative*, which is “a generic description of any narration that employs image to transmit an

idea. Film and comics both engage in graphic narrative” (6). A story told primarily through graphical means retains the same basic structure of a textual story: there is an introduction or setting, a problem, a segment in which that problem is dealt with, a solution, and an end (Eisner 9). In Eisner’s words, “the skeleton is the same” (9). This similarity of structure makes it easy to interchange graphical and textual elements in a story, which is of course done in comic books but is also a technique that may be used in future versions of my *EDNA-E* narrative software.

A story, then, can be drafted first in an abstract form. In this form, the individual images that are drawn have no relationship to one another. After the individual story elements are created (Figure 27), the story is given a sequence and an order by connecting these various elements in a way that makes logical sense (Figure 28). For instance, in the sequence of events illustrated in Figure 28, it would make no sense if the caveman’s branch broke off before he had climbed the tree to escape the wolf. Of course this is only one simple example; the same idea can be expanded to take into account emotional reactions, character development, and so on. Also, as in the computer generated textual storytelling programs discussed earlier in this chapter, we can measure the creativity and novelty of a visual story by looking at how the reader’s expectations in terms of story-predictability are violated.



Figure 27: Story Elements Unorganized (Eisner 10)

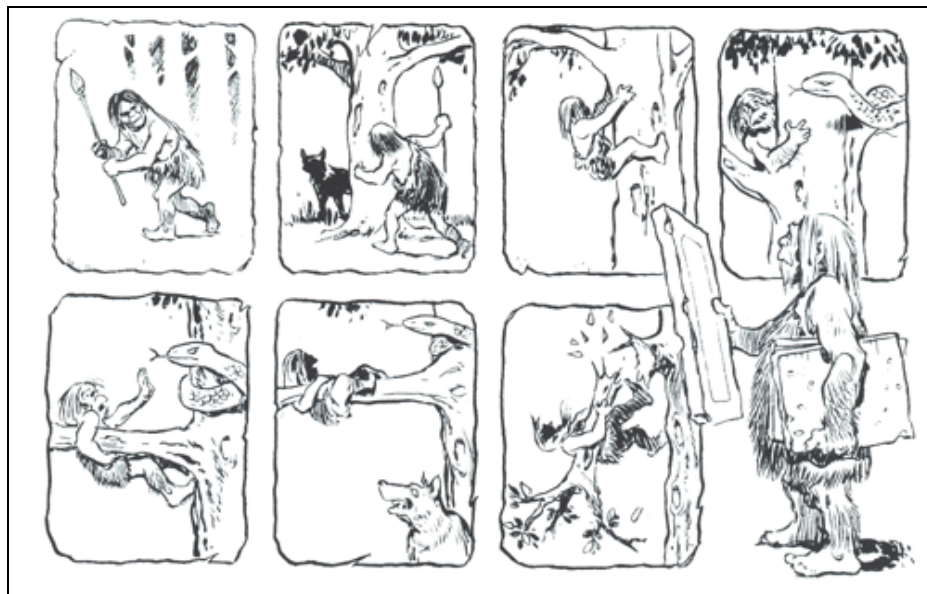


Figure 28: An Organized Story (Eisner 10)

Another premier researcher of comic theory is Scott McCloud. McCloud writes of the importance of separating form and content in comics, which he describes as “aesthetic surgery” (5). Here, the form represents the particular sequence of visual images (what Eisner calls “sequential art”). McCloud revises and expands this definition of the comic to include “juxtaposed pictorial and other images in deliberate sequence, intended to convey information

and/or produce an aesthetic response in the viewer” (9). McCloud’s book is written in comic book format with McCloud himself as the main character in the comic book. The page in which McCloud defines the comic, for example, is shown in Figure 29.

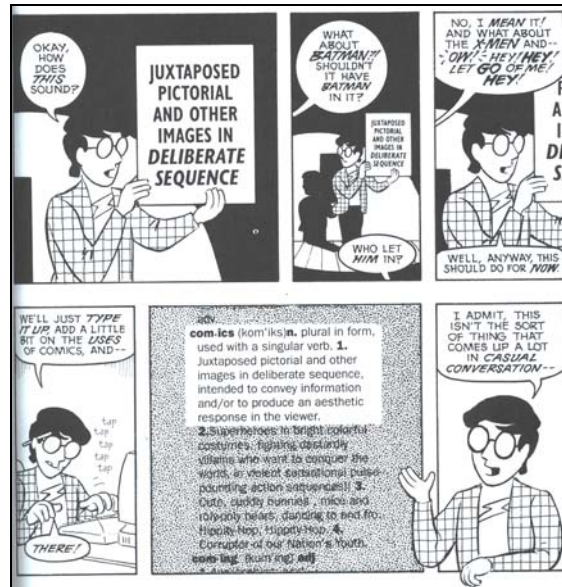


Figure 29: Comics Defined (McCloud 9)

Hans-Christian Christiansen has further researched the relationship between comics and film by studying the narrative devices such as “close-up, chiaroscuro, point of view, and dynamic editing of camera angles” used in each text (108). By observing both the form and substance of cinematography and comics, he has identified four devices that are useful in studying the development of visual narratives. The first of these four devices are conventions, which are “devices that rely primarily on culturally specific skills and knowledge” (110). The second of these devices are norms, which are “devices and techniques more loosely related to cultural codes” (110). There are then cross-cultural universals, which are “figures and

techniques instantly recognizable across cultures and time,” and deep-structures, which refer to “techniques based on general models of storytelling or even universal human experience ...” (110). By examining these four devices, we can see that film and comic media are both able to reach cross-cultural audiences immediately and without translation in places that would not be possible using written language.

For example, in the *SparkleTech* story from Chapter Three, the entire narrative could be represented in a visual form. In one frame, a drawing of Chris might show him hunkered down in front of a stack of network equipment with an expression of agony on his face. This expression, which is both a cross-cultural universal expression as well as a deep-structure of sorts, shows the “reader” that Chris is clearly unhappy and is experiencing some type of problem with his technical machinery. Without having to translate, “Chris glances at his watch, then back at the server array in disgust” into several different languages, the same event transfer in the narrative fabula of the *SparkleTech* story can be represented using only two frames of images. In the first, Chris and Jane (who has just entered the room and is asking Chris a question) are standing in a room with Chris looking down at his watch. Here Chris and Jane are what Claire Harrison calls the “represented participants” (RPs) in that image, a subset of the larger “representational metanarrative” which includes all people, objects, and places shown in an image (50). In the second frame, a close up reveals Chris, who is now wearing that cross-cultural universal expression of disgust/agony as he looks at flashing lights on a piece of networking hardware. Visual stories can therefore be especially useful for organizations with multi-cultural or international employees and clients, and they can provide valuable narrative signposts without relying on the added distraction of written translations. In addition, entire

stories can be created from still images simply from observing how the represented participants in that image are portrayed. Harrison explains, “Narrative images allow viewers to create a story about the RPs because the images include vectors of motion” (51).

Will Eisner explains that this ability of comics to elicit universal recognition is also present when using the graphic stereotype: “... the stereotype is a fact of life in the comics medium. It is an accursed necessity – a tool of communication that is an inescapable ingredient in most cartoons” (17). The stereotype, despite its pejorative connotations, is actually a useful tool in situations where themes and ideas are communicated primarily through imagery. To again return briefly to the *SparkleTech* story, Chris might be represented in his comic frames as a disheveled man wearing a pocket protector and with various tools sticking out from his many pockets. This image caters to the popular stereotype of the scientist/technician, so the image is able to communicate even more information about Chris’ occupation and probable job duties in this particular situation.

The next chapter of this dissertation incorporates the best theories and ideas regarding artificial intelligence and narratology in order to build a working web application that collects and distributes stories. This system can function as a knowledge management tool in any distributed organization, and is flexible enough to be implemented in a variety of situations. Despite the non agent-based examples shown in this chapter, it is quite possible that the next evolution of computer-based storytelling systems may utilize agents or multi-agent systems to create stories. Examples of these types of systems include the *Virtual Storyteller* developed at the University of Twente, which can automatically develop story plots using different methods (Theune et al. 1). Speculations on additional implementations for and variations of the static

model shown in Chapter Five, including an agent-based implementation, follow this chapter in the conclusion.

CHAPTER FIVE: MODELING *EDNA-E*

“The quantity of useful information potentially available online has increased beyond all control, whereas the technology whereby the network actually allows us to retrieve the data we are looking for has improved much more slowly.” – Luciano Floridi

Background

Chapter Two of this dissertation was the most exploratory chapter; Chapter Five is the most concrete and focused chapter. The chapter is concrete in that it actually presents an applied model, *EDNA-E*, which was created using many of the theoretical ideas presented in Chapters One through Three. It is the most focused in that it also challenges many of these theories in order to focus only on the best and most useful paradigms for manipulating data and creating a suitable environment for information and knowledge exchange.

The intent of earlier chapters was to study a wide range of technologies and theoretical ideas to find the best possible methods and tools with which to create a unique and original model. This chapter is divided into two parts. The first part connects the various theories from the first three chapters, discarding those ideas and tools that do not fit my idea for an intelligent knowledge management system and otherwise revising and extending those ideas and tools that do. The second and much longer part of Chapter Five explains in detail how one instantiation of this narrative tool for knowledge management was created using modern Internet technologies. This discussion starts out by modeling some key features in UML and explaining the user and

story administrator control panels. Additional program features are then explained by chronicling two typical *EDNA-E* sessions: the first an administrative preparatory session which occurs in two separate phases, and the second a user story-creation session which also requires two different phases.

I will show that this tool is in fact flexible enough to handle the knowledge management needs of almost any type of distributed organization with three or more hierarchical divisions. These divisions can also be arbitrary rather than actual entities; in other words, a small organization's hierarchy could be set up as "small organization, employees, local employees" just as easily as a large organization could be organized as "large organization, banking division, bank teller." By discussing features such as organizations, groups, and scripts, I will show that this tool can be used in many different types of environments, from counseling centers to technical research corporations, in order to gather stories and facilitate the exchange of certain types of socially-constructed knowledge. Because of the complexity of certain program features, I have inserted numerous screen captures taken directly from my software in order to help illustrate functionality.

EDNA-E, the event-driven narrative analysis engine, is intended to function as a useful tool for encoding stories into XML and facilitating user-driven searches between common story elements. The software relies on a new model for knowledge representation which mixes various elements discussed in the first three chapters of this dissertation. It is based primarily on the theories of Mieke Bal, Marvin Minsky, and Roger Schank, as discussed extensively in Chapter Three, but it also takes into account artificial intelligence research from Chapter Two and incorporates ideas from the existing narrative tools explained in Chapter Four. The software

is not intended to serve as a foolproof means for categorizing and representing stories at their fundamental units of semantic meaning. Such a program would likely not be possible, especially since the narrative fabula requires certain rules in order to link with the story and textual layers. It is more accurately described as an interesting tool with which to examine stories and the relationship between various stories. As Mieke Bal writes, narratology itself is an imprecise theory: “Readers are offered an instrument with which they can describe narrative texts. This does not imply that the theory is some kind of machine into which one inserts a text at one end and expects an adequate description to roll out the other. The concepts that are presented here must be regarded as intellectual tools” (3-4). Such theoretical tools are in this chapter integrated into an applied tool which is useful for managing and categorizing collected knowledge.

With that in mind, I will now describe each of the theoretical approaches and tools from earlier chapters that was important to *EDNA-E*’s design.

XML and the Semantic Web were discussed in the Introduction to this dissertation. It was vitally important for this somewhat technical analysis to open the dissertation because XML serves as the core language in which data is encoded in this scheme. As promised in this Introduction, XML is indeed discussed in more detail in this chapter, but here it is defined primarily through example and its function as a classification markup language in the software. In *EDNA-E*, everything from the stories and scripts themselves to the data dictionaries and even the error logs are marked up in XML. While Chapter One presented an example of data describing this dissertation being marked up in XML, this chapter will show many more examples of XML-encoded data that are actually parsed and manipulated by a software system. The XML used by this software needs to be well-formed, but not necessarily valid. Because of

the way in which the XML files are parsed, invalid data will simply not be read during the parsing process. The script files used by *EDNA-E*, though, are both well-formed and valid, and have DTD files included in order to reflect the proper structure and syntax for these files. Such a structure is necessary in order to ensure data integrity for the XML data files used by the system.

Also mentioned in Chapter One were potential uses for XML data files. For the XML dissertation example provided, one suggested implementation was a software agent that could navigate through dissertation files marked up in XML and gather listings and indices of information that could later be used for searches and research collaborations. In this chapter, a more robust example is provided which shows how scripts and stories can be encoded in XML in order to represent data in a more meaningful format than would be possible using standard HTML. Even the error file is saved in an XML format and is parsed during error message generation; such encoding makes it very easy to gather detailed information about user and run-time errors that can be further used to improve the system.

Free exchange of information with other systems is another benefit of XML data representation. This advantage is explained in more detail by describing the functioning of *EDNA-E*'s data storage in this chapter. As all stories and scripts are stored both in a database and in XML text files, stories could conceivably be swapped and traded with other narrative systems. Other systems can simply look at the DTD headings in each file and determine the acceptable elements and structure for that particular story or script. Then, the systems can adapt that story or script accordingly to meet their own requirements and display or store that narrative using an entirely different methodology.

With the benefits of XML now thoroughly explained, I will continue to discuss the operation of this knowledge management software in a bottom-up fashion as this chapter progresses. In other words, I will start with the fundamental data types (mostly objects) and then move on to show how these objects combine into actual program functions and web application features. Some objects and features will also be shown using UML to demonstrate how the different actors/users and object methods interact within this application. The most generalized description of this software's operation will reveal that this system is in fact a smaller-scale version of other Semantic Web-enabled applications such as IBM's *WebFountain*.

Chapter Two explained how traditional mechanical and computerized systems have dealt with the task of processing data. In particular, the idea of a boundary system is considered, with an emphasis on the problems involved with trying to achieve artificial intelligence using a single binary computer. Jacques Derrida claims that oral language itself is unstable and unsuitable for a binary representation with a signified concept and a signifying sound-concept (14). When writing is involved, even another layer of interpretation is added, and the original concept may become even more detached from the written language describing that concept. Computer programming languages, which are often translated from code a programmer can understand to binary machine language that only a computer can make sense of, represent yet another step away from the pure essence or meaning of a particular signified sound concept.

Machine languages, whether compiled from object-oriented languages such as C++, compiled and interpreted from object-oriented languages like Java, or even simply interpreted from AI-tailored languages like Lisp or Prolog, all explain to a computer how to represent concepts or even how to build new knowledge through an inference process. Traditional AI

software programming usually pairs a search phase, defined as “a general method for finding appropriate solutions from many possible,” with a knowledge representation phase (Gams 9).

What these programmed applications do not do, however, is tell the machine what this represented knowledge actually *means*. The distinction between whether or not a computer can actually understand the data it is processing is in fact what separates strong AI from weak AI.

John Searle writes,

According to weak AI, the principal value of the computer in the study of the mind is that it gives us a very powerful tool. For example, it enables us to formulate and test hypotheses in a more rigorous and precise fashion. But according to strong AI, the computer is not merely a tool in the study of the mind; rather, the appropriately programmed computer really is a mind, in the sense that computers given the right programs can be literally said to *understand* and have other cognitive states. In strong AI, because the programmed computer can have cognitive states, the programs are not mere tools that enable us to test psychological explanations; rather, the programs are themselves the explanations. (“Minds, Brains, and Programs” online)

In this description, strong AI certainly seems to be the model that the groundbreaking AI technologies shown in film and described in science fiction would fit into. Unfortunately, strong AI is also very difficult (some argue impossible) to achieve in a software program. Jean-Francois Lyotard writes that “the body might be considered the hardware of the complex technical device that is human thought” (13). Such a definition assumes that the human body and human thought are separable, which is a claim many philosophers such as John Searle and

Hubert Dreyfus do not agree with. Dreyfus, in his follow-up effort to his originally controversial *What Computers Can't Do*, writes the following in his introduction:

Almost half a century ago computer pioneer Alan Turing suggested that a high-speed digital computer, programmed with rules and facts, might exhibit intelligent behavior. Thus was born the field later called artificial intelligence (AI). After fifty years of effort, however, it is now clear to all but a few diehards that this attempt to produce general intelligence has failed. This failure does not mean that this sort of AI is impossible; no one has been able to come up with such a negative proof. Rather, it has turned out that, for the time being at least, the research program based on the assumption that human beings produce intelligence using facts and rules has reached a dead end, and there is no reason to think it could ever succeed. Indeed, what John Haugeland has called Good Old-Fashioned AI (GOFAI) is a paradigm case of what philosophers of science call a degenerating research program. (ix)

Here GOFAI, or the idea of a “general intelligence,” is what Searle is referring to in his discussion of strong AI. As Dreyfus mentions, no one has yet been able to prove that this type of intelligence is impossible to achieve in a computer, but progress in strong AI research has been quite disappointing. While Dreyfus has been a well-known critic of AI research for many years, beginning with *What Computers Can't Do* published in 1972, even well-known advocates of AI research have begun to express their doubts. John McCarthy, for example, in his FAQ “What is Artificial Intelligence” which is housed online at Stanford University, lists the following question and answer as part of his collection:

Q: How far is AI from reaching human-level intelligence? When will it happen?

A: A few people think that human-level intelligence can be achieved by writing large numbers of programs of the kind people are now writing and assembling vast knowledge bases of facts in the languages now used for expressing knowledge. However, most AI researchers believe that new fundamental ideas are required, and therefore it cannot be predicted when human level intelligence will be achieved. (online)

This dissertation does not answer the question of whether or not strong AI, or human-level AI, or GOFAI, is possible. This question is one in which entire volumes of books have attempted to address; none so far have presented a simple answer. It is my own opinion that we do not know nearly enough about human intelligence yet to even attempt to try simulating this intelligence artificially in a machine environment. What this dissertation does do is present some new fundamental ideas about using Internet technologies and XML to achieve a weak AI that can act as a tool to help study both how we think and how we internalize data to create information and knowledge. My hope is this new idea is something along the lines of what McCarthy is referring to in his question and answer pair above. In order to express this new idea about weak AI, I consider the idea of intelligence from a group perspective and examine the idea of intelligence as a group or system entity rather than as an individual computer entity. To accomplish this change in perspective, I depend on a model of distributed intelligence in which intelligence is exhibited as an emergent property of a cooperative system.

Intelligence through Distributed Cognition

EDNA-E functions as a distributed cognition mechanism, which allows this tool to overcome some of the limitations which have acted as barriers against traditional AI research as described in Chapter Two. In his Bank of Canada narrative ethnography, Graham Smart writes about the ways in which group narratives can coexist with human intellect to create intelligent emergent knowledge properties from collections of data. He writes that "... the marriage of judgment and mathematical modeling in the specialists' storytelling can be viewed as a case of distributed cognition, where a professional group, to accomplish its work, has created an artifact combining technology and a symbol system, with the 'intelligence' of the artifact merging with the insights and intuitions of the human intellect—a meshing of mind and tool—in a process of collaborative knowledge making" (261). Whereas the Bank of Canada model used an internal and private computerized model for combining group narratives, though, the *EDNA-E* model is meant to serve as an open source and customizable Web-based model that can be adapted to fit any organizational group's knowledge management needs.

EDNA-E's intelligence, then, is an emergent property that is observable through the synthesis of a larger narrative from individual stories. Like the Bank of Canada *White Book* described by Smart, knowledge is not automatically transformed from data but is rather the result of combined efforts from both the technology and the human operators of that technology. Smart's comments about the *White Book* also apply to *EDNA-E*: "The knowledge inscribed in the document derives from an intelligence distributed across the people, technology, symbol systems, and structured social interaction that constitute the Projection Exercise" (265). In

Smart's ethnography the Projection Exercise was the meeting of *White Book* authors with the bank executives who would be reading the document; *EDNA-E*'s Projection Exercise is the combining of related stories into a larger story by the Story Administrator.

The active process of knowledge building can also be considered intelligent here. As the knowledge for a particular story may be expanded or revised through an original author's edits or a new author's comments, data, information, and knowledge are all in states of constant movement. Recalling the finite state mechanism shown for the *Dragon* stories in Chapter Three (Figure 12), we can create a similar mechanism for *EDNA-E*'s story mechanics in which stories are continually moved from one state to another. At any given moment, one visitor's data may be another visitor's information; with a revision to that story or a link to another story the data may move into a form that contains expert knowledge. These stories can also take on new purposes when opened up across different genres of distribution when they are released to different groups. Smart explains, "... narrative can function an evolving—in a sense, organic—knowledge-bearing representation that 'moves,' cumulatively and chronologically, across a number of different genres, both distributing cognition among and synthesizing intellectual contributions from colleagues working in different organizational sites and time frames" (269). The possibilities of such a framework are limitless in terms of future story applications. For this reason, it was necessary to devise a strict control system for the release of gathered stories. The complexity of the administrative options discussed later in this chapter attests to the fact that such a control system is indeed in place for this software.

Finally, the narrative is important to building intersubjectivity, which I explain in Chapter Four as the common meeting ground between private cognition and public ideologies. In this

third chapter, Smart explained how a narrative model enabled intersubjectivity between bank economists: "... narrative, as a contributor to intersubjectivity among the bank's economists, plays a key role in establishing the realm of shared understandings that supports collaborative knowledge making" (269). By allowing experts and non-experts to converse with one another through stories and the shared ideologies embedded within those stories, *EDNA-E* widens the boundaries of intelligent discourse beyond the traditional markers suggested by the differences in academic and professional training for various fields.

Software Overview

Both the original version of this software, *EDNA*, and the second-generation version, *EDNA-E*, were written using the PHP programming language. PHP is a recursive acronym for "hypertext preprocessor" (PHP.net online). To program in PHP, a programmer simply embeds bits of PHP code surrounded by specialized tags that the PHP server-side module program then knows to examine. Code inside of these tags is then executed, and a file containing only HTML and other browser-side languages such as JavaScript is returned to the user's web browser. The software was built and tested using PHP 4.3.1 running on a Linux 2.40.20-20.7 operating system with the Apache 1.3.27 httpd web server. The software is composed of:

- Approximately 15 PHP script files for general user functions
- Approximately 25 PHP class files containing object definitions
- Approximately 10 PHP script files for administrative user functions
- A templates directory with 10 customizable XML file templates

- A dictionary directory containing two subdirectories: a parts of speech subdirectory with approximately 10 parts of speech XML dictionaries, and a thematic subdirectory with approximately 20 thematic dictionaries
- An XML directory with various default XML files, XML script files, and the main XML settings file which contains database settings, file paths, story administrator information, and other critical information
- An include directory with a class loader script and an include file containing a few minor JavaScript functions
- A documentation directory containing program documentation
- A debug directory containing the main program's XML error log
- Other directories for images and user-generated stories
- A "readme.txt" file containing information on initial setup and installation.

The other possibility for designing this software would have been to use the purely object-oriented language Java. I ended up choosing PHP over Java for several reasons. Most importantly, I was more comfortable writing in PHP as I have already developed several other web applications using this software. Secondly, the elements of object-oriented programming not currently supported by PHP 4.4.1, such as multiple inheritance and object abstraction, were not features I felt were critical to the core functionality of this program. PHP also has some XML parsing functions I wanted to use based on the Simple API for XML (SAX). Finally, I wanted most of the processing to occur on the server side to minimize any Internet browser plugins or database drivers that would have needed to be downloaded by users or story

administrators. In its current state, all you need to use *EDNA-E* is an HTML 4 compliant Internet web browser.

There are only five JavaScript functions used, which are included for simple tasks like browsing back one page in the history file (clicking a back button to mimic the back function of a web browser), closing a window, redirecting a user after they select an element from a drop down menu, or opening a small popup window. For all database functionality, the MySQL version 3.23.49 server was used. For automatic generation of software documentation, the PHPDoc software was installed and run on my source code files.

Rather than discussing each of these source files in detail, which would be an enormous task, I have chosen to instead walk through some typical user and administrator sessions later in this chapter. For access to the source code and full PHPDoc documentation, I have provided the two Internet links listed in Table 1. With that in mind, I will proceed to explain some of the general concepts and ideas used to design this software.

Table 1: *EDNA-E* Program and Documentation Links

Link Description	URL Address
<i>EDNA-E</i> Homepage	http://www.textsandtech.org/~rudym/edna-e/
<i>EDNA-E</i> Documentation	http://www.textsandtech.org/~rudym/edna-e/documentation/

Planning and Design

EDNA-E's knowledge-base does not need to be very sophisticated. Unlike the computerized storytelling systems discussed in Chapter Four, *EDNA-E* does not need to write its

own stories. Rather, it needs only to collect stories from users, to decide how to categorize and classify these stories, and then to distribute them to the appropriate groups. Computerized story *telling* systems require several types of knowledge, including “rhetorical knowledge, story-world knowledge, and common-sense knowledge” in order to adjust the content and style of generated stories (Pérez y Pérez and Sharples 16). Computerized story *collection* systems, on the other hand, only need a type of common-sense knowledge that is imposed by a script mechanism. This knowledge helps the computer to determine which types of stories should be accepted for a given event stored in the program’s database.

Deciding on which data structures should be used to store the gathered knowledge, however, is slightly more complicated. Hubert Dreyfus explains that when choosing a set of representation primitives (a process known as “ontological engineering”) the data structures “must represent objects and their properties, individuals, collections, space, time, causality, events and their elements, agency, institutions, and oddly, from a traditional philosophical point of view, recurrent social situations such as dinner at a restaurant or a birthday party” (xix). In other words, a single data structure object is not adequate here. In order to effectively encompass this variety of ontological data, various objects for scripts, stories, and events are all necessary. The stories themselves, as higher-order data structures, can then encapsulate more complex and abstract information such as causality, agency, and specific social knowledge.

The overall approach to building this web application was to use an iterative development model. This model, also known as a cyclical or spiral model or a Rapid Application Development (RAD) model, works to “develop an application by successive approximations” using analysis, design, code, and test phases of development (Harmon and Watson 29). This

means that during the first iteration of development, the core functionality of the system was implemented. Then, in each subsequent cycle, the model was further tested, refined, and improved until the final production model was achieved.

The first iteration of development (including analysis, design, coding, and test phases) yielded a functional but ineffective application. This model, which was named *EDNA* rather than *EDNA-E*, was intended to be an “English Department Narrative Agent.” This original model’s logo image is captured in Figure 30. If nothing else, the first attempt at designing this software gave me an idea of how I wanted to design my visual interface and other aesthetic components of the application. The visual metaphor I finally chose was that of the grandmother. This metaphor seemed appropriate to personify my computerized knowledge management system for two reasons. For one thing, grandmothers (as well as grandfathers) are often associated with good stories; the stereotypical storytelling grandparents can be found in what Schank identifies as our “culturally common” stories (37). Secondly, the image fit in well with previous incarnations of computerized information processing systems such as the natural language processors *ELIZA* and *ALICE* discussed in Chapter Two. *EDNA*, who wears a more distinguished look than her predecessors, therefore draws on some of the characteristics of her well-known ancestors for her visual design.

Two additional notes about the visual design also need to be stated. Firstly, all images, graphics, and icons in this application were created for me by graphic designer Carole McDaniel. Carole was able to work up a design based on the description of the grandmother figure I gave her. Secondly, the short paragraph above is the only place in which *EDNA* is actually referred to

as a “she.” In the rest of this chapter, the application is referred to as an “it,” which again emphasizes its role as a weak AI application and not a mechanism of human-level intelligence.



Figure 30: *EDNA* Logo (Phase 1 RAD)

This initial web application was able to effectively login and register users (without any approval needed), collect stories with no limitations, and display those stories listed by author (the author was a user’s username). Unfortunately, I soon found that this model was severely lacking. While it was very simple to enter and read stories, there were notable problems with the software’s mechanics and functionality. The problems with this first-pass instantiation included the following major issues:

- The name implied that the model was only intended to work within an English Department. Also, the term “agent” was misleading in suggesting the system used agent-based technologies such as Java aglets to collect and disseminate stories. This problem was simple enough to fix simply by renaming the application to *EDNA-E*; the new name

also takes into account the new features implemented in the next iteration of this software.

- Users were unwilling or hesitant to release their stories to the public. This prompted me to create a means for controlling access to stories, which is described later in this chapter.
- The collected stories were too open-ended and not very useful. This suggested some means of control for story boundaries was necessary, and led to the researching and implementation of a script-based model. This observation also led to the creation of organizational, divisional, and sub-divisional categories.
- The collected stories were difficult to relate to one another using textual analysis methods alone. This led to the event-driven model used in *EDNA-E* which requires users to respond only to acceptable events listed in the database which are added by the story administrator.
- The collected stories were difficult to analyze for thematic elements. This observation led to the creation of the thematic dictionaries, as well as the parts of speech dictionaries used to automatically extract the story's keywords, which are used in *EDNA-E*.
- Inappropriate stories were entered during the testing phase. This led to the requirement for the story administrator to approve both new users and new stories before they are displayed on the webpage.
- Despite the flexibility allowed by collecting textual stories in a database, the stories were still marked up using HTML only. This format is not acceptable for sharing these stories with other web applications or other Semantic Web-enabled devices. *EDNA-E* therefore includes XML-markup features including the ability to import settings from XML files.

- It was confusing to determine which functions should be limited to which users in the original design. The next iteration of the development cycle therefore involved UML models to plan out how to limit access and program functionality to either a standard user or an administrative user known as the story administrator.

At this point, a functional application was created and some of the preliminary problems present in this software had been identified. This group of problems presented an excellent set of metrics with which to carefully define a problem for the next design phase. The main design question I identified was this: how can I build a robust, useful, and secure web application for dispensing embedded knowledge in narrative form throughout a distributed organization? This question was suitable for moving the development cycle into the next design phase.

It was then necessary for me to refine this existing application into a more useful form. As described in earlier chapters, *EDNA-E*'s theoretical design is very conducive towards the use of classes and objects. This is particularly true of certain knowledge representation schemas such as Marvin Minsky's frame model or Roger Schank's script model. This feature made it simple to extend the original design of *EDNA* into *EDNA-E* simply by adding some new class templates and by slightly revising some of the original classes. Certain elements of the new narrative analysis engine are therefore represented using object-oriented data structures. An object is simply "a complex data type that contains other data types, usually called attributes or variables, and modules of code, usually called operations or methods" (Harmon and Watson 9). This data type can then be called into memory from a class template which allows the object-code to be reused whenever that particular data type is needed.

Objects are also quite useful for simulating multiple computers when there is actually only one computer available. As Robert Sebesta notes, each object can act as a computing device which can communicate to other computing devices (objects) through message-passing (441). Here messages are simply “the subprograms that define the operations on objects of a class” (Sebesta 438). Using objects and object-messaging, then, is a natural type of programming strategy which facilitates the distributed-cognition model of knowledge management described earlier in this chapter.

Fortunately, there is a robust language available for modeling classes and objects: UML, which is the Unified Modeling Language. UML offers five different types of diagrams which offer different perspectives on solving a given problem using OO techniques. The various types of UML diagrams include (Harmon and Watson 49):

1. Use Case Diagrams
2. Static Structure Diagrams
3. Interaction Diagrams
4. State Diagrams
5. Implementation Diagrams

Some functions of *EDNA-E* are modeled later in this chapter using use case and interaction diagrams to demonstrate how these models can be useful in the planning stages of object-oriented development. An example of a state diagram was shown in Chapter Three with the *Dragon* stories. The additional UML models are simply too complex to include in this chapter; the models I do include will show how UML can help to highlight the operation of specific program functionality.

Building the User Interface

My initial approach for planning the second version of this software was much the same as Curt Herzstark followed when designing the mechanical Curta calculator discussed in Chapter Two. Like Herzstark, I “began with the user interface, rather than letting the mechanism control the design” (Stoll 93). From *EDNA* I had an idea of the visual design I wanted to include; at this point I only needed to decide how I wanted to lay out the program features using these visual elements. The interface I decided upon is shown in Figure 31, with major features labeled and highlighted. The visual design was also slightly modified, with the grandmother figure now displayed on a computer screen and a binary background image displayed along the top image header. The idea with the main user interface was to create an application that was easy to use, and to minimize all non-relevant features as much as possible on each screen. For instance, when first opening the webpage a user will see an option to either register a new account or to login; the additional links to actually use the software will not appear until their login has been validated.

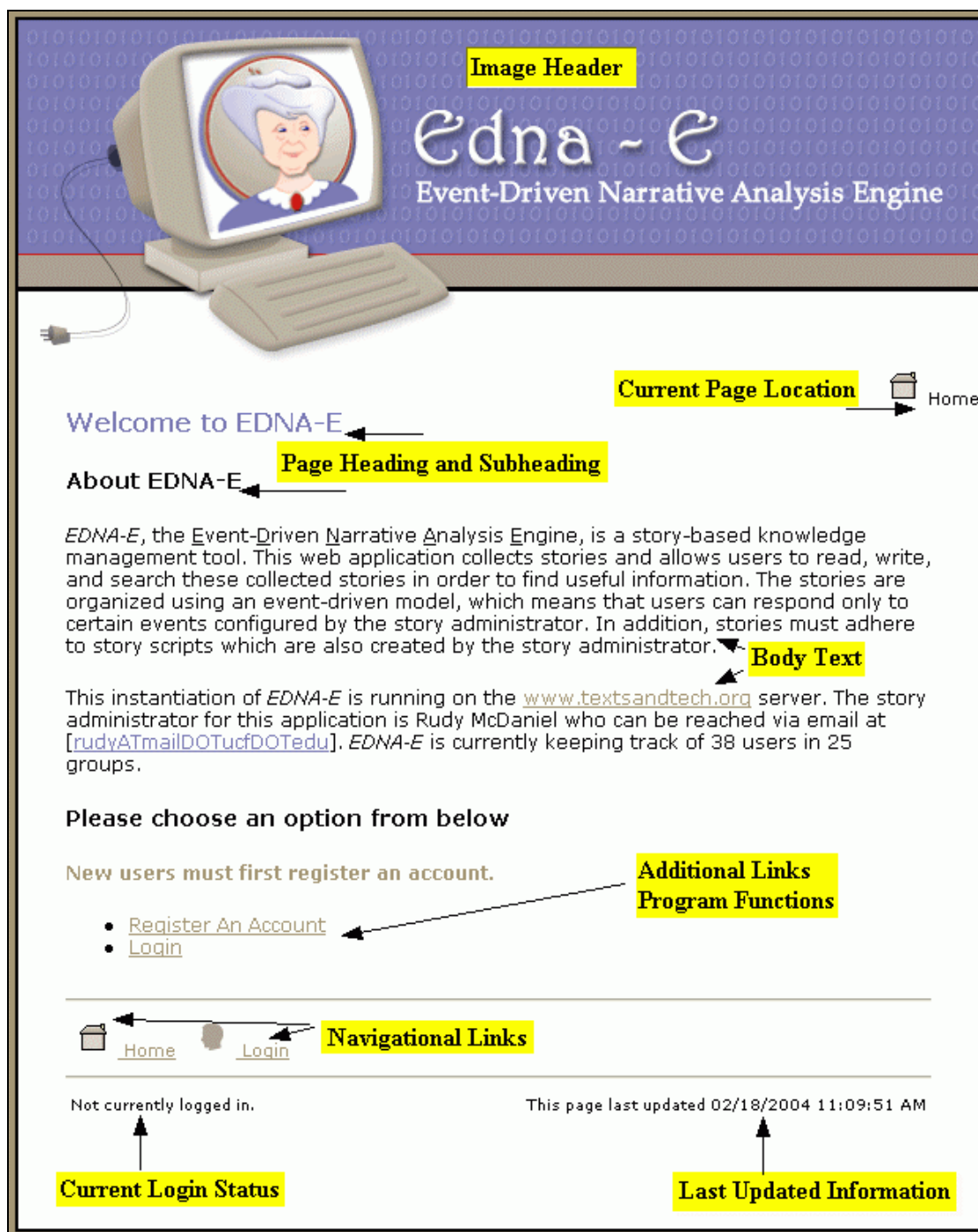


Figure 31: *EDNA-E* User Interface (Phase 2 RAD)

An example of this same page (the index or home page) after the user has logged in is shown in Figure 32. Note the additional links and program features that appear at this point after the user has logged in and the system has verified their account. In this case, there is also a link to administrative functions since this particular user is also the story administrator for this installation. Individual web pages or sections of web pages are also referred to as panels in this chapter in order to functionally separate different components of the software. For example, the navigational links shown in this screen capture and in Figure 31 are also referred to as control panel icons in other parts of this chapter. The term “panel” is used interchangeably with the terms “web page” or “web page feature.”

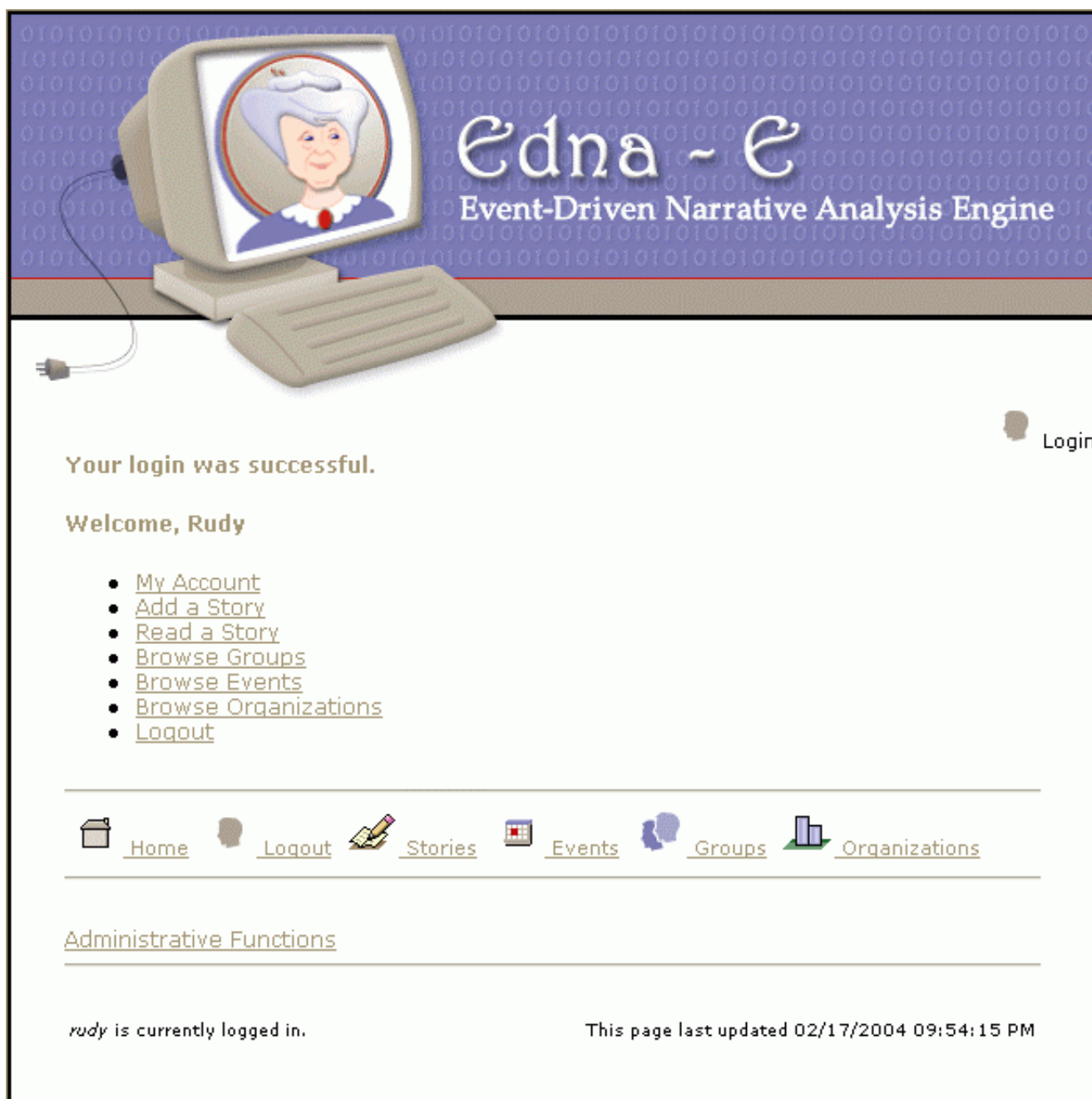


Figure 32: EDNA-E User Interface after Login

Additional Design Considerations

With an interface decided upon, I needed to expand on some of the limitations and try to predict some of the user reactions to this program's features. Two of the most pressing problems I identified from the start were controlling the scope of allowable stories (briefly mentioned in Chapter Three) and reassuring users of their personal privacy when submitting stories. These two concerns suggested that the mechanism should be more complex than I initially considered; there now needed to be some form of security permissions mechanism to ensure that certain stories can only be read by certain audiences. Also, there needed to be some way to enforce certain topics for each story; this was accomplished by accepting only stories about pre-established events as well as by specifying which events could be written about using the scripting mechanism. Both of these program features are described in detail later in this chapter.

It is also important to identify a storytelling manager who can analyze the stories and determine whether or not these stories are valuable and should be enabled for viewing and reading in the system. Such a manager is especially important in the beginning phases of story collecting, since the general idea of an acceptable story has not yet been understood by the system's users. After the story has been approved, it is now transformed into a form of knowledge. As Stephen Denning explains, "Now that the experience has been identified as valuable, it can be captured and stored in the electronic knowledge base for use by others—inside or outside the organization—who are confronted with a similar issue" (124). When this approval mechanism is combined with the permissions and user/group level functionality of *EDNA-E*, a powerful tool for both managing and distributing knowledge begins to emerge.

Graham Smart, in Chapter Four, also identified certain characteristics useful for a narrative knowledge management system. For instance, when discussing the creation of the *monetary-policy story*, he wrote about the tendencies of executives to supplement the official *White Book* stories with their own judgments and expert observations about a given issue. One executive wrote “that in certain instances, the executives may simply apply their own judgment as a ‘mental overlay’ on the *story* in the *White Book*” (266). It is therefore useful to specify a mechanism for allowing these “mental overlays” to exist in *EDNA-E* in order to allow expert users to insert their own observations and suggestions into the framework of a given story. This task is accomplished using the “Allow Story Comments” feature that is enabled by specifying the appropriate option when creating a particular script.

Use Case Diagram

Use case diagrams are “used to describe the main processes in a system and the interactions between the processes (use cases) and external systems or individuals, called actors” (Harmon and Watson 49). The actors for *EDNA-E* are the various types of users that will interface with the system. *EDNA-E* has two primary actors: the story administrator and the normal class of user, who adds stories using predefined scripts created by the story administrator. *EDNA-E*’s use case diagram is shown in Figure 33.

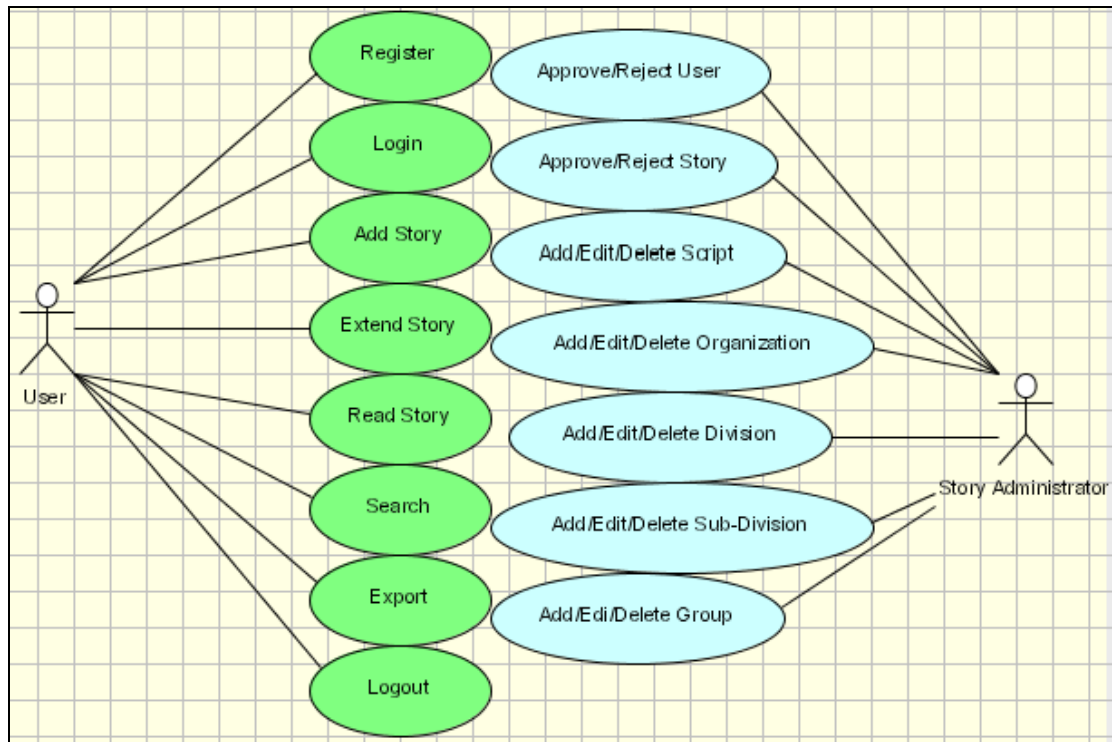


Figure 33: EDNA-E Use Case Diagram for Major Functions

A use case diagram basically shows a group of users (actors) who will be using the software along with the common functions that will be available to those users. In this case, the general user for *EDNA-E* is described as the actor “User” or the non-proper version “user.” This type of user will have the following functions available to them, as revealed in Figure 34.



Figure 34: User Functions

- Home: This icon returns the user to the default index page/homepage.
- Register: The register function allows a user to create an account with a username and password. The registration link is available under the “Additional Links and Program Features” section of the website for users that have not yet logged in.
- Login: This icon enables a user to login with their username and password after their account has been approved and enabled. The *Login* icon is shown in the user control panel if a user is not currently logged in.
- Logout: This icon logs a user out of the system, deletes all session information (session information includes the user’s name, email address, and other identifying information), and resets certain system session variables. A user logs out by clicking on the *Logout* icon on the user control panel.
- Stories: Story functions include displaying, adding, and searching stories. These functions are accessible through the *Stories* icon on the user control panel. After a particular story has been displayed, a user will also have the option to edit that story if they are the owner of the script that the current story was created with.
- Events: The *Event* icon allows a user to view events that have been added by the story administrator.
- Groups: The *Groups* icon allows a user to view groups that have been created by the story administrator.

- Organizations: The *Organizations* icon allows a user to view the organizational hierarchy created by the story administrator.

As the program matured, additional minor functions were added to the user interface but the above panel reflects the core user functionality.

The other type of user in *EDNA-E* is the “administrative user.” This administrative user is also referred to as the “story administrator.” This type of user sets up the various groups, organizations, and divisions for a particular organization. This user also adds events which can be responded to in story format. Finally, the story administrator performs bookkeeping type functions such as approving users and stories and keeping organizational information updated.

The use case UML diagram in Figure 33 shows that a story administrator is a critical component of the narrative analysis engine. The story administrator authenticates accounts, adds events and user groups, and, most importantly, verifies stories. The presence of a human user in such a process is invaluable. For one thing, a human user is able to capture details about individual stories and make connections amongst stories that would be difficult if not impossible for a computerized agent. As Rafael Pérez y Pérez and Mike Sharples explain, “These elements, of interestingness, appropriate content, narrative flow, coherence, suspense, and originality, cannot be captured by objective measures (for example of syntactic complexity). They are the elements of literary criticism and in order to provide a comparative measure they need to be assessed, either singly or in combination, by human readers” (17). While they were speaking of this concept in terms of computerized story generators, their statement regarding automated analysis is equally true for pre-existing stories that have been entered by human users. A person

can easily enter a story that is as bad as, or worse than, a computer-generated story. These types of stories would be both distracting and useless for those users searching for stories with thematic elements that might happen to match those found in the bad story. It is the job of the story administrator to identify, disable, and effectively filter these types of stories from the active group of stories processed by the software.

The story administrator user also solves another major problem of Roger Schank's script theory as identified by Hubert Dreyfus in his critique of artificial intelligence research. Dreyfus writes of Schank's script theory: "Schank's proposal leaves completely unanswered the problem of how we *do* choose the right script" (45). In my model this process is simple enough: we don't. The appropriate script is chosen for us by the story administrator and linked specifically to the appropriate event that has been created in order to solicit a story from us.

The story administrator has the following procedures available to them, as grouped in the control panel screen capture in Figure 35.



Figure 35: Story Administrator Functions

- Home: This icon returns the administrator to the default administrative index page/homepage. This is a different page than the user-level homepage. The default administrative index page is shown later in this chapter in Figure 37.

- **Events:** The event functions enable the story administrator to release events or to delete or modify existing events. Since stories can only be added once an appropriate event has been created, this is an important feature. Event functions are available in the *Events* section of the administrative control panel.
- **Organizations:** These functions are used for controlling organization objects. Organization objects include organizations, divisions, and subdivisions. Each of these objects also has a corresponding group. Organizational functions are found in the *Organizations* area of the story administrator control panel.
- **Stories:** The administrative story functions provide a means for ensuring only appropriate stories are posted to the narrative analysis engine if automatic analysis is not able to determine whether or not a story is valid. Story features are available through the *Stories* icon on the administrative control panel.
- **Scripts:** Script functions are the most complicated and probably the most important functions available to the story administrator. These functions allow the story administrator to create scripts which directly specify allowable story elements. Such elements include protagonists, antagonists, central themes or concerns, and time and place settings. Script options are configurable by way of the *Scripts* icon in the administrative control panel.
- **Groups:** The group features control group memberships and group names. These items are available from the *Groups* icon on the control panel.
- **Edit XML:** The *Edit XML* control panel icon leads to another important administrative tool. This tool allows the story administrator to directly edit XML files used by *EDNA-*

E. In addition, administrators can create new XML files to be used as thematic dictionaries or part-of-speech dictionaries. These dictionaries are described in more detail later in this chapter.

- **Config:** The *Config* icon is simply a link to a webpage which prints out current configuration settings and allows the story administrator to click on an additional link which will load this file into the XML editor tool. This icon is useful for debugging purposes and to check system settings such as file and directory paths and default filenames.
- **Users:** User functions allow the story administrator to control who is creating accounts and registering to use the system. These features become available by clicking on the *Users* icon in the administrative panel.
- **Import:** The *Import* icon is used to import large batches of users into *EDNA-E*. This feature is useful when an administrator wants to register a large group of people at once without having to enter in each user manually. The import feature requires the list of users to be marked up in an XML file; an example file containing the correct structure is provided on the file importer web page. For this import file, a DTD is not required.

Sequence Diagram

Sequence diagrams are used in UML to show how messages are transferred between various objects. This type of diagram, which is a subset of the broader class of interaction diagrams, “allows you to study how a set of objects interact in time” (Harmon and Watson 51).

A sequence diagram for *EDNA-E*, which shows the registration process and how this process occurs in temporal succession, is shown in Figure 36.

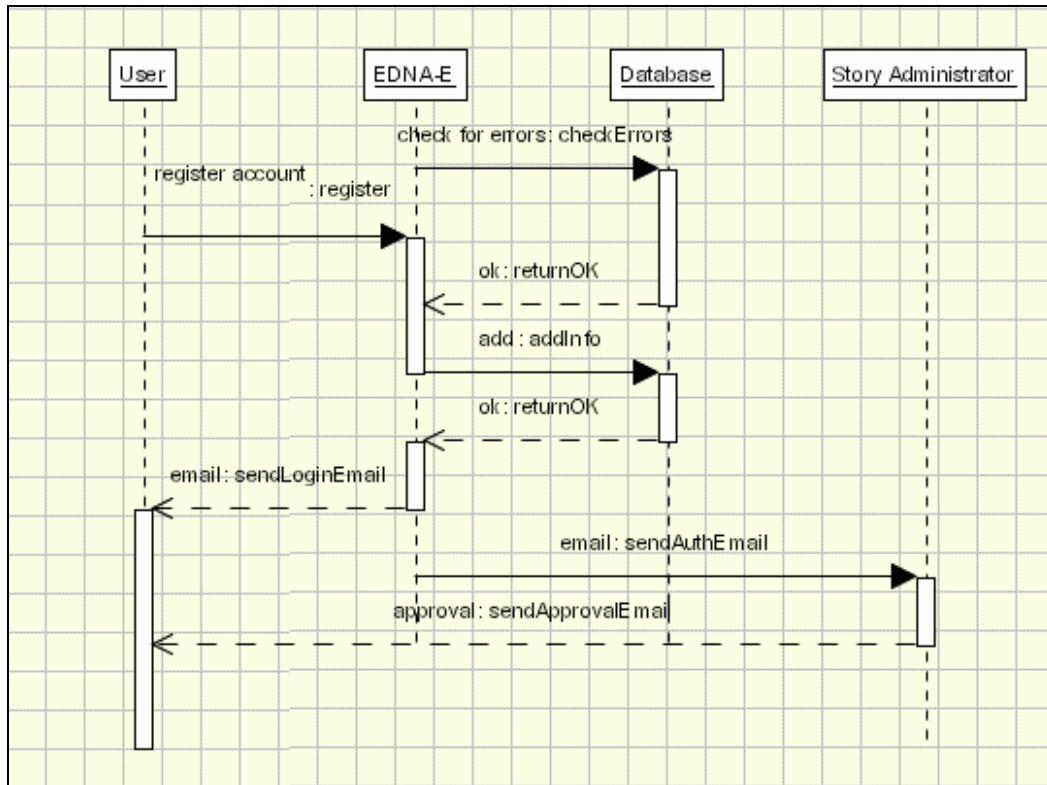


Figure 36: Sequence Diagram for *EDNA-E* Registration

Later in this chapter, I will show how the registration sequence diagram translates from XML into an actual web form. From this point on, all different application screen shots will be referred to as panels. Figure 52, for example, shows the registration panel which is presented to a new user wishing to register an account with *EDNA-E*.

For the remainder of this chapter, I will use a logical order to describe each panel as they would be accessed in a typical session. This means that I will first show an administrative session which will prepare a suitable environment for storytelling. For this session, I will

assume a specific task, and I will assume that I am the story administrator for this task. As the story administrator, I want to create an environment in which University of Central Florida (UCF) Texts and Technology (T&T) students can share stories with one another. These stories should be about these students' experiences in certain core Texts and Technology courses. The next stage of this process, in which a user will actually add a story, will describe screenshots of each panel as they would appear in sequence to a T&T student user accessing the system for the first time. Thus the user sequence will contain two phases: a registration phase and a story addition phase which occurs after the user's registration has been approved. To save space, the header images are cropped out of all screen captures except for the first program screen shot in the story administrator session. Another brief administrative session follows, in which I perform the administrative role of approving a new user's account. Then, the final phase consists of the user adding a story after their account has been approved.

The Story Administrator Session: Phase One

Administrative Homepage

Upon logging in, since I am an administrative user, I will see a link at the bottom of the home page labeled "Administrative Functions." This link will bring me to the administrative homepage, and will open up the administrative control panel as shown in Figure 37. From this page, I can build groups of users, create new events, or add organizations. In this case, since groups can be automatically created when organizations, divisions, and subdivisions are created,

let's assume that I am going to add a new division. Keeping our task in mind, the division I need to create will be "T&T Students."

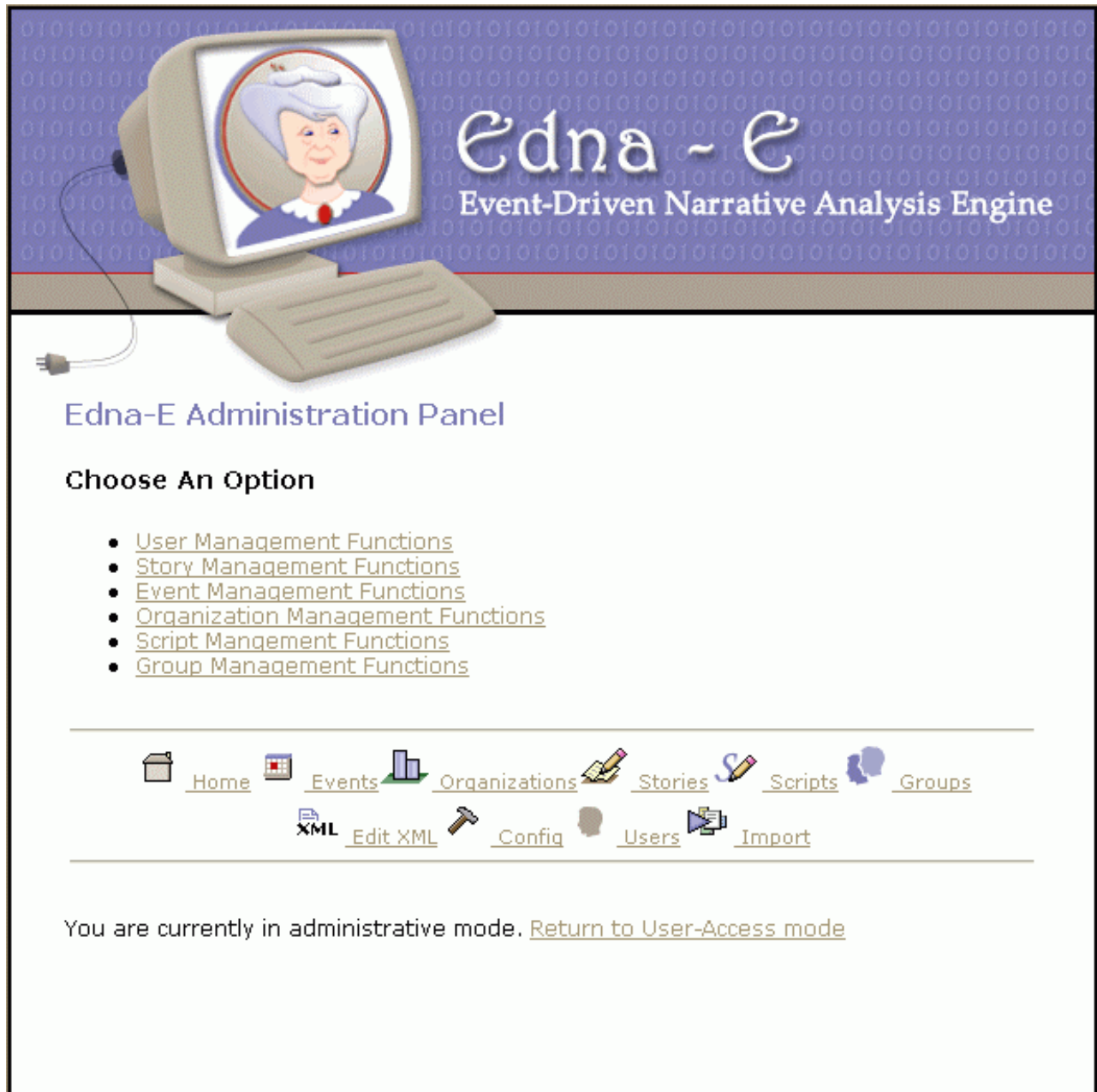


Figure 37: Administrator Homepage

Organization Panel

After accessing the organizational panel, I am given the option to add a new organization, division, or sub-division. I am also given the choice of automatically creating a new group which corresponds to this addition. At the bottom of the panel, a hierarchy showing the existing organizations, divisions, and subdivisions is displayed. This hierarchy is shown in Figure 39. In this hierarchy, we see that a University of Central Florida organization already exists. I can then scan divisions and subdivisions to see if the “T&T Students” group exists as well. There is in fact a “Texts and Technology” group, but no “T&T Students” group. I will therefore add “T&T Students” as a subdivision using the last “Add” button shown in Figure 38. For the “Parent Division” option I will choose “English Department” to keep “T&T Students” grouped at the same level in the hierarchy as “Texts and Technology,” which may be a more inclusive group containing faculty as well as students. The results of this operation are shown in Figure 40. Since the “Add Group for this Subdivision” checkbox was also checked, a new group for this subdivision is automatically generated and I am informed of its creation.

Add An Organization

Name

☒ Add Group for this Organization

Add A Division

Name

Parent Organization

☒ Add Group for this Division

Add A Sub-Division

Name

Parent Division

☒ Add Group for this Subdivision

Figure 38: Add Organization Panel

Existing Organizations

Organization Name	Date Added	Date Updated	Delete
Sparkletech	02/10/2004	02/10/2004	DELETE
---Management	02/10/2004	02/10/2004	DELETE
-----General Management	02/11/2004	02/11/2004	DELETE
---Technical	02/10/2004	02/10/2004	DELETE
-----Network Administration	02/10/2004	02/10/2004	DELETE
-----Documentation	02/15/2004	02/15/2004	DELETE
UCF	02/10/2004	02/10/2004	DELETE
---English Department	02/10/2004	02/10/2004	DELETE
-----Texts and Technology	02/10/2004	02/10/2004	DELETE
-----Creative Writing	02/10/2004	02/10/2004	DELETE
-----Literature	02/11/2004	02/11/2004	DELETE
-----Technical Writing	02/11/2004	02/11/2004	DELETE
---Digital Media	02/14/2004	02/14/2004	DELETE
-----Faculty	02/14/2004	02/14/2004	DELETE

Figure 39: Existing Organizations Panel



A new sub-division **T&T Students** has been added.

A new group **UCF English Department T&T Students** has been added.

Figure 40: Adding Subdivision Message

At this point, I could continue on to add some events since the group was automatically created. This story administrator is cautious, though, and wants to double-check to make sure the group was actually created. To do that, I click on the *Groups* icon to access the group administration panel shown in Figure 41. Here groups are shown in alphabetical order, and our group doesn't exist within the first ten groups listed in the panel. I can then browse to the appropriate page using the triple right-arrows "> > >" and see that a group has indeed been created for "T&T students." The table informs us that no members exist within that group, which makes sense because it is newly created.

Group Administration Panel

Group Administration				
Add group members by using the edit feature next to each group.				
Name	Members	Last Update	Add Delete	
UCF	33	02/11/2004	Edit	Delete
UCF Digital Media	2	02/15/2004	Edit	Delete
UCF Digital Media Faculty	2	02/15/2004	Edit	Delete
UCF English Department	32	02/11/2004	Edit	Delete
UCF English Department Creative Writing	9	02/10/2004	Edit	Delete
UCF English Department Literature	11	02/11/2004	Edit	Delete
UCF English Department T&T Students	There are currently no users in this group.	02/19/2004	Edit	Delete
UCF English Department Technical Writing	14	02/11/2004	Edit	Delete
UCF English Department Texts and Technology	4	02/19/2004	Edit	Delete
<<< >>>				

Figure 41: Group Administration Panel

After I have verified that the group does indeed exist by clicking on the *Groups* icon, I need to create some events for T&T students to write stories about. Clicking on the *Events* icon reveals several existing events with various group owners. As shown in Figure 42, these events can also be edited or deleted if the story administrator deems it necessary. The appropriate link at this point, however, is the “Add New Event” link shown at the bottom of the events panel.

Event Name	Event Group	Added	Updated	Edit	Delete
Email and computer virus problems.	UCF	02/19/2004	02/19/2004	[edit]	[delete]
Lang contract renewal	Sparkletech Management	02/19/2004	02/19/2004	[edit]	[delete]
Loss of Internet	Sparkletech	02/16/2004	02/18/2004	[edit]	[delete]
Sparkletech newsletter	Sparkletech Technical Documentation	02/19/2004	02/19/2004	[edit]	[delete]
Writing a science fiction novel	UCF	02/19/2004	02/19/2004	[edit]	[delete]
Writing a T&T Dissertation	UCF English Department Texts and Technology	02/16/2004	02/16/2004	[edit]	[delete]
[Add New Event]					

Figure 42: Events Panel

Add Event Panel

Clicking on the “Add New Event” link brings up the “Add an Event” form shown in Figure 43. From this panel, I observe that the event group select list has been auto-populated by the system. My newly created group (which was in fact created automatically by creating a subdivision) is shown in this list in the appropriate position in alphabetical order. After selecting this entry, I can then add some events to the database.

Add An Event

Event Name

Event Group UCF English Department T&T Students ▼

You are currently in UCF English Department T&T Students

[Scripts](#) [Groups](#) [Port](#) [mode](#)

Event Group List:

- Administrators
- Everyone
- Public
- Rudys Dissertation Committee
- Sparkletech
- Sparkletech Management
- Sparkletech Management General Management
- Sparkletech Technical
- Sparkletech Technical Documentation
- Sparkletech Technical Network Administration
- UCF
- UCF Digital Media
- UCF Digital Media Faculty
- UCF English Department
- UCF English Department Creative Writing
- UCF English Department Literature
- UCF English Department T&T Students**
- UCF English Department Technical Writing
- UCF English Department Texts and Technology

Figure 43: Add an Event Panel

Assuming I add the events “Research Methods in Texts and Technology,” “Theory of Texts and Technology,” and “Visual Texts and Technology,” my event-list would now look as it appears in Figure 44. Here the newly created groups are highlighted for emphasis. After checking the information to make sure a valid event name and group have been selected, this panel will then tell inform me as to whether or not the event has been added to the database. If it has, the database record id number will be printed to the screen as shown at the top of Figure 44.

Checking your information..			
This event has been added to the database as id 27.			
Event Name	Event Group	Added	Updated
Email and computer virus problems.	UCF	02/19/2004	02/19/2004
Lang contract renewal	Sparkletech Management	02/19/2004	02/19/2004
Loss of Internet	Sparkletech	02/16/2004	02/18/2004
Research Methods in Texts and Technology	UCF English Department T&T Students	02/19/2004	02/19/2004
Sparkletech newsletter	Sparkletech Technical Documentation	02/19/2004	02/19/2004
Theory of Texts and Technology	UCF English Department T&T Students	02/19/2004	02/19/2004
Visual Texts and Technology	UCF English Department T&T Students	02/19/2004	02/19/2004
Writing a science fiction novel	UCF	02/19/2004	02/19/2004
Writing a T&T Dissertation	UCF English Department Texts and Technology	02/16/2004	02/16/2004

Figure 44: Adding Events Panel

Scripts Panel

At this point, I have now created a new subdivision, “T&T Students,” a new group for this subdivision, and three events which correspond to this new group. I now need to define scripts which will control the types of stories accepted for these events. I access the scripts panel by clicking on the *Scripts* icon in the administrative control panel. This panel lists current scripts along with the events these scripts correspond to. Existing scripts at this point are listed as shown in Figure 45. Much like the events panel shown earlier in Figure 42, scripts in this list can be edited or deleted; new scripts can also be added using the “Add New Script” link at the bottom of this panel. For this particular task, that is the link I want to click.

Scripts					
Script Name	Event	Added	Updated	Edit	Delete
Fairy Tale Script	Writing a T&T Dissertation	02/19/2004	02/19/2004	[edit]	[delete]
General writing script	Writing a T&T Dissertation	02/19/2004	02/19/2004	[edit]	[delete]
Sparkletech Technical Script	Loss of Internet	02/16/2004	02/18/2004	[edit]	[delete]
Writing the Sparkletech Newsletter	Sparkletech newsletter	02/19/2004	02/19/2004	[edit]	[delete]
[Add New Script]					

Figure 45: Scripts Panel

Adding a script involves a five-step process which is displayed on a single web page. To keep this description manageable, each step in the process is discussed individually. In Step One, I simply assign a name for the script and choose an event that this script will control. Like the group select list before, the event list for the script function is auto-populated. For this task, I choose an event to control the “Research Methods in Texts and Technology” event and assign the new script a title, as shown in Figure 46.

Add A Script

Step One: Name and Event Link

Type the script name below, and choose which events this script should control when accepting stories for that event.

Script Name

Script Controls This Event Type

Figure 46: Add Script Step One

In Step Two, I assign permissions to this script. Here I have chosen myself as the script owner, and given myself “R+W” permissions. For the group owner I chose our new group and used the same permissions. This is the step in the script creation process that solves the personal privacy problem mentioned earlier in this chapter. The privacy problem is one that has been mentioned in other types of narrative systems. For example, in his book about springboard stories (see Chapter Four for a detailed discussion of these types of stories), Denning recounts his experiences in attempting to recruit staff members to help him gather new stories to manage knowledge in a particular organization. His immediate problem was that people were unwilling to open up to strangers, which eventually led him to “provide them with an escort, to get the door open—someone to help set up the interview, participate, and cement the social connections” (121). This problem, which can be equated to one of personal privacy, is also very evident in any online system that needs to collect stories from users at one point or another during its operation. I found that using a permissions system was the best way to deal with this problem.

The model I devised for story privacy is shown in Figure 47. This function, which is a simplified version of the permissions process used by the UNIX operating system, allows the story administrator to assign both a user owner and a group owner to a given script. A story which uses that script then inherits those permissions. These permissions, which are specified during the creation of a script, work in the following way. Each script is assigned a user owner with either “R+W” or “R” permissions and a group owner with either “R+W” or “R” permissions. Here the “R” stands for “Read” and the “W” stands for “Write”; the “R+W” designation is therefore both “Read” and “Write” permission. There is no option for “W” or “Write” access only, seeing as a user will need to be able to read a script in order to write a story

using that script. The UNIX scheme provides an additional option for file permissions—“X” or “Execute”—but this permission is not needed in *EDNA-E*. “X” permissions are assumed by all scripts automatically.

Step Two: Access Options

Choose the level of access for stories using this script below. R+W gives read and write access for stories using this script. R gives only read access for stories using this script. W (write only) is not supported.

Script User [Lookup Wizard] Permissions

Script Group Permissions

Figure 47: Add Script Step Two

After specifying permissions, I then need to specify allowable story elements in Step Three of the add script function in the *Scripts Panel*. These elements include the story’s time(s), place(s), protagonist(s), antagonist(s), and concern(s). These are the essential story-frame elements identified by Minsky as I discussed in Chapter Three. A list of elements for each of these terminals is loaded from various XML files which can be edited directly from this screen or using *EDNA-E*’s built-in XML editor. To save space, only the first element box is listed in Figure 48. In the actual program, all five boxes are listed in a linear fashion, each with their own XML file and a checkbox to allow all values to be accepted. For my “Research Methods” event, I will input the following values to script a specific type of story:

- Time Setting: Any time
- Place Setting: At home, in a classroom, or at UCF
- Antagonist: Me (Myself)

- Protagonist: Me (Myself)
- Central Concern: Writing a large document

This type of story will therefore be a first-person narrative in which someone struggles to overcome some internal obstacle to write a large document at home, in a classroom, or at UCF. The multiple allowed elements for the place setting in this example are achieved by holding down the control button when selecting elements. This type of system is infinitely configurable since all option boxes are populated by XML files which are directly editable using the built-in XML editor.

Step Three: Story Options

Hold down control to select multiple allowed elements, OR check the box to allow any value for this story element

Story Element	Allowed Values	Allow All
Time Setting [Edit Defaults]	once upon a time a long time ago during childhood many years ago a few years ago a few months ago a few days ago last week earlier this week earlier this month earlier this year yesterday today earlier today today during breakfast today during lunch today during a break today during dinner a few hours ago a few minutes ago	<input checked="" type="checkbox"/> Allow all times.

Figure 48: Add Script Step Three

Step Four of the script-creation process simply adds support for additional story options, some of which have not yet been implemented. These options are added to the script regardless in order to ensure future compatibility. Features not yet implemented include image, audio, and video uploads to enable multimedia accompaniment for a story using that particular script. For my current task, I check the box to “Allow Story Comments” but I will not enable the other boxes as these features do not seem necessary for this type of story. This configuration is captured in Figure 49.

The image shows a screenshot of a web form titled "Step Four: Miscellaneous Options". Below the title is a prompt: "Choose to allow additional story options below." There are four rows of options, each with a label and a checkbox. The first three options are "Allow Still Image Elements", "Allow Video Elements", and "Allow Audio Elements", all with unchecked checkboxes. The fourth option is "Allow Story Comments", which has a checked checkbox.

Step Four: Miscellaneous Options	
Choose to allow additional story options below.	
Allow Still Image Elements	<input type="checkbox"/>
Allow Video Elements	<input type="checkbox"/>
Allow Audio Elements	<input type="checkbox"/>
Allow Story Comments	<input checked="" type="checkbox"/>

Figure 49: Add Script Step Four

The last step in the script creation process, Step Five, is shown in Figure 50. This step gives me an opportunity to provide additional instructions for the users who will be creating stories from this script. Here I provide some brief supplemental instructions for this script.

Step Five: User Instructions

Type a brief note explaining to users how to write a story using this script.

User Instructions

For this script, write a story about your personal experiences writing your research project for the research methods class. Be creative!

Add Script Clear Script

Figure 50: Add Script Step Five

With all options now configured, I now click on the “Add Script” button to create the script and link it to my “Research Methods in Texts and Technology” event. The output from this operation is shown in Figure 51. At this point several critical functions are performed by *EDNA-E*. These functions include:

- Error-checking. Here the system checks to make sure information is valid and conflicting options have not been selected (you cannot select “allow all” and then select specific options for a particular story element, for example).
- Generating the script’s XML-file. Here the XML file script is generated and previewed on the user’s screen. A DTD for this script is also created and displayed.
- Adding the script to the database. Here the script is copied into a database in order to achieve even more flexibility by using dual-formats for the script.
- Updating the events table. Since the users will only see events which have scripts associated with them when they add stories, the system needs to update the events table to reflect the fact that the “Research Methods in Texts and Technology” event now has a

script associated with it. After this operation has been performed, users will be able to add stories for this event.

```
Checking your information..

Preview Script File

Your script file has been generated below.

<? xml version="1.0" ?>
<script>
<script_name>Writing A Research Paper</script_name>
<event_link>25</event_link>
<user_owner>rudy</user_owner>
<user_owner_permissions>R+W</user_owner_permissions>
<group_owner>UCF English Department T&T Students</group_owner>
<group_owner_permissions>R+W</group_owner_permissions>
<times_allowed>all</times_allowed>
<places_allowed>at home;in a classroom;at UCF</places_allowed>
<protagonists_allowed>Me (Myself)</protagonists_allowed>
<concerns_allowed>Writing a large document</concerns_allowed>
<antagonists_allowed>Me (Myself)</antagonists_allowed>
<image_enabled>FALSE</image_enabled>
<video_enabled>FALSE</video_enabled>
<audio_enabled>FALSE</audio_enabled>
<comments_enabled>TRUE</comments_enabled>
<user_instructions>For this script, write a story about your personal experiences
writing your research project for the research methods class. Be creative!
</user_instructions>
</script>

File script_writing_a_research_paper.xml has been created. 1154 total bytes
written to file.

This script has been saved as script_writing_a_research_paper.xml in the XML
directory

This script has been added to the database as id 20.

The event table has been updated to reflect this script.
```

Figure 51: Script Output

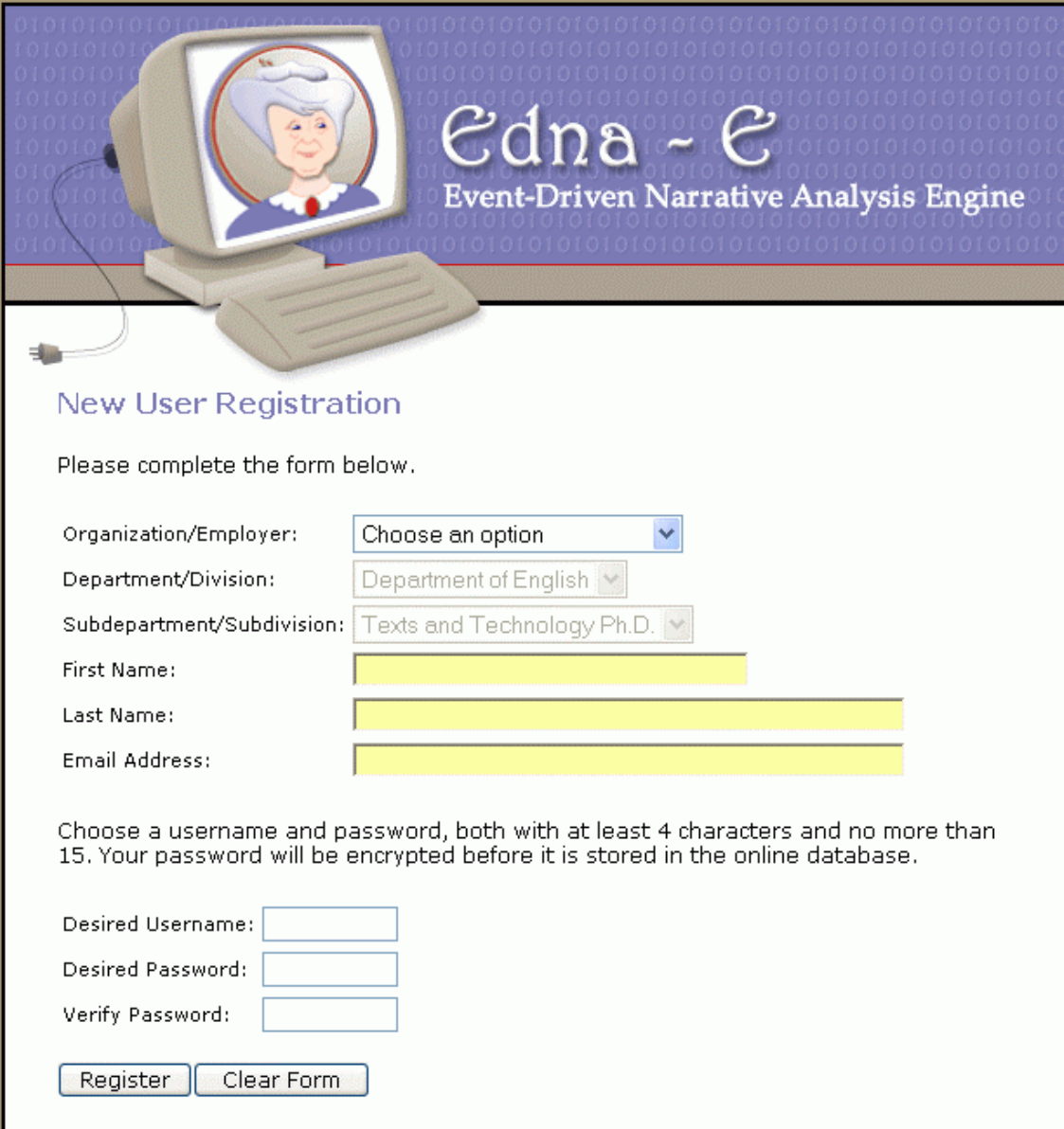
Now that a script has been created for my event, I am done with this first administrative session. Though it may seem like an arduous task when reading about it, the process is actually quite simple and painless. The following session, including adding a new subdivision, group, event, and script, can typically be accomplished in 2-3 minutes, assuming the XML data files for the story elements have already been populated. Even if these files have not yet been created, it is a one-time effort and they will remain in place for all future scripts once they have been populated with data.

The User Session: Phase One

The Registration Panel

Luckily, the user's first session is much simpler. The first step is for the user to register an account, which he or she can do by filling out a short form. The user registration form is shown in Figure 52. The *Registration Panel* is a short form which allows users to register with *EDNA-E* manually. The user chooses their appropriate organization, division, and subdivision, then enters their name and email address. Then, they select a username and password and click on the "Register" button to complete the registration process. The registration form then checks the inputted data for errors (minimum and maximum string lengths as well as duplicate entries) and if no errors are found the information is added to *EDNA-E*'s user database. After registration, though, the account remains inactive until approved by the story administrator.

For this session, I assume that the user's name is Joe Student, and that he is a T&T student. During registration, he would choose "English Department" for the division and "UCF" for the organization. With these selections he would then be able to choose "T&T Student" from the subdivision select box.



The image shows a web-based registration panel for the EDNA-E system. At the top, there is a header banner with a blue background featuring a binary code pattern. On the left side of the banner is a computer monitor displaying a cartoon illustration of an elderly woman with white hair and a blue dress. To the right of the monitor, the text "Edna - E" is written in a large, stylized font, with "Event-Driven Narrative Analysis Engine" written in a smaller font below it. Below the banner, the main content area has a white background. It starts with the title "New User Registration" in a blue font. Below the title is a prompt: "Please complete the form below." The form consists of several fields: "Organization/Employer:" with a dropdown menu showing "Choose an option"; "Department/Division:" with a dropdown menu showing "Department of English"; "Subdepartment/Subdivision:" with a dropdown menu showing "Texts and Technology Ph.D."; "First Name:" with a yellow text input field; "Last Name:" with a yellow text input field; and "Email Address:" with a yellow text input field. Below these fields is a paragraph of instructions: "Choose a username and password, both with at least 4 characters and no more than 15. Your password will be encrypted before it is stored in the online database." This is followed by three input fields: "Desired Username:", "Desired Password:", and "Verify Password:". At the bottom of the form are two buttons: "Register" and "Clear Form".

New User Registration

Please complete the form below.

Organization/Employer: Choose an option

Department/Division: Department of English

Subdepartment/Subdivision: Texts and Technology Ph.D.

First Name:

Last Name:

Email Address:

Choose a username and password, both with at least 4 characters and no more than 15. Your password will be encrypted before it is stored in the online database.

Desired Username:

Desired Password:

Verify Password:

Register Clear Form

Figure 52: EDNA-E Registration Panel

Registration Results Panel

A screenshot of a web page titled "Registration Results" in blue. The page has a light green background. It contains several lines of text in a brown font. The first line is "Checking your information..". The next two lines are "New mail [Your New EDNA-E Storytelling Account] has been sent to [joe@nowhere.com].". The next two lines are "New mail [User Joe Student needs account approval] has been sent to [rudy@mail.ucf.edu].". The next line is "Group Registration Results". The next line is "This user, T&TJoe, is a member of UCF, UCF English Department, UCF English Department Texts and Technology, and Everyone.". The final line is "Your registration has been completed successfully. An email with login information has been sent.".

Registration Results

Checking your information..

New mail [Your New EDNA-E Storytelling Account] has been sent to [joe@nowhere.com].

New mail [User Joe Student needs account approval] has been sent to [rudy@mail.ucf.edu].

Group Registration Results

This user, T&TJoe, is a member of UCF, UCF English Department, UCF English Department Texts and Technology, and Everyone.

Your registration has been completed successfully. An email with login information has been sent.

Figure 53: Registration Results

After filling out the registration form, Joe Student is informed that the story administrator has been sent an email informing this person about his registration. The post-registration screen is captured in Figure 53. In addition, Joe will receive an email with usage instructions and general information. Finally, the group registration mechanism automatically adds Joe into the appropriate groups based on the selected organization, division, and subdivision from the registration form.

Here we should notice that Joe has mistakenly registered for the “Texts and Technology” subdivision rather than the “T&T Students” subdivision. This mistake will later need to be corrected in an administrative session using the *Group Administration* panel.

The Story Administrator Session: Phase Two

User Administration Panel

During phase two of the administrative session, I will approve Joe Student's account using the *Groups* icon in the administrative control panel. The *User Administration Panel* lists all users currently registered with *EDNA-E*. This includes users who registered manually as well as users imported from an XML file. Links next to each user allow a particular user to be edited or deleted. In addition, there is an "Access" link which allows a user's account to be enabled or disabled depending on the current status of that account. Two navigational links above the "Update Users" button provide access to additional screens of users, which are listed in segments in order to avoid vertical scrolling as much as possible.

Users are shown in a fashion similar to the administrative group listing shown earlier in Figure 41 and in the upcoming Figure 55, with only minor modifications. Much like I would do with the groups panel, I navigate to the appropriate screen using the "> > >" triple right-arrows until Joe Student's "T&TJoe" user account appears on my screen. Then, I click the "enable" link to activate his account. This process is shown in Figure 54, with the appropriate "enable" link for this task highlighted. A one-click enabling process is also available; a direct link to activate a user's account is sent to the story administrator by email with each new user registration.

User Administration						
Last Name	First Name	Username	Last Updated	Access	Edit	Delete
Notreal	Jane	STJane	02/15/2004	[disable]	[edit]	[delete]
Roney	Lisa	lcroney	02/11/2004	[enable]	[edit]	[delete]
Rushin	Pat	rushin	02/11/2004	[enable]	[edit]	[delete]
Saper	Craig	csaper	02/11/2004	[enable]	[edit]	[delete]
Scott	Blake	bscott	02/11/2004	[enable]	[edit]	[delete]
Sommer	Elisabeth	sommer	02/11/2004	[enable]	[edit]	[delete]
Stap	Don	stapd	02/11/2004	[enable]	[edit]	[delete]
Steward	Sherry	sherry	02/19/2004	[disable]	[edit]	[delete]
Student	Joe	T&TJoe	02/19/2004	[enable]	[edit]	[delete]
Vick	Erik	evick	02/15/2004	[enable]	[edit]	[delete]
<<< >>>						

Figure 54: Enabling a User Account

After I have approved Joe's account, Joe can then login and access any stories for reading or writing that he has been given the appropriate permissions for. The only remaining task for this administrative session then is to remove "T&TJoe" account from the "Texts and Technology" group and instead add him to the "T&T Students" group.

Group Administration Panel

The *Group Administration Panel* shown in Figure 55 is used for viewing and editing group names and group memberships. Within a selected group, group members are listed by user identification numbers separated by semicolons in the "Group Members" text field. New members can be added to a selected group in edit mode by clicking on the "Lookup User IDs" link and finding the associated user identification number for a user. This number can then be appended to the end of the "Group Members" text field or added to the beginning of this field

with a semicolon following the number. Once changes have been made, the “Edit Group” button will commit these changes to the database.

There is also an option here to add a new group, but that is not necessary since my group has been created automatically in my first administrative session. In this session, I need to first remove Joe Student from the “UCF English Department Texts and Technology” group and then add him to the “UCF English Department T&T Students” group. This is accomplished by removing Joe Student’s user id number, 63, from the “Member Ids” list illustrated in Figure 56.

Group Administration

Add group members by using the edit feature next to each group.

Name	Members	Last Update	Add	Delete
UCF	35	02/11/2004	Edit	Delete
UCF Digital Media	2	02/15/2004	Edit	Delete
UCF Digital Media Faculty	2	02/15/2004	Edit	Delete
UCF English Department	34	02/11/2004	Edit	Delete
UCF English Department Creative Writing	9	02/10/2004	Edit	Delete
UCF English Department Literature	11	02/11/2004	Edit	Delete
UCF English Department T&T Students	2	02/19/2004	Edit	Delete
UCF English Department Technical Writing	14	02/11/2004	Edit	Delete
UCF English Department Texts and Technology	5	02/19/2004	Edit	Delete

<<< >>>

Add New Group

Separate user members by semicolons.

Group Name:

Group Members:

Figure 55: Group Administration Panel

Edit Group 18

Edit group information below.

Group Name: UCF English Department Texts and Tech

Members Ids: 19;39;41;42;63

Lookup Ids: [\[Lookup Wizard\]](#)

Member Names:

- Rudy McDaniel (19)
- Paul Dombrowski (39)
- Dan Jones (41)
- Karla Kitalong (42)
- Joe Student (63)

Last Updated: Feb 19, 2004 at 05:43 PM

[Edit Group](#)

Figure 56: Edit Group 18 Panel

After removing Joe Student from “Group 18,” which is the “UCF English Department Texts and Technology” group, I can add him to the correct group by adding his user id number into the appropriate group’s member list. This group contains one other member, so I can add his group id after the other id separated by a semicolon and click the “Edit Group” button to update the database table. I now see Joe Student listed in the “Member Names” section of this panel, so I know he has been successfully added. This new group is shown in Figure 57. I can now logout and Joe Student has the access he needs to login. This login screen is shown in Figure 58.

Edit Group 51

Edit group information below.

Group Name:

Members Ids:

Lookup Ids: [\[Lookup Wizard\]](#)

Member Names:


- Joe Student (63)
- Sherry Steward (62)

Last Updated: Feb 19, 2004 at 11:00 PM

Figure 57: Edit Group 51 Panel

The User Session: Phase Two

Login Panel

 Login



Welcome to the Event-Driven Narrative Analysis Engine

Please login below

Accounts must be approved before your login is enabled.

Username:

Password:

 [Home](#)
 [Login](#)

Not currently logged in. This page last updated 02/17/2004 09:54:15 PM

Figure 58: Login Panel

After registering his or her account, a user will be sent an email informing that user that the story administrator has been notified of their account registration. The story administrator then confirms that the person registering is authorized to use the system. Once this has been confirmed, they use the *Users* icon described earlier to approve this new user's account. Then, the user will be able to login using the *Login* icon in the user control panel. If a user attempts to login before his or her account has been approved, the user will receive an error message. An example of this error message is shown in Figure 59.



Figure 59: Login Error

If a user's account has been approved, though, that user will be presented with the interface shown in Figure 32. After the interface has loaded, the user would then click on the *Stories* icon to access the story functions. There is the now familiar layout of a story list along with the option to add a new story provided by a link at the bottom of the panel, as revealed in Figure 60.

Add Story Panel



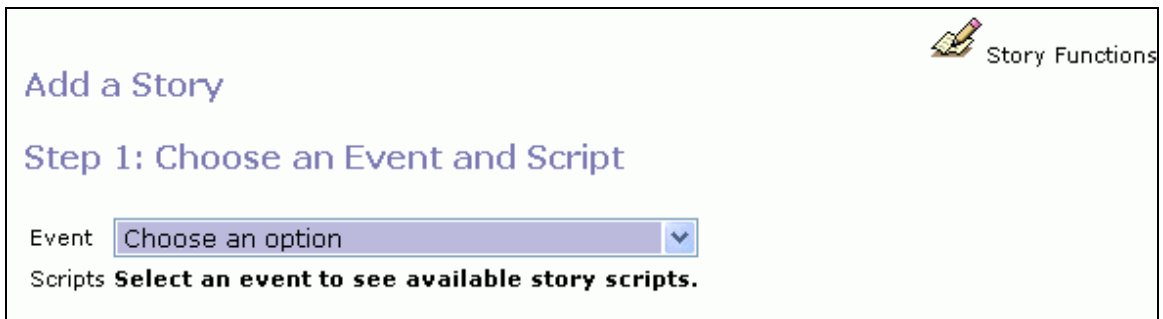
The screenshot shows a web interface titled "Read Stories". In the top right corner, there is a "Story Functions" link with a pencil icon. Below the title, there is a table with four columns: "Story Title", "Author Group", "Date Added", and "Date Updated". The table contains one row of data. Below the table, there is a link "[Add New Story]" in a blue box.

Story Title	Author Group	Date Added	Date Updated
Chris Fakeuser's Loss of Internet Story	STChris Sparkletech	02/18/2004	02/18/2004

[Add New Story]

Figure 60: Read Stories Panel

Here Joe Student, username “T&TJoe,” wants to jump right in and add a new story. To do this, he clicks on the “Add New Story” link that is shown below the current list of enabled stories in the table. This link then opens the *Add Story* panel captured in Figure 61.



The screenshot shows a web interface titled "Add a Story". In the top right corner, there is a "Story Functions" link with a pencil icon. Below the title, there is a section titled "Step 1: Choose an Event and Script". Under this section, there is a label "Event" followed by a drop-down menu with the text "Choose an option" and a downward arrow. Below the drop-down menu, there is a label "Scripts" followed by the text "Select an event to see available story scripts."

Event

Scripts **Select an event to see available story scripts.**

Figure 61: Add Story Panel

The *Add Story* panel contains a drop-down select list of events that have associated event scripts. Joe, curious, tries to access an event and script other than the “Writing a Research Paper” script created earlier in the first administrative session. Since he does not have access to

this other script, an error message is returned. In Figure 62, it is evident that Joe Student is not the script owner nor is he in a group which has write access for this particular script and story.

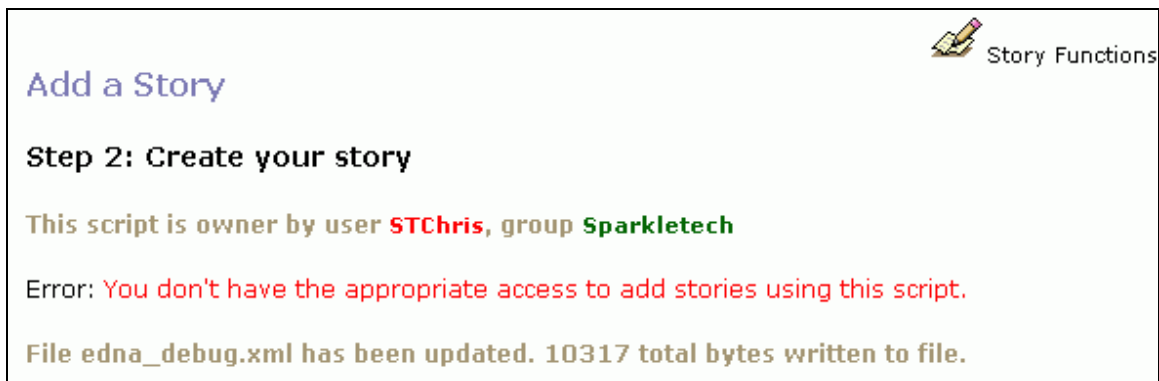


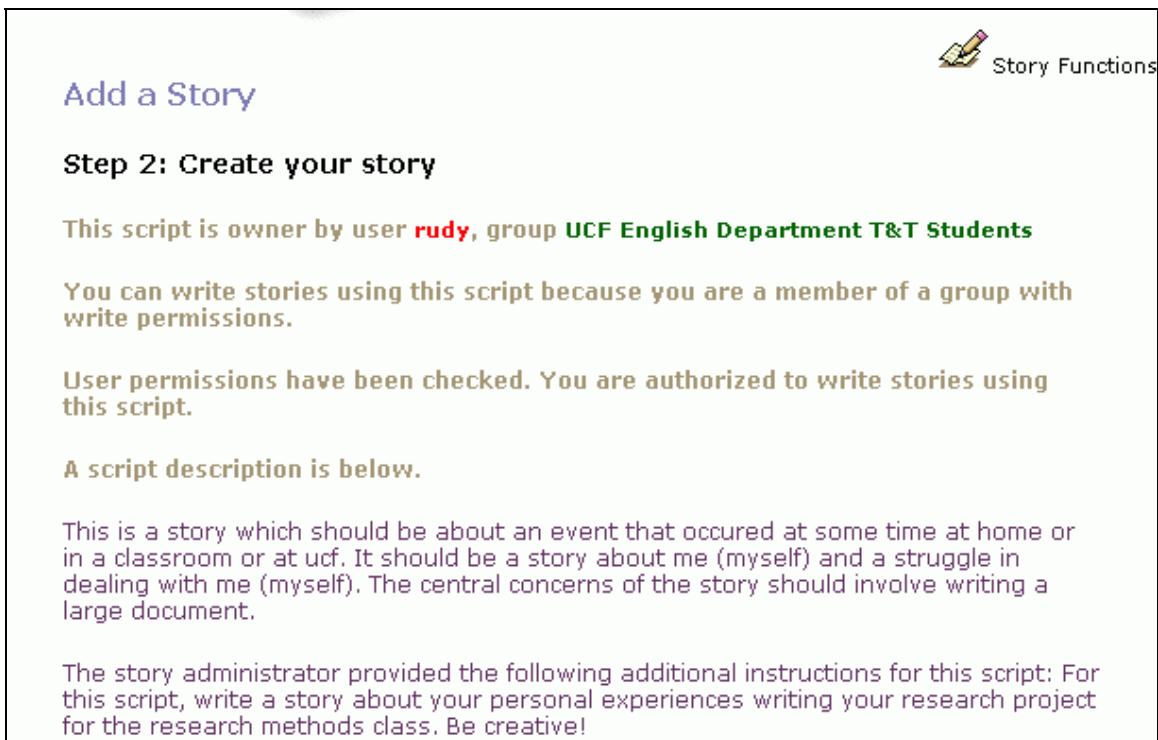
Figure 62: Invalid Permissions Error


As a member of the “T&T students” group, Joe does have access to the “Writing a Research Paper” script. An event associated with this script can then be selected as shown in Figure 63.



Figure 63: Add a Story Step 1

Clicking on the “Step 2” button then loads the larger form which allows Joe to actually write his story. This form is shown in Figure 64. The top panel of this page provides Joe with information about the script’s user and group owners, and explains to him that he was given access to write a story using this script because of his membership in the “T&T Students” group. The top panel then gives Joe a general description of the script which is automatically generated from the script as configured by the story administrator. This is followed by the administrator-authored text from the “User Instructions” portion of the script in Step Five of the script creation process.



 Story Functions

Add a Story

Step 2: Create your story

This script is owner by user **rudy**, group **UCF English Department T&T Students**

You can write stories using this script because you are a member of a group with write permissions.

User permissions have been checked. You are authorized to write stories using this script.

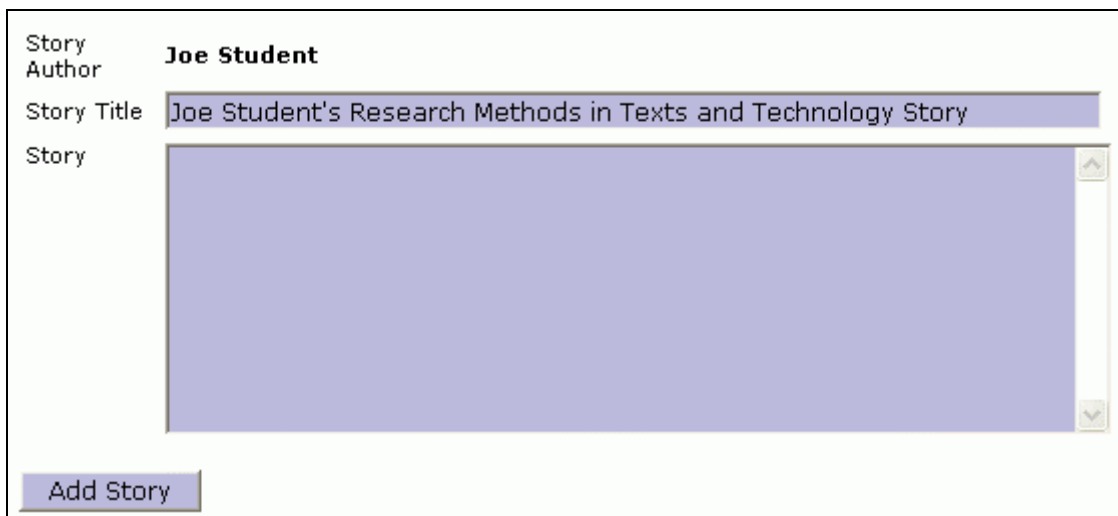
A script description is below.

This is a story which should be about an event that occurred at some time at home or in a classroom or at ucf. It should be a story about me (myself) and a struggle in dealing with me (myself). The central concerns of the story should involve writing a large document.

The story administrator provided the following additional instructions for this script: For this script, write a story about your personal experiences writing your research project for the research methods class. Be creative!

Figure 64: Create a Story (Top Panel)

The bottom panel of the Add Story panel provides a text field for Joe to enter a title for his story. A default title is also suggested here, which Joe can accept for his story's title if he is not feeling especially creative. A blank text box then provides space to capture Joe's actual story. This bottom panel is shown in Figure 65.



The screenshot displays a web form titled "Create a Story" with the following elements:

- Story Author:** A text field containing the name "Joe Student".
- Story Title:** A text field containing the default title "Joe Student's Research Methods in Texts and Technology Story".
- Story:** A large, empty text area for entering the story content.
- Add Story:** A button located at the bottom left of the panel.

Figure 65: Create a Story (Bottom Panel)

Analyzing Story Panel

Analyzing Story

Attempting to see if your story meets the script requirements.

Using script id 20

Please be patient while your story is analyzed.

Original Story

Joe Student's Research Methods in Texts and Technology Story (335 total words)

My name is Joe and I was a student in Professor Dombrowski's Spring 2000 research methods class. This was the first doctoral level course I had ever enrolled in, so I was very nervous about it. In the beginning of the semester, Dr. Dombrowski told us there would be a large research paper required, and that this paper would be due by the end of the semester. Boy was I worried! I sat in that classroom just wondering how I could ever finish a project that big. I have always enjoyed writing but I have never written anything longer than about 20 pages or so, and he wanted 40! I decided I needed to formulate a plan to actually get my project done in time.

What I finally decided to do was this. In my notebook, I began commenting and writing notes about possible topics in the margins next to my class notes. When we discussed a topic I thought was interesting, I went back later that night and highlighted the comments next to the topic we were discussing at the time. Since we also reviewed technical communication research each week, I was able to build up a good list of references, too.

It turned out that I had no trouble reaching my 40 page requirement. By taking lots of notes and commenting on topics I found interesting, I already had a good collection of ideas and research when it came time to start writing my paper. So if you are taking this class and are worried about the research project don't panic! If I got through it then anyone can. I ended up writing my paper on graphics and visuals in technical communication. If you are interested in this topic feel free to contact me and I would be happy to share my resources or give you a copy of my paper. Oh yeah, and I got an A!

Figure 66: Analyzing Story Top Panel

After Joe submits his story, it is then displayed back to him in the *Analyzing Story* panel. The top part of this panel, shown in Figure 66, simply prints out his story using the CSS style sheets configured by the story administrator and prints a word count to the screen. The middle panel, shown in Figure 67, prints a list of extracted keywords to the screen. These keywords are

generated by taking Joe's story, removing common words such as pronouns, conjunctions, common verbs, and articles, and then cleaning up the text by removing white space and punctuation. The resulting keywords are then used in the search function which is found under the *Stories* icon in the user control panel. Common words are also editable by the story administrator using XML data files which serve as mini-dictionaries for various functions in *EDNA-E*.

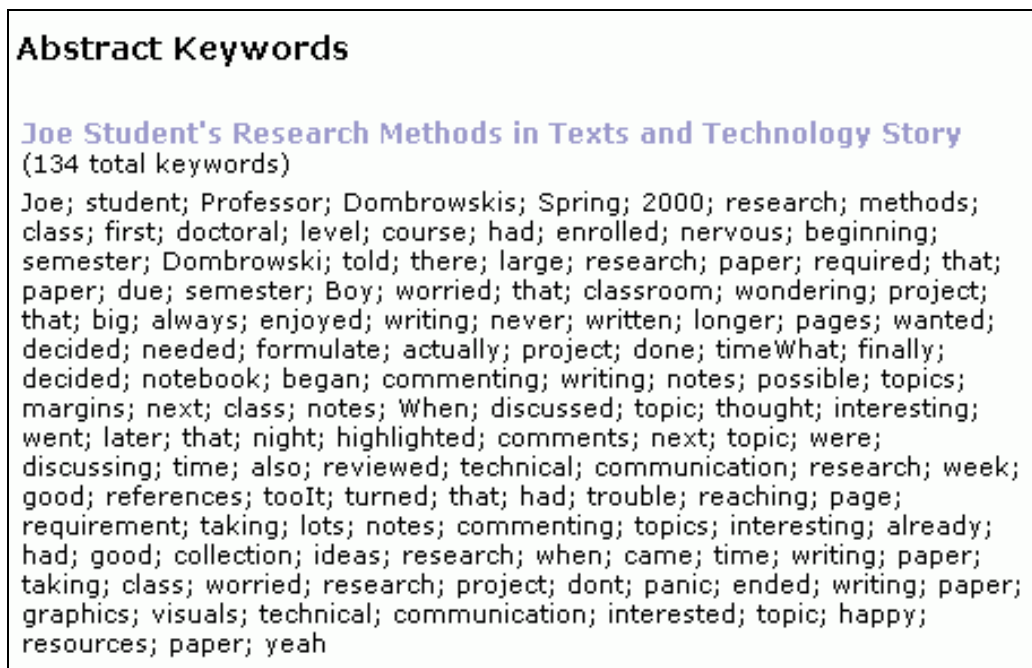


Figure 67: Analyze Story Middle Panel (Keywords)

The bottom and most important part of the *Analyze Story Panel* is the automatic analysis engine shown in Figure 68. The purpose of the automatic analysis engine is to verify that a particular story meets the requirements of the script controlling that story. This feature scans

through the submitted story looking for the various story elements specified by the script. These elements are scored based on how often they appear in the text, and they work primarily by keyword matching. The story administrator can adjust independent thresholds for each story element. These thresholds must then be surpassed by the relevant scores or the analysis will fail overall verification based on that particular element not meeting the threshold score.

After scoring has been completed, the results are listed in Figure 68 under the “Checking Thresholds” heading. If all thresholds have been passed, then the system indicates that the story meets the requirements of the script and the script is listed under the administrative *Stories* icon with the label “verified.” If a story fails verification, it is still added to the database and the XML story file is still created. In this case, though, the story is listed with an “unverified” label. This flag indicates to the story administrator that they will need to read the story closely to determine if it actually meets the requirements of the script.

This keyword matching technique would not work well for more complicated story elements, however, including the concerns or thematic elements configured by the script. For this type of analysis, I again used customizable XML-based dictionaries to search for certain keywords and phrases that relate to a given topic. For instance, in this case, the thematic dictionary I linked was “writing.xml.” The “writing.xml” dictionary file, as it appears in *EDNA-E*’s integrated XML editor, appears in Figure 69.

Scanning Story

Allowable places

- At home [not found]
- In a classroom [ok] score=1
- At UCF [not found]

Allowable time settings

- Any elements allowed [ok]

Allowable protagonists

- Me (Myself) [ok] score=5

Allowable antagonists

- Me (Myself) [ok] score=5

Allowable themes

- Writing a large document [ok] score=3

Checking Thresholds

- Protagonists OK
- Antagonists OK
- Places OK
- Times OK
- Themes OK

Story OK

Figure 68: Analyze Story Bottom Panel

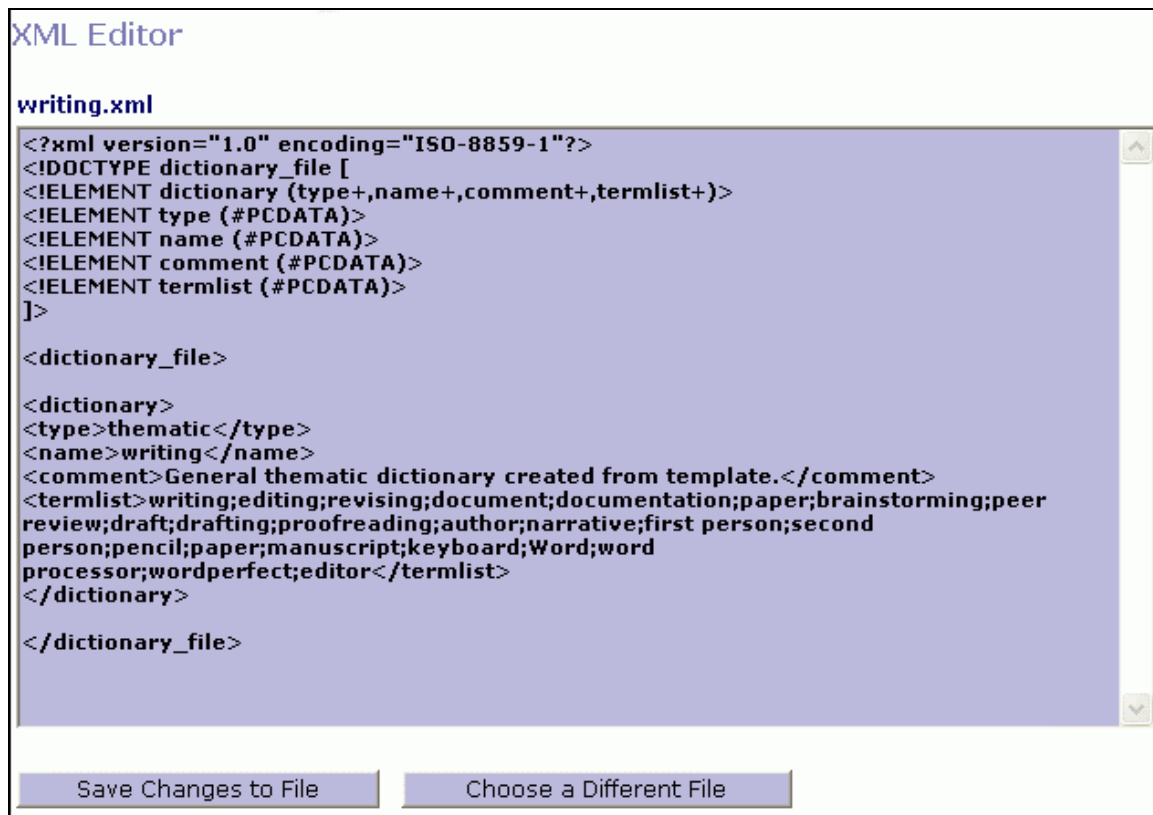


Figure 69: "Writing.xml" File in XML Editor

At this point, Joe's story has now been verified and stored in the database. In reality, a third administrative session would now be required as I would need to go back in to the administrative control panel and enable this particular story in order for it to be displayed in the story listing underneath the Joe's *Stories* control panel icon. This process is virtually identical to the user approval process shown in Figure 54, though, so an explanation of this process is not necessary.

I have now shown how a typical user and administrator would use this software to first set up a storytelling environment and then to actually tell a story. More advanced features such as story linking, story searching, and importing were omitted from this discussion to keep this

chapter manageable, but these functions are available in the software and are discussed in EDNA-E's online documentation. With that in mind, I will proceed to discuss a few of the key advantages and disadvantages of this software before writing a few final remarks that will lead us into this dissertation's conclusion.

Program Features

EDNA-E is designed to be extremely flexible and useful for an implementation in a variety of different environments. The flexibility of the scripting engine makes it possible to have literally millions of different story template combinations, depending on the specific XML files configured by the story administrator. In addition, the software offers the following integrated features:

- The software is documented and commented using the PHPDoc documentation system, which will be very helpful in porting this application to other languages such as *Java* or a *.NET* implementation.
- The software has a built-in XML editor for editing common script files, configuration files, and other integrated XML files stored in the XML directory.
- The system features a comprehensive error logging system, which logs the error date and time, the user running the application, the script file, and the specific error message being generated to an XML file. This file can then be examined by the story administrator for troubleshooting purposes or to find out how to refine the system in order to better meet the needs of users.

- Dynamic forms are used as much as possible in all user-level functions. This means that a user will not see irrelevant choices when filling out forms or using the software. For example, in the *Registration* panel, as the user selects the organization a JavaScript function will refresh the page and only allow appropriate divisions which exist underneath that organization to be selected. Then, the same process occurs for the subdivision process. Dynamic forms are also used in the story adding functions; only events which have been assigned scripts are listed as available events during the story input process.
- *EDNA-E* has a comprehensive administrator mode which makes all story administration functions as easy as accessing a web-based control panel.
- Cascading Style Sheets (CSS) are used extensively so any aesthetic changes can be made simply and efficiently. Changes a simple variable value in one file will modify all elements on all fifty plus script files that generate web pages using CSS styles and attributes.
- XML dictionaries allow the story administrator to configure specific keyword and phrase files that can be linked to and configured for any central concern or theme for a story's script.

Design Limitations

Despite its many advantages due to its flexible configuration, there are also plenty of ways in which this software could be extended or improved. *EDNA-E* has certain limitations due

to the way the software was designed. Many of these limitations appear in the advanced administrative functions of the software. These limitations include the following issues:

- Automatically created group names can only match the names of the organization, division, or sub-division paired with that group during its creation.
- All users must register for at least one organization, division, and sub-division in order for their registration to be processed.
- Deletions do not prompt the user to confirm before removing information from the database tables.
- When deleting an organization, division, or subdivision within the organizational administration panel, the corresponding group is not deleted and needs to be deleted manually in the group administration panel.
- Certain XML import functions will only work if the XML tags are in the correct order. For instance, on the user import feature the password element must be listed last in each user parent element in order for the data to be processed correctly.
- Some XML import functions will only work if the appropriate divisions, subdivisions, and organizations have already been created in the organization administration panel.
- The story analysis phase will not work very well if there are many misspellings present in the original story. An integrated spell-checker would help to solve this problem. Multi-language spell-checking dictionaries could also be implemented to extend functionality into international environments.
- The scripts functions do not allow for certain metaphorical constructs such as a metaphorical protagonist or antagonist. These types of constructs can certainly be added

to the default protagonist or default antagonist data files, but they will function as though they were actual protagonists and antagonists in the story and not metaphorical agents.

Discussion

By modeling and describing a flexible and robust knowledge management software package, I have shown that various theories from computer science, technical communication, psychology, and business management can be combined to create a useful device or tool for studying and communicating certain types of knowledge. Specifically, this system works best in communicating certain types of tacit technical knowledge and socio-cultural types of knowledge which need to be framed in a story in order to make sense. To emphasize this convergence of these various theoretical constructs, I will once again describe each of the theories and how they were integrated into this software application.

The overall nature of this software is based on the principle of weak artificial intelligence. As I explain in the opening of this chapter, this means that the software functions as a useful tool that allows us to study the way we think about and manage knowledge. Intelligence in this system is evident in the system through the same type of distributed cognition Graham Smart identified in the Bank of Canada *monetary-policy story* discussed in Chapter Four. In this model, an event produces a one-to-many mapping of knowledge representations. This means that one event with one script can generate many different types of stories which integrate different users' experiences with that event and encapsulate their own personal knowledge about that event. In this manner, an overall body of knowledge is produced that users with appropriate

access can then peruse in order to make their own subjective connections with story themes and events.

Such an approach also takes into account the know-how model of knowledge as explained by Hubert Dreyfus. The know-how model explains why some of the commonsense knowledge-problems have accompanied so many of our AI applications. An example of this type of problem is found in the story comprehension AI applications I explain in Chapter Two. In these types of applications, computers were programmed to understand simple children's stories, only for the programmers to find that an enormous body of common sense knowledge was necessary for even this simple type of narrative understanding to occur. Dreyfus explains that these types of problems were not predicted because we were thinking of the problem non-holistically. He writes,

... the commonsense-knowledge program was not really a problem about how to represent *knowledge*; rather, the everyday commonsense background understanding that allows us to experience what is currently relevant as we deal with things and people is a kind of *know-how*. The problem precisely was that this know-how, along with all interests, feelings, motivations, and bodily capacities that go to make a human being, would have to be conveyed to the computer as knowledge – as a huge and complex belief system (xi-xii)

Dreyfus continues to explain how he believed such a task to be impossible, but he was unable to provide a justification for his beliefs until he considered the process of skill acquisition (xii). Skill acquisition “usually begins with a student learning and applying rules for manipulating context-free elements” (Dreyfus xii). As we become more and more familiar with a task

requiring that skill, though, we begin to rely less on these context-free rules and more on our own collected experiences and expert knowledge about that task. At this point we no longer even need these general rules; as Dreyfus notes, “There is no reason to think that the rules that play a role in the *acquisition* of a skill play a later role in its *application*” (xiii).

The linear progression of human knowledge according to Dreyfus can then be represented as follows: context-free rule manipulation, then context-dependent rule manipulation, then finally no conscious rule manipulation at all. At this point a computer, which needs rules during all phases of operation, would have no computational process to assist in the knowledge-building process. Dreyfus believes, as I do, that strong AI will therefore be forever plagued by such a dilemma, and that strong AI based on symbol systems and rules is therefore unlikely to succeed. *EDNA-E*, though, is based around a system that is strongly conducive to the know-how knowledge model. By allowing users to create and exchange stories, individual experiences are framed in these complex belief systems or ideologies that influence a person’s reactions to an event and the skills they use to react to that event. Such an organization relies on emergent knowledge, or knowledge based on the emergent properties of narrative systems as discussed in Chapter Three. Another field of study *EDNA-E* is similar to here is that of neural network modeling, in which “the neural network modeler provides not rules relating features of the domain but a history of training input-output pairs, and the network organizes itself by adjusting its many parameters so as to map inputs to outputs, that is, situations into responses” (Dreyfus xv). The event-driven construction of *EDNA-E* provides just such an environment for mapping the events representing administrator-added situations into the users’ stories

representing specific responses to those events based on individual ideologies and their social belief systems.

When another user reads those stories, then, they are interpreting these stories using their own belief systems and modifying the skills and other observable elements of explicit knowledge in order to take into account those modified beliefs. In both cases, the final-order series of knowledge creation in which no rules are used is done by a human user. Granted, this means the application is now a weak AI application, but it also means that it is based on a feasible theory.

Additional theories of information processing are found in the actual design principles I followed when creating my software. The general design of this software takes advantage of object-oriented programming techniques. Key features of this OO methodology, including encapsulation and inheritance, provide extensibility and portability to program features for future applications or conversions into other programming languages. This dissertation thus expands upon my earlier research concerned with extending technical communication tools using object-oriented design principles (McDaniel, 2001). In this chapter, a fully functional application which is based almost entirely around objects was described in detail. The idea of objects as individual computing devices was also touched upon, and lends credibility to the idea that *EDNA-E* is a weak AI application by supporting the distributed cognition model explained by Lewis Thomas in Chapter Two and Graham Smart in Chapter Four.

The *Events* icon, which contains event-management functions, was fashioned from Mieke Bal's ideas of the narrative fabula, in which a story is constructed from a series of finite events. Event features serve as the backbone for the story collection system, an idea which I applied

after considering the application of narrative fabula for constructing finite state automata to represent stories.

The *Scripts* icon, which houses script creation and editing processes, was devised using Roger Schank's ideas about conversation as stories. In this paradigm, Schank theorizes that the process of conversation is simply responding to stories we hear with our own stories using the appropriate scripts for fashioning these stories. The scripts in *EDNA-E* are therefore used as models for building appropriate stories. The scripts model also makes heavy use of Marvin Minsky's frame theory. Using his story-frames model, each script contains a general structure for representing a story with blank terminals that can be filled in with appropriate protagonists, antagonists, places, times, and themes.

Scripts also serve to overcome some of the additional problems identified with narrative systems in Chapter Four. Specifically, the user-privacy problem and story solicitations problem identified by Stephen Denning in Chapter Four are overcome by adding a UNIX-like schema for defining story permissions in Step Two of the script creation process. Since human users generate the actual stories rather than a computer, the problems with creativity and novelty are now the responsibility of the user and not the computer. This helps the software program to avoid the problems of repetition and blandness exhibited by the *TALE-SPIN* program discussed in Chapter Four. To ensure only appropriately creative stories are accepted for events, the story administrator is installed as a human expert and knowledge manager in this process. The only stories which will be displayed by the system, then, are the stories which the story administrator feels are appropriate.

In the conclusion to this dissertation, I will wrap-up a few loose theoretical ends and speculate on some possible applications for this software. In addition, I will theorize on the future of narrative knowledge management techniques by briefly considering some agent-based implementations and other techniques based on emergent technologies.

CHAPTER SIX: CONCLUSION

“...intelligence requires understanding, and understanding requires giving a computer the background of common sense that adult human beings have by the virtue of having bodies, interacting skillfully with the material world, and being trained into a culture”
– Hubert L. Dreyfus

With my own software model for managing and distributing stories now defined and in place, it is interesting to think about how this type of system might be refined using additional theories and observations from the fields of technical communication and computer science. For instance, how might this model change if it were programmed in a list-oriented fashion using *Lisp* or *Prolog* rather than in an object-oriented fashion as it now exists? Would it be possible to adjust this program to perform data mining functions in order to create stories from pre-existing knowledge bases? What complications might arise from trying to integrate different languages into the user interface or different character sets into the XML files? These are all interesting questions, and I intend to address them in future research and future versions of this software.

The major remaining theoretical issue in this dissertation that needs closure is the idea that binary representation somehow limits the progress of strong artificial intelligence research. I hope it was clear that binary representation of data is only one example of the many problems that AI researchers and developers face in their quest to mimic human intelligence. This problem is indicative of much larger problematic issues: the embodiment thesis (whether the mind and body can exist as separate entities), the issues involved with modeling brain functions

computationally, and even the inherent structures and ideas that are language, communication, and human intelligence. My approach throughout this dissertation was to touch upon these issues, provide examples of complications where possible, and show how weak AI applications could be created to make up for the lack of progress in generalized (strong) AI research.

There are some theorists, though, that believe an even stronger correlation between storytelling and intelligence may exist. For example, AI researchers such as Roger Schank continue to claim that storytelling may be the key to making computers understand the data they are manipulating. By extension, this means that stories are the key to creating strong artificial intelligence. John Searle explains, "... one can describe Schank's program as follows: the aim of the program is to simulate the human ability to understand stories. It is a characteristic of human beings' story-understanding capacity that they can answer questions about the story even though the information that they give was never explicitly stated in the story" (Searle "Minds, Brains, and Programs" online). In other words, computers that can analyze stories in this way can make inferences that lead to the common-sense types of knowledge found in human thinking and reasoning. An example of this might be a computer that reads Joe Student's story about his experience in a research methods class that is given in Chapter Five. The computer, when asked whether or not Joe enjoyed the class, would respond with an affirmative answer, because the nature of the story suggested Joe's positive tone would reflect an overall positive experience. In order to do this, Schank suggests that computers can be programmed to store representations about certain types of story situations and react to the stories fitting these profiles in specific ways.

The debate again hinges on our social and cultural constructions of knowledge. Joe Student's story can certainly be interpreted in different ways according to different ideological values; it may be improper for him to speak negatively about his academic experiences in some cultures, for instance, so the overall impression of his experiences may be more ambiguous to readers from these cultures.

Personification of technology is also problematic when attempting to address the problem of strong AI. Searle explains,

We often attribute 'understanding' and other cognitive predicates by metaphor and analogy to cars, adding machines, and other artifacts, but nothing is proved by such attributions. We say, 'The door *knows* when to open because of its photoelectric cell,' 'The adding machine *knows how* (*understands how*, *is able*) to do addition and subtraction but not division,' and 'The thermostat *perceives* changes in the temperature.' The reason we make these attributions is quite interesting, and it has to do with the fact that in artifacts we extend our own intentionality; our tools are extensions of our purposes, and so we find it natural to make metaphorical attributions of intentionality to them ("Minds, Brains, and Programs" online)

Part of the reason we use these personifying terms is that many people do not understand how to actually state how these machines are working in more precise language; and if they do, they do not know how to phrase these operations in a language that other people will actually understand. So terms describing our technologies as *knowing*, *understanding*, and *perceiving* have become so ingrained in our technological literature and our culture that it is not surprising

so many people are willing to believe strong artificially intelligent technology is possible. Many people, including well-educated computer scientists, philosophers, and academics, not only believe that such a technology is possible but also that it is imminent.

The binary issue is therefore a subcomponent problem of a much larger issue: our understanding of human intelligence and our general ideas about how to package and distribute human intelligence. Until we fully understand how humans think, process, and reason about problems, how we manipulate long and short term memories, and how we deal with the layers of subconscious and conscious psychologies, the attempts to achieve such constructs in machinery are almost certain to fail. Some researchers such as Hubert Dreyfus even argue that the symbol-system hypothesis itself is so flawed that strong artificial intelligence will never be possible using this model as a basis for development (3). The good news, however, is that these fields of psychology, computer science, and technical communication studies can work together towards the goal of understanding human intelligence and knowledge manipulation. By taking an interdisciplinary approach and using combined research from these fields with overlapping goals, studying about human intelligence will lead to improvements in artificial intelligence and similar studies of artificial intelligence will help provide insight into our own ways of thinking.

The problems with defining intelligence and the process of intelligent reasoning have long been noted by prominent AI researchers. In many cases, definitions are divided amongst various paradigms which span multiple interdisciplinary fields. For example, Randall Davis, Howard Shrobe, and Peter Szolovits identify five fields which have produced distinguishable definitions of intelligent reasoning: mathematical logic, psychology, biology, statistics, and economics (25). In mathematical logic, “intelligent reasoning is some variety of formal

calculation, typically *deduction ...*” (24). In biology, it is a “characteristic stimulus/response behavior that emerges from *parallel interconnection of a large collection of very simple processors*” (24). In psychology, intelligent reasoning is seen “as a characteristic of human behavior” (24). Finally, the statistical definition “adds to logic the notion of uncertainty, yielding a view in which reasoning intelligently means *obeying the axioms of probability theory*,” and the economic version “... adds the further ingredient of values and preferences, leading to a view of intelligent reasoning defined by adherence to the tenets of *utility theory*” (24).

What I have shown in this dissertation is that these various independent definitions can actually be combined into one all-encompassing definition of intelligence and reasoning which uses fundamental ideas from each field. In addition, the existing field of technical communication and the newly defined field of texts and technology contribute to this definition in a humanistic fashion, suggesting as others have done that technical communication researchers and practitioners are ideal choices for knowledge managers or story administrators in this model. Table 2 breaks down these various properties by field and explains how each of these properties is met through the software features described in Chapter Five.

Table 2: Intelligent Reasoning Models

Field	Property	Present in Software?
Texts and Technology / Technical Communication	Humanism	Yes; The knowledge manager here is the human story administrator.
Mathematical Logic	Logic	Yes; Numeric and textual processing, logical selection of elements from database and XML files based on queries.
Psychology	Frames	Yes; Representation of scripts as story-frames. Terminals allow flexible creation of story elements.
Biology	Connectionism	Yes; Object-oriented design suggests independent processing units with messaging capabilities.
Statistics	Uncertainty	Yes; Human creativity in story-inputting process ensures that no two stories will be exactly the same. Even rigidly defined scripts will gather different stories.
Economics	Utility	Yes; utility theory is concerned with making subjective value interpretations from choices. A storytelling model certainly facilitates this theory, as subjectivity can easily be inferred from the choices made during fabula transitions in a narrative.

The final issues to consider in regards to my software and artificial intelligence are as follows: since the software model for a narrative knowledge management system created in

Chapter Five is indeed binary and is processed by a digital computer, how is this system in fact intelligent? What types of knowledge management is this software capable of performing? Are there certain types of knowledge at which this program is better at manipulating? After addressing these three questions, I will describe two examples of how this model might be extended in the future to achieve an even more immersive storytelling environment and perhaps improve on my extended model of distributed cognition using agent-based and virtual reality technologies.

In addition to the theoretical properties described in Table 2, I believe that *EDNA-E* exhibits the same types of intelligence found in other weak AI applications because this software presents a model for storing and searching data that is similar to the models used in expert systems. Some consider expert systems to be types of artificial intelligence applications, while others argue that they represent a new category of computing devices. This may be true, but it is clear that expert systems originated from the research stemming from advancements, however slight, in strong AI. Janet Vaux writes, "... the goal of expert-systems design was to produce a system that would be quasi-human" (235). Whether or not expert systems are now in fact artificial intelligence applications is another debate entirely, but it is clear that these systems at least have overlapping goals to some extent. With this in mind, it is important to note that the same types of expert knowledge that can be stored in lists and databases can often be better served by framing that knowledge inside a story. Readers of that story will then not only find practical advice or operational definitions, but also keywords and situational elements that will remind them of their own stories and/or similar experiences with that subject matter. The customizability and flexibility of this model is also useful for extending the domain in which

EDNA-E can work intelligently; the creation of a few new XML data files can move this software's expertise from the field of medicine to the field of high-school counseling, for example. As Searle explained in my introduction to Chapter Five, weak artificial intelligence applications are useful for helping us to study how we ourselves think and process information. In this respect, *EDNA-E* is also intelligent. By allowing users to respond differently to the same event, this software allows the story administrator and other users of the system to see how various people with different life experiences write stories using the same script. Creativity, which is of course another major component of human intelligence, can also be studied in this fashion.

The second question I asked above was in regards to the specific types of knowledge management this software is capable of performing. Because of the nature of *EDNA-E*'s design, this software can excel in managing both explicit and tacit types of knowledge. Explicit knowledge can easily be embedded in a narrative form, as a user can simply include a paragraph into their story which lists a series of steps or actions take to resolve a problem. For instance, if a user were writing a story about their experiences removing a computer virus this might be done in the following manner: "After deciding on an appropriate software program to remove the virus, I proceeded to install the software by:" This story could then continue by listing the sequential steps in a descriptive fashion or even by separating steps onto separate lines as though a bulleted list were being used.

Tacit knowledge can also be manipulated using this narrative software system. Here, in fact, the story-based system edges out other competing knowledge-management systems precisely because of the framing elements allowed by a narrative format. In an expert system,

for example, a user is not given the option of reading a story which frames an author's moods and motivations for solving a particular problem. In an expert system, these types of influences are considered irrelevant to the most probable solutions for a given problem. This setup works well for situations in which the problem is defining the most likely disease given a set of symptoms or trying to find the best stock to purchase given a ten-year pattern of market fluctuations. For tacit knowledge, though, both mood and motivation can be critical when trying to figure out how an author solved a problem when that author cannot explicitly explain that process using a series of steps.

The final question to consider about this software is whether or not certain types of knowledge are better represented and disseminated in this type of narrative format. Many different researchers have attempted to classify and categorize knowledge into different categories: tacit knowledge versus explicit knowledge, universal knowledge versus socially constructed knowledge, body-situated knowledge versus mind-situated knowledge, etc. Of course some of these categories have blurred boundaries, too; many researchers claim that all knowledge is socially constructed, for example.

Harry Collins defines four kinds of knowledge: symbol-type knowledge, embodied knowledge, embrained knowledge, and encultured knowledge (69). Symbol-type knowledge is "knowledge that can be transferred without loss..." (69). Many early AI researchers believed all knowledge could be encapsulated and represented in symbolic form, which is what led to the enormous hopes for, and financial backing of, the strong AI industry leading into the early 1980s. Vaux explains, "In the 1970s and 1980s, the computability of human intelligence was usually explained through the theory of thought as symbol manipulation" (232). This thesis was

also responsible for much of the neural network research in early AI efforts, in which scientists attempted to determine the biological symbols used by our bodies when transmitting nerve impulses across synaptic gaps.

Body-situated or embodied knowledge is based on a theory in which “... the way we cut up the world around us is a function of the shape of all our bodies” (Collins 68). Recalling Thomas Davenport’s definition of data as our “world observations” from Chapter One and Chapter Two of this dissertation, this type of definition affects the entire structure of knowledge management from data, to information, to knowledge itself. Collins gives a particularly memorable example of this type of knowledge, and it is even in narrative format. He writes about a brainwashed Vietnam veteran who as a result of his brainwashing becomes particularly adept at learning and acquiring new knowledge. This veteran then obtains the knowledge of a champion tennis player through a colander-shaped metal bowl device which connects the head of the tennis player to the head of the veteran. The veteran acquires all of the championship tennis knowledge and steps on the court to play a game. Collins continues,

He goes to serve in his first match—Wham!—his arm falls off. He just doesn’t have the bone structure or muscular development to serve that hard. And then, of course, there is the little matter of the structure of the nerves between brain and arm, and the question of whether the brain of the champion tennis player contains tennis playing knowledge which is appropriate for someone the size and weight of the recipient. (67)

In other words, the knowledge containing instructions for how to perform a certain task is explicitly tied to the body in which this task has previously been performed. This seems like an

obvious enough connection, but it is one that many traditional AI researchers have overlooked or ignored.

Embrained knowledge is very similar to embodied knowledge: it is knowledge that depends on the physical configuration of the brain. Collins explains, “There is the matter of how the neurons are interconnected, but it may also have something to do with the brain as a piece of chemistry or a collection of solid shapes. Templates, or sieves, can sort physical objects of different shapes or sizes; perhaps the brain works like this, or like the working medium of analogue computers” (68). We are now again at the critical issue of complexity: can real world and analogue knowledge be represented in a digital fashion? The answer my research has suggested is “No,” but a narrative process can in fact ease the transfer of knowledge from one human agent to another using a computerized system. Consider the diagram shown in Figure 70. Here there are four circles which represent actors or agents in the knowledge exchange systems; in this case there are two actors: the champion tennis player and the Vietnam veteran. In this model, the colander-type knowledge transfer system is suggested at the top of this figure. The knowledge for championship tennis play is transferred directly from the tennis champion to the Vietnam veteran, which makes no accommodation for embodied or embrained characteristics. In Collins’ story, the veteran’s arm falls off after attempting to serve a tennis ball using this model. In the bottom portion of Figure 70, the diagram is revised to take into account a narrative model of information transfer. Here, the tennis champ tells a story about tennis, or perhaps tells many stories about tennis, which encapsulate(s) some knowledge about tennis playing or tennis playing strategy. The Vietnam veteran then reads these stories and translates this knowledge into something his brain and body can process and make functional use of. The two dashed squares

in this figure represent independent life experiences and anatomies, while the gray-filled square with rounded edges represents an overlap in life experiences where connections can be made with the person telling/writing the story and the person hearing/reading the story. This shaded area represents the domain of intersubjectivity for this narrative knowledge management system.

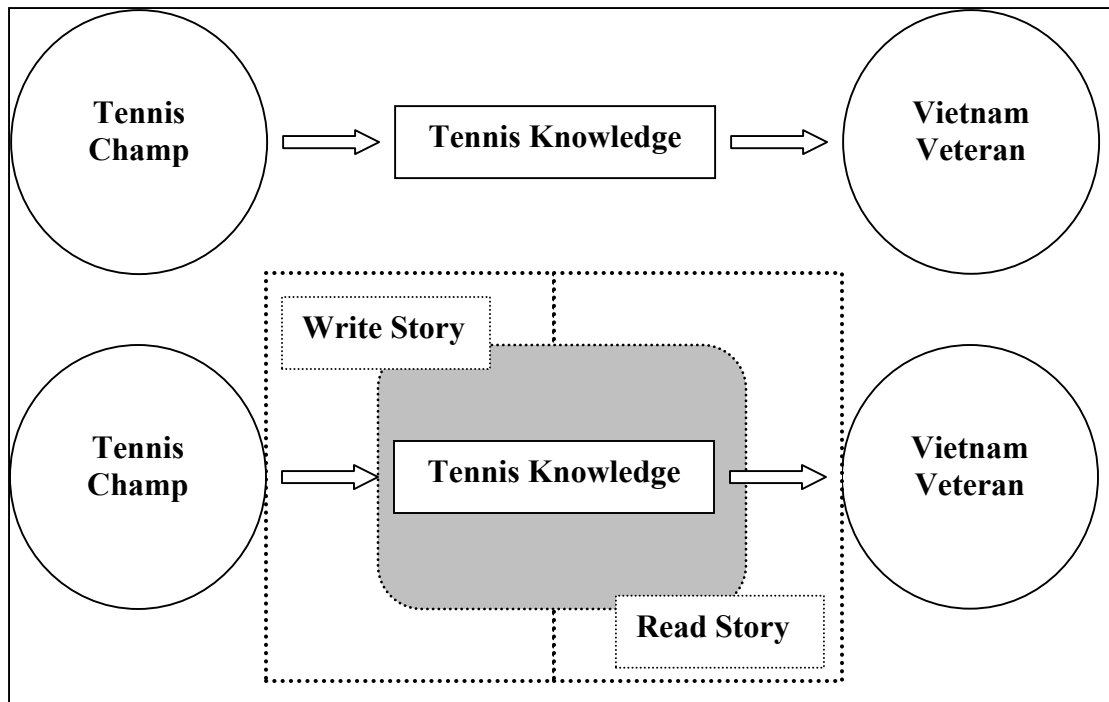


Figure 70: Embodied/Embrained Knowledge Representations: Two Models

I have shown in above paragraphs that *EDNA-E* does well at handling both explicit and tacit types of knowledge, but I have not addressed my software's ability to handle body-situated (embodied) or brain-situated (embrained) knowledge. In both cases, the translation of experience into narrative format eases the transfer of information from sender to recipient by embedding these body or brain-specific ideas into a narrative medium. For example, using the

software features described in Chapter Five, a process could be developed to help distribute tennis playing knowledge from the champion to the Vietnam veteran. The process can be defined as follows:

1. A new organization is created for the tennis player, depending on his ranking or perhaps his nationality. For instance, the organization, division, and subdivision could be as follows: “Athletes, Tennis Players, Ranked (United States).”
2. A similar organizational hierarchy is created for Vietnam veterans. This hierarchy might reflect “Veterans” for the organization, “United States” for the division, and “Vietnam War” for the subdivision.
3. A group is created for those wishing to learn about tennis playing. This group could be named “Tennis Players.” After the two users are given accounts, the story administrator then adds these two users into the “Tennis Players” group.
4. An event is created in order to solicit stories. This could be the “Learn How to Play Tennis” event.
5. A new thematic dictionary named “tennis.xml” is created. This dictionary contains common keywords and phrases related to playing and learning about tennis.
6. A script is created by the story administrator with the following properties:
 - Script Name: How to Play Tennis Script
 - Permissions: Owner: Story Administrator (R+W); Group: Tennis Players (R+W)
 - Allowed Places: All
 - Allowed Times: All

- Default Protagonist: Me (Myself)
- Default Antagonist: Me (Myself) or Tennis Opponent
- Default Theme or Concern: Playing Tennis

With these settings and this script in place, the Vietnam veteran is then able to learn about tennis through reading various stories added by the tennis expert. By connecting themes and ideas from the story with his own experiences, the veteran is then able to internalize the knowledge he needs with no fear of loss of limb. Granted, this process does not lead to the immediate creation of another tennis expert as the colander-model hypothesized by Collins would suggest, but then again this model for knowledge management actually exists; the colander-model is still science fiction.

Collins' final category of knowledge is encultured knowledge, which is located irremovably from society much as embodied and embrained types of knowledge are permanently situated in their respective biological organ systems. Encultured knowledge "changes as society changes, it could not be said to exist without the existence of the social group that has it; it is located in society" (Collins 68). My software model also works with these types of ideologically-dependent stories; in this case the story scripts will need to require certain universal story elements similar to those found in the structure of the culturally common stories Schank describes in Chapter Three.

One additional type of knowledge that *EDNA-E* handles that is worth mentioning is emergent knowledge. Emergent knowledge relies on emergent properties of the storytelling system; this type of knowledge results from linking two stories together to create a new narrative. For example, in the *SparkleTech* story narrated in Chapter Three, emergent knowledge

is evident in the overall narrative Jane develops from the individual stories of herself, Chris, and Sandra. This type of knowledge is especially well-served in my software model using the story-linking function. This function can easily build comprehensive metanarratives using existing stories as the building blocks for this process.

At this point I have thoroughly explained the evolution of intelligent systems in Chapter Two, outlined narrative theories and models in Chapter Three, considered examples of narrative systems in Chapter Four, and described and modeled my own narrative KM system in Chapter Five. The conclusion up to this point has been concerned with discussing a few additional ideas about artificial intelligence and knowledge management and with showing how my software has approached and dealt with these issues. The remainder of this conclusion briefly explores the future of narrative knowledge management technologies.

In Chapter Three, I discussed the various levels of narrative theory from Mieke Bal's research: the fabula, the story, and the text. The future of narrative technologies will rely on advancements in virtual reality and augmented reality in order to expand the textual level, which equates to the medium in which a particular story is displayed. For example, Mariet Theune et al. write about the *Virtual Storyteller*, a semi-autonomous multi-agent system that inhabits a virtual environment and "aims at story presentation by a traditional storyteller: an embodied, talking agent that tells a story using appropriate prosody, gestures etc." (1). In this type of environment, the story is not confined to the two dimensional boundaries of a computer screen. The story can be presented "in the form of text or speech, or in the form of virtual drama, where the story is enacted on screen by animated figures in a virtual, graphical environment" (Theune et al. 2).

Interestingly enough, the authors of the virtual storyteller discuss the needs for two prerequisite properties for story plots: they need to be consistent, with natural event sequences and in-character actors, and they need to be well-structured, ala Freytag's triangle (Figure 71) in which they "should have a beginning, where some problem is introduced and the action rises, a middle, containing the action's climax, and an ending (in success or failure), where the action falls" (Theune et al. 2). What is interesting about this is the fact that XML files, as discussed in the Introduction of this dissertation, also can be structured to require validity (consistency) and to be well-formed (well-structured). This suggests that in the future we may be able to control the plots of virtual stories by manipulating simple text files in XML format much in the way they were used to configure program options and thematic dictionaries in Chapter Five.

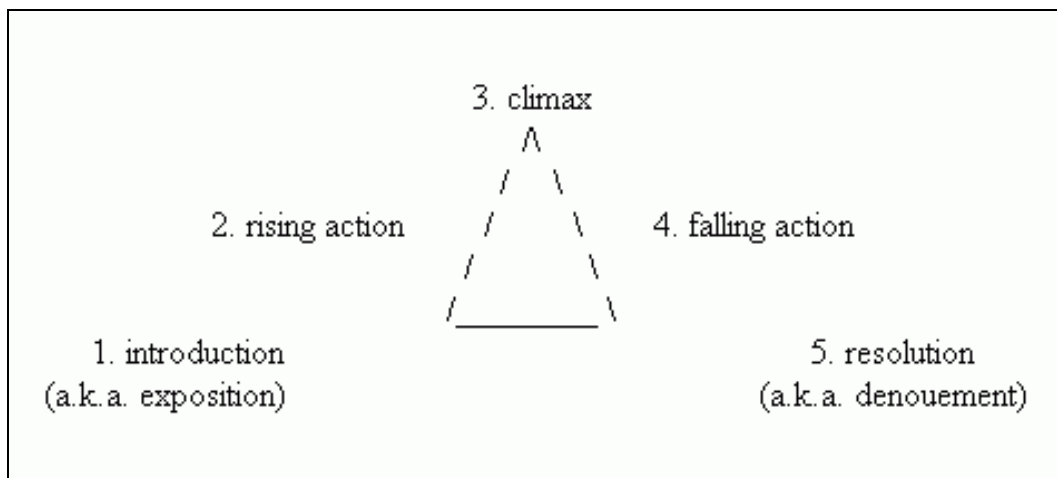


Figure 71: Freytag's Triangle (UIUC online)

Another storytelling system mentioned by Theune et al. is the Teatrix system for virtual drama. This model "is designed for collaborative story creation by children" (10). In this

environment, "... some of the story characters are controlled by the children using the system; the other characters are autonomous agents. There is also an omniscient director agent which can insert new items and characters into the story world, and which can control the characters' actions on behalf of the story coherence" (10). Here the computerized director agent is taking on many of the roles and responsibilities of the human story administrator in Chapter Five, which is a hopeful indicator that perhaps future models of narrative systems will indeed support more robust artificial intelligences. Even with the agent characters and the agent director, though, the actual stories are still generated by the children working with the story building-blocks provided by autonomous agents. In terms of knowledge management, such story building-blocks could be linked to real-world problem solving examples. For example, automated characters could be designed to create some type of conflict, and the director could instruct the children to write a story about resolving this conflict using available tools and resources.

Regardless of the specific mediums that future storytelling systems may require, it is clear that newer and faster technologies will not eliminate the need for stories and the storytelling process. In this dissertation I have shown how such a system can be created to serve as a knowledge management tool; it remains to be seen what other uses for narratives may emerge in future research. I will therefore conclude on a positive note, for I think I have explained that storytelling offers our culture much more than entertainment. Stories, when properly formed and verified by a human manager, can in fact form the data backbone of the next generation Semantic Web envisioned by Berners-Lee and other Internet visionaries. When the Semantic Web emerges, however and whenever that is, even Sven Birkerts and his apocalyptic

contemporaries will share their knowledge and wisdom through stories. Whether or not they use an automated storytelling system to do that, though, is another question entirely.

LIST OF REFERENCES

- Applen, J.D. "Technical Communication, Knowledge Management, and XML." Technical Communication 49.3 (2002): 301-13.
- Arrowood, L. F. et al. "Challenges in Applying Artificial Intelligence Methodologies to Military Operations." ACM Digital Library. 26 March 2002.
<<http://delivery.acm.org/10.1145/20000/12853/p408-arrowood.pdf>>
- AI. Dir. Steven Spielberg. Perf. Haley Joel Osment, Jude Law, Frances O'Connor, and William Hurt. WB Dreamworks, 2001.
- Bal, Mieke. Narratology: Introduction to the Theory of Narrative. Toronto: U of Toronto P, 1985.
- Baudrillard, Jean. Simulacra and Simulation. Ann Arbor: The U of Michigan P, 1994.
- Berners-Lee, Tim, Hendler, James, and Lassila, Ora. "The Semantic Web." Scientific American.com. 21 January 2004.
<<http://www.sciam.com/article.cfm?articleID=00048144-10D2-1C70-84A9809EC588EF21>>
- Berners-Lee, Tim. "What the Semantic Web Can Represent." The World Wide Web Consortium. 22 January 2004. <<http://www.w3.org/DesignIssues/RDFnot.html>>
- Bhabha, Homi. "The Other Question: The Stereotype and Colonial Discourse." Visual Culture: The Reader. London: Sage Publications, 1999: 370-78.
- Bicentennial Man. Dir. Chris Columbus. Perf. Robin Williams, Sam Neill, Hallie Kate Eisenberg, and Oliver Platt. Touchstone, 2000.
- Birkerts, Sven. The Gutenberg Elegies: The Fate of Reading in an Electronic Age. New York: Fawcett Columbine, 1994.
- Bolter, Jay David and Grusin, Richard. Remediation: Understanding New Media. Cambridge: MIT P, 1998.

- Bonabeau, Eric, Dorigo, Marco, and Theraulaz, Guy. Swarm Intelligence: From Natural to Intelligent Systems. New York: Oxford UP, 1999.
- Born, Rainer P. Artificial Intelligence: The Case Against. New York: St. Martin's P, 1987.
- Brackenbury, Ian F. and Ravin, Yael. "Machine Intelligence and the Turing Test." IBM Systems Journal 41.3 (2002): 524-539.
- Brookshear, J. Glenn. Computer Science. 6th ed. Reading: Addison-Wesley, 2000.
- Buchanan, Bruce. "Brief History of Artificial Intelligence." American Association of Artificial Intelligence. 26 March 2002. <<http://www.aaai.org/AITopics/bbhist.html>>
- Campbell, Neil A. and Reece, Jane B. Biology. 6th ed. San Francisco: Benjamin Cummings, 2002.
- Cass, Stephen. "A Fountain of Knowledge." IEEE Spectrum 41.4 (2004): 68-75.
- Christiansen, Hans-Christian. "Comics and Film: A Narrative Perspective." Comics & Culture: Analytical and Theoretical Approaches to Comics. Copenhagen: Museum Tusculanum P, 2000: 107-121.
- Coles, Stephen L. "An On-Line Question-Answering System with Natural Language and Pictorial Input." ACM Digital Library. 26 March 2002. <<http://portal.acm.org/citation.cfm?id=810577>>
- Collins, Harry M. "Humans, Machines, and the Structure of Knowledge." Stanford Humanities Review 4.1 (1995): 67-84.
- Crabbe, Ric and Dubley, Amit. "Artificial Intelligence FAQ." Carnegie Mellon University. 26 March 2002. <http://www-2.cs.cmu.edu/afs/cs.cmu.edu/project/ai-repository/ai/html/faqs/ai/ai_general/ai_1.faq>
- Crevier, Daniel. AI: The Tumultuous History of the Search for Artificial Intelligence. New York: BasicBooks, 1993.
- The Curta Calculator Page. 23 January 2004. <<http://www.vcalc.net/cu.htm>>
- Davenport, Thomas H. Information Ecology. New York: Oxford UP, 1997.
- Davis, Martin. The Universal Computer. New York: W. W. Norton & Company, 2000.

- Davis, Randall, Shrobe, Howard, and Szolovits, Peter. "What is a Knowledge Representation?" AI Magazine 14.1 (1993): 17-33.
- De Beauvoir, Simone. The Second Sex. New York: Knopf, 1989.
- De Landa, Manuel. "Virtual Environments and the Emergence of Synthetic Reason." The South Atlantic Quarterly 92.4 (1993): 790-802.
- De Weert, Egbert. "Contours of the Emergent Knowledge Society: Theoretical Debate and Implications for Higher Education Research." Higher Education 38 (1999): 49-69.
- Derrida, Jacques. Of Grammatology. Trans. Gayatri Spivak. Baltimore: Johns Hopkins UP, 1976.
- Desouza, Kevin C. Managing Knowledge With Artificial Intelligence. Westport, Connecticut: Quorum Books, 1979.
- Denning, Stephen. The Springboard: How Storytelling Ignites Action in Knowledge-Era Organizations. Boston: Butterworth Heinemann, 2001.
- Dreyfus, Hubert L. What Computers Still Can't Do. Cambridge, Massachusetts: The MIT P, 1997.
- Eisner, Will. Graphic Storytelling. Tamarac, FL: Poorhouse P, 1996.
- Feinberg, Cara. "Frontiers of Free Marketing." The American Prospect Online. 26 March 2002. <<http://www.prospect.org/print-friendly/print/V12/14/feinberg-c.html>>
- Fenichel, Otto. "The Scopophilic Instinct and Identification." Visual Culture: The Reader. London: Sage Publications, 1999: 327-39.
- Floridi, Luciano. Philosophy and Computing: An Introduction. London: Routledge, 1999.
- Gams, Matjaz. Weak Intelligence: Through the Principle and Paradox of Multiple Knowledge. Huntington, New York: New Science Publishers, 2001.
- Generation 5. Generation 5 Webpage. 26 March 2002. <<http://www.generation5.com>>
- Genette, Gérard. Narrative Discourse: An Essay in Method. Trans. Jane Lewin. Ithaca, New York: Cornell UP, 1980.
- Gibson, William. Neuromancer. New York: Ace Books, 1984.

Glassner, Andrew. "Interactive Storytelling: People, Stories, and Games." Virtual Storytelling: Using Virtual Reality Technologies for Storytelling. International Conference ICVS 2001, Avignon, France, September 27-28, 2001: 51-60.

Google.com. Google Search Engine. 22 October 2003. <<http://www.google.com>>

---. "Chatterbots". 26 March 2002. Google Web Directory.
<http://directory.google.com/Top/Computers/Artificial_Intelligence/Natural_Language/C_hatterbots/>

Güzeldere, Güven and Franchi, Stefano. "Dialogues with Colorful Personalities of Early AI." Stanford University. 26 March 2002.
< <http://www.stanford.edu/group/SHR/4-2/text/dialogues.html>>

Haraway, Donna J. Simians, Cyborgs, and Women: The Reinvention of Nature. New York: Routledge, Chapman, and Hall, 1991.

Harding, Sandra. Is Science Multicultural? Postcolonialisms, Feminisms, and Epistemologies. Bloomington: Indiana UP, 1998.

Harmon, Paul and Watson, Mark. Understanding UML: The Developer's Guide. San Francisco, California: Morgan Kaufmann Publishers, 1998.

Harrison, Claire. "Visual Social Semiotics: Understanding How Still Images Make Meaning." Technical Communication 50.1 (2003): 46-60.

Heidegger, Martin. The Question Concerning Technology and Other Essays. New York: Harper Torchbooks, 1977.

Hildreth, Paul, Wright, Peter and Kimble, Chris. "Knowledge Management: Are We Missing Something?" In Brooks, L. and Kimble C. Information Systems - The Next Generation. Proceedings of the 4th UKAIS Conference, York, UK (1999): 347-356.

Holland, John H. Hidden Order. Cambridge, Massachusetts: Perseus Books, 1995.

Hughes, Michael. "Moving from Information Transfer to Knowledge Creation: A New Value Proposition for Technical Communicators." Technical Communication 49.3 (2002): 275-85.

Intel Research. 2003. "Moore's Law." 16 January 2004.
<<http://www.intel.com/research/silicon/mooreslaw.htm>>

- Jackson, Jesse. Address. Presidential Campaign. Philadelphia, Pennsylvania. 16 January, 1984. PBS Great American Speeches. 30 December, 2003. <<http://www.pbs.org/greatspeeches/timeline/index.html>>
- Jacobs, Russel et al. "The Human Brain Project at Caltech." 16 January 2004. <<http://www.gg.caltech.edu/hbp/>>
- Kantrowitz, Mark. "Milestones in the Development of Artificial Intelligence." 26 March 2002. <<http://128.174.194.59/cybercinema/aihistory.htm>>
- Kohanski, Daniel. Moths in the Machine: The Power and Perils of Programming. New York: St. Martin's Griffin, 1998.
- Kolko, Beth. "Erasing @race: Going White in the (Inter)Face." Race in Cyberspace. New York and London: Routledge, 2000.
- Kotz, John and Treichel, Paul. Chemistry and Chemical Reactivity. 5th ed. Stamford: Brooks Cole, 2002.
- Kurzweil, Ray. The Age of Spiritual Machines: When Computers Exceed Human Intelligence. New York: Penguin Books, 1999.
- Lanham, Richard A. "The Electronic Word: Literary Study and the Digital Revolution." The Electronic Word. Chicago and London: U of Chicago P, 1993.
- Lieberman, Henry, and Maulsby, David. "Instructible Agents: Software That Just Keeps Getting Better." IBM Systems Journal 35.3-4 (1996): 539-56.
- Lunenfeld, Peter. The Digital Dialectic. Cambridge: The MIT P, 1999.
- Lyotard, Jean-Francois. "Can Thought go on without a Body?" The Inhuman: Reflections on Time. Palo Alto, California: Stanford UP, 1988.
- Maes, Pattie. "Agents that Reduce Work and Information Overload." The MIT Media Lab. 28 March 2002. <<http://pattie.www.media.mit.edu/people/pattie/CACM-94/CACM-94.p7.html>>
- The Matrix. Dir. Andy and Larry Wachowski. Perf. Keanu Reeves, Laurence Fishburne, and Carrie-Anne Moss. Warner Brothers, 1999.
- The Matrix Reloaded. Dir. Andy and Larry Wachowski. Perf. Keanu Reeves, Laurence Fishburne, and Carrie-Anne Moss. Warner Brothers, 2003.

The Matrix Revolutions. Dir. Andy and Larry Wachowski. Perf. Keanu Reeves, Laurence Fishburne, and Carrie-Anne Moss. Warner Brothers, 2003.

McCarthy, John. "What is Artificial Intelligence?" Stanford University Computer Science Department. 29 April, 2003. <<http://www-formal.stanford.edu/jmc/>>

McCulloch and Pitts' Neural Logical Calculus. 26 March 2002.
<<http://www.dlsi.ua.es/~mlf/nnafmc/pbook/node10.html>>

McDaniel, Rudy. "Object-Oriented Technical Communication." MA Thesis U of Central Florida, 2001.

McCloud, Scott. Understanding Comics. Northampton, Massachusetts: Kitchen Sink P, 1993.

Meehan, James. "TALE-SPIN." Inside Computer Understanding: Five Programs Plus Miniatures. Hillsdale, New Jersey: Lawrence Erlbaum and Associates, 1981. 197-226.

Minsky, Marvin. The Society of Mind. New York: Simon & Schuster, 1985.

Murch, Richard, and Johnson, Tony. Intelligent Software Agents. Upper Saddle River, New Jersey: Prentice Hall, 1999.

Norman, Donald. Things that Make Us Smart. Reading: Perseus Books, 1993.

Ong, Walter J. Orality and Literacy: The Technologizing of the Word. London: Routledge, 1982.

Pérez y Pérez, Rafael and Sharples, Mike. "Three Computer-Based Models of Storytelling: BRUTUS, MINSTREL, and MEXICA." Knowledge-Based Systems 17 (2004): 15-29.

PHP.net. "PHP General Information." 20 February 2004.
<<http://us4.php.net/manual/en/faq.general.php#faq.general.acronym>>

Post, Todd. "The Impact of Storytelling on NASA and Edutech." KM Review 5.1 (2002): 26-29.

Pratt, Jean A. "Where is the Instruction in Online Help Systems?" Technical Communication Online 45.1 (1998).

Reagan, Ronald. Address. Challenger Explosion Address to the Nation. 28 January, 1986. PBS Great American Speeches. 30 December, 2003.
<<http://www.pbs.org/greatspeeches/timeline/index.html>>

Roads, Curtis. "Research in Music and Artificial Intelligence." ACM Computing Surveys 17.2 (1985): 163-90.

Schank, Roger C. Tell Me a Story: Narrative and Intelligence. Evanston, Illinois: Northwestern UP, 1998.

Searle, John R. "Is the Brain a Digital Computer?" University of Southampton School of Electronics and Computer Science. 29 April 2003.
<<http://www.ecs.soton.ac.uk/~harnad/Papers/Py104/searle.comp.html>>

---. "Minds, Brains, and Programs". 9 March 2004.
<<http://members.aol.com/NeoNoetics/MindsBrainsPrograms.html>>

Sebesta, Robert W. Concepts of Programming Languages. 4th Edition. Reading, Massachusetts: Addison-Wesley, 1999.

Shackelford, Russell L. Introduction to Computing and Algorithms. Reading, Massachusetts: Addison-Wesley, 1998.

Sharples, et al. Computers and Thought: A Practical Introduction to Artificial Intelligence. Cambridge, Massachusetts: The MIT P, 1989.

Silverman, Kaja. "The Subject." Visual Culture: The Reader. London: Sage Publications, 1999: 340-55.

Sloman, Aaron. "What is Artificial Intelligence?" University of Birmingham School of Computer Science. 29 April, 2003.
<<http://www.cs.bham.ac.uk/~axs/misc/aiforschools.html>>

Smart, Graham. "Storytelling in a Central Bank: The Role of Narrative in the Creation and Use of Specialized Economic Knowledge." Journal of Business and Technical Communication 13.3 (1999): 249-73.

Smith, Anthony. Books to Bytes: Knowledge and Information in the Postmodern Era. London: BFI Publishing, 1993.

Stoll, Cliff. "The Curious History of the First Pocket Calculator." Scientific American 290.1 (2004): 92-100.

Tecuci, Gheorghe. Building Intelligent Agents. San Diego: Academic P, 1998.

Theune, Mariet, Faas, Sander, and Heylen, Dirk. "The Virtual Storyteller: Story Creation by Intelligent Agents." The NEC Research Institute. 9 January 2004.
<<http://citeseer.nj.nec.com/582379.html>>

Thomas, Lewis. "Computers." The Lives of a Cell. Viking Penguin, 1974.

Tonge, Fred M. "A View of Artificial Intelligence." Proceedings of the 21st ACM/CSC-ER International Conference. ACM Digital Library. 26 March 2002.
<<http://portal.acm.org/citation.cfm?id=810717>>

Turing, Alan M. "Computing Machinery and Intelligence." 26 March 2002.
<<http://www.abelard.org/turpap/turpap.htm>>

University of Illinois Literature References. "Freytag's Triangle." 23 February 2004.
<http://www.english.uiuc.edu/lit_resources/english%20102/miscellaneous/freytag.htm>

Vaux, Janet. "From Expert Systems to Knowledge-Based Companies: How the AI Industry Negotiated a Market for Knowledge." Social Epistemology 15.3 (2001): 231-45.

Von Neumann, John. "The Computer and the Brain." The Computer and the Brain. Yale UP, 1958.

W3C. "Extensible Markup Language (XML)." 22 January 2004. <<http://www.w3.org/XML/>>

W3Schools.com. "XML Introduction – What is XML." 22 January 2004.
<http://www.w3schools.com/xml/xml_what_is.asp>

Wallace, Richard S. The A.L.I.C.E. Artificial Intelligence Foundation. 21 October 2003.
<<http://www.alicebot.org/>>

Warner, Julian. From Writing to Computers. London: Routledge, 1994.

Warnick, Barbara. Critical Literacy in a Digital Era. London: Lawrence Erlbaum Associates, 2002.

Washington, Booker T. Address. Cotton State Exposition. Atlanta, Georgia. 18 September 1895. PBS Great American Speeches. 30 December, 2003.
<<http://www.pbs.org/greatspeeches/timeline/index.html>>

Weizenbaum, Joseph. Computer Power and Human Reason: From Judgment to Calculation. New York: W.H. Freeman and Company, 1976.

- Whitaker, Randall. "Overview of Autopoietic Theory." Association for Computing Machinery. 29 November, 2001. <<http://www.acm.org/sigois/auto/ATReview.html>>
- Wick, Corey. "Knowledge Management and Leadership Opportunities for Technical Communicators." Technical Communication 47.4 (2000): 515-29.
- Williams, Michael R. A History of Computing Technology. Los Alamitos, California: IEEE Computer Society Press, 1997.
- Zimmerman, Muriel. "Technical Communication in an Altered Technology Landscape: What Might Be." Technical Communication Online 48.2 (2001).