FROM TRADITIONAL TO STRUCTURED AUTHORING: TRANSFORMING PEOPLE AND INFORMATION

THESIS

Presented to the Graduate Council of
Texas State University-San Marcos
in Partial Fulfillment
of the Requirements

for the Degree

Master of ARTS

by

Cheri Lou Mullins, B.L.S.

San Marcos, Texas
December 2006

DEDICATION

For Nate and Toni

ACKNOWLEDGEMENTS

This project could not have been completed without the cooperation of several of my associates. First, I must thank my professional colleagues at the Society of Technical Communication, both those who took time out of their busy schedules to be interviewed for this research and those who contributed in a more informal manner. Although I cannot mention them by name here, I am grateful for their honest and straightforward input and for the stories they shared. Thanks to Richard Green for reading and commenting on several drafts of this project. Thank you also to my colleagues at GENBAND, Jim Odom, Len Kubiak, and Doug Thies, for their input and support. I am grateful to my thesis committee members, Dr. Pinfan Zhu and Dr. Deborah Morton for their input. And I am particularly grateful for the encouragement of Dr. Rebecca Jackson, my thesis director and professor, without whose encouragement, guidance, and support I never would have embarked upon this project.

This manuscript was submitted on 7 April 2006.

TABLE OF CONTENTS

Page

ACKNOWLEDGEMENTS v

LIST OF TABLES ix

LIST OF FIGURES x

CHAPTER

I. INTRODUCTION 1

Definition of Structured Authoring
  Traditional Writing Environments
  Structured Authoring Environments
Research Focus

II. RESEARCH METHODS 11

Secondary Research
Interviews
Interview Methods
Backgrounds of Research Participants

III. FINDINGS 20

What is Structured Authoring?
The Model for the Structured Authoring Paradigm
  Why Topics? Components, Topics, and Objects in Structured Authoring
The Role of XML and Semantic Markup in Structured Authoring
Model-Based Documentation: Advantages and Disadvantages

Visual Discontinuity: Separation of Content and Format
Using Schemas for Advanced Features
Specializing Elements, Topics, and Domains
Defining Document Ontologies and Discovering Data: Metadata and Topic Maps
Topic Maps
Intrinsic Benefits of Structured Authoring: Single Sourcing
Intrinsic Benefits of Structured Authoring: Faster Turnaround Time for Content Revision and Reuse
Extrinsic Benefits for Technical Communicators: Money
"Voice" and the Interview Data
Resistance to Change
Extrinsic Benefits of Structured Authoring: Job Growth and Security
Extrinsic Benefits for Technical Communicators: Technological Literacy
Identity Formation: "Technical" People
Does Structured Authoring Replace Writing?
Is Structured Authoring "Just Another Tool"?
Current Status: Hyper-Interest and Hypo-Adoption
Overall Benefits of XML-Based Structured Authoring
Effect on Academe

IV. IMPLICATIONS: WHAT THE FUTURE MAY HOLD 57

Changing Technologies and Methodologies
The Need for Knowledge Management
Effect on Technical Writer Voice
Effect on Academe
Effect on Industry
Embracing Structured Authoring
The Changing Role of Technical Writer
Areas for Further Research
Summary

APPENDIX A. 74

APPENDIX B. 83

REFERENCES 89

LIST OF TABLES

Table **Page**

1. HTML and XML Markup Examples 26

2. XML Example: Contact Data Base 32

LIST OF FIGURES

Figure **Page**

1. Compensation: USA and Central Texas 42

CHAPTER 1

INTRODUCTION

XML, shorthand for extensible markup language, is a hot subject in professional technical communication circles. Evidence of its growing popularity is everywhere. Professional conferences for technical communicators feature XML and related topics in a plethora of technological, management, and case study sessions presented to capacity audiences. Articles and web sites abound, discussing XML along with its associated tools and technologies. Meetings and seminars on the subject are some of the best-attended professional events. By all appearances, XML is a requisite technology for the technical communicator's toolkit. Indeed, taken by itself, XML might be dismissed as merely another new tool. However, in common usage with regard to technical communication, the term "XML" rarely refers to its namesake technology only, but rather to that technology as a building block of structured authoring. Much more than merely the tools, technologies, and methodologies it comprises, structured authoring is a burgeoning new technical communication paradigm that is poised to have a significant impact on the field of technical communication.

My interest in structured authoring originally was pragmatic: to determine how to capture the reported benefits of XML by using it as the technological foundation on which to develop technical documentation. Specifically, I was researching XML support for developing a single content base from which to quickly and easily create multiple outputs, in multiple formats, that include various parts of the source content. In addition to this *single sourcing* benefit, XML support for exchanging data between software applications and the documentation (*automatic data exchange* or *data interchange*) was compelling. Prior experience and my initial research led me to believe that a successful implementation would necessitate that the source content—the *knowledge base*—exist in an architecture based on discrete topics designed for reuse rather than in a "book" format, which is linear, monolithic, and not well-suited for content reuse. I began my initial research in order to determine the efficacy of converting my company's large legacy documentation set to a topic-based architecture utilizing XML for semantic markup. Structured authoring as a research topic for this thesis sprang from that original interest.

In my literature research, I found some information about structured authoring. However, the literature largely individually addresses the component parts of structured authoring such as XML, content management, and architecture. Because of this, it is difficult for technical communicators to get "the big picture" about structured authoring. To begin with, there currently is not an agreed upon term in the field of technical communication to describe this paradigm. The concept goes by many different names, each with its own focus. Content management, single sourcing, XML documentation, and structured writing are some common terms that often are used interchangeably. I propose *structured authoring* as a unifying term to discuss this new technical communication paradigm holistically, encompassing these technologies and methodologies where appropriate. As a contributing research methodology, I interviewed professional technical communicators who have experience in structured authoring in order to determine how professional technical communicators perceive this paradigm. The information they provided offers valuable insights into structured authoring, its potential benefits, and its importance to the field of technical communication.

Because structured authoring typically is not discussed in terms of the overall concept, it is sometimes dismissed as merely another new technology, methodology, or model. That is a mistake. Structured authoring, in fact, is causing a paradigmatic shift in technical communication that is dramatically affecting the roles, processes, and products of technical communicators.

Within the next few years, structured authoring using XML in a topic-based architecture will become the *de facto* standard for creating technical documentation.

Definition of Structured Authoring

Structured authoring as I define it goes by many names, among them: topic-based architecture (Hackos), XML-based documentation, content reuse, single sourcing, content management, and structured writing. Yet I found each of these terms and a host of others to be too restrictive. They name components of structured authoring rather than the whole. "Topic-based architecture" implies the model for structured authoring without the underlying technologies or actual writing. "XML-based documentation" and its derivatives imply the technology without the data model or writing. "Single sourcing" addresses a significant objective of structured authoring projects, creating a single source for content and, thus, facilitating content reuse. However, single sourcing does not comprehend the methodologies and technologies inherent in structured authoring nor any of a myriad of other objectives of structured authoring. Likewise, "content management" refers to a methodology and supporting technologies that are useful for management of technical information. A good content management system can facilitate content reuse. However, content management is not synonymous with nor limited to structured authoring; it is useful in both traditional and structured authoring paradigms. The closest term, "structured writing" is both too specific in its original information mapping sense and too general and "loose" in its current popular usage (Horn). In its original sense defined by Robert E. Horn and others almost forty years ago, structured writing refers specifically to writing that uses information mapping[1] as its concept of data modeling. In its popular usage, the term "structured writing" is the process of writing information content according to a model. However, structured writing does not enforce the content model but relies on the authors' adherence to it, typically through the

1. Information Mapping is a registered trademark of Information Mapping, Inc.

application of guidelines and templates. Each of these are important aspects of structured authoring. However, what makes structured authoring paradigmatic is that it encompasses all of these.

As I use it herein, *structured authoring* is a technical communication paradigm for planning, creating, and managing technical information in a structured manner according to a defined and enforced model, utilizing a markup language to create discrete information topics that are stored and managed independently from their format, organization, and structure in order to optimize the information content for reuse in other formats, for other purposes, and within other content. A topic based architecture provides the model for structured authoring and XML is its foundational technology. Although content reuse and repurposing are primary objectives, structured authoring may be motivated by various rationales and objectives and may lead to various results.

In order to understand structured authoring, we must understand how it has developed differently from the traditional writing environment.

Traditional Writing Environments

In traditional technical communication environments, technical writers write mostly linear, static documents. The advent of word-processing and technical publishing tools in the 1980s was a major technological leap that facilitated editing, provided the ability to "cut-and-paste" text, and enabled authors to send files electronically to the typesetter, thus shortening the time required for publishing by weeks. Incremental improvements over the years have further enabled technical writers to author documents that are already formatted in the target "style" and for the target distribution medium.

However, even with the advent of tagging languages such as the Hyper-Text Markup Language (HTML) and help and web authoring tools that utilize HTML and provide output in HTML, the majority of technical writers continue to produce long "manuals" in a linear, "book" style. This traditional technical writing approach has remained essentially unchanged for more than twenty years. Even online help and context-sensitive help typically are static documents. The book paradigm is important in technical communication; there are certainly applications that warrant a linear book. However, most technical communication audiences do not read an entire book or even an entire article. Rather, they skim the text or search if the "book" is online and peruse only the areas that specifically pertain to the information they seek.

Further, one of the features that many technical writers now consider intrinsic to the writing process—the ability to format text themselves as they write—has become burdensome. Documents that are traditionally developed for distribution as books, web pages, and help mix the style, content, and to a large extent, the structure of information. This results in monolithic information created for a particular presentation medium. Because the style, structure, and content is integrated, the output typically is optimized for one medium and inappropriate for a different medium or not quite optimal for either. Further, integrating the format with the content depends on an underlying presumption that the person who can author the content can also format it correctly for the audience, for the context, and for consistency with the prevailing corporate or other standards.

A further problem resulting from such an intricate link of style with content is that the content thus formatted may not be appropriate for reuse in other contexts. Reusable content in a traditional paradigm is generally limited to "cut-and-paste" or file duplication and then modification of the content for the next context. This is true even though some of the more powerful technical publishing tools support reusable text at some level. Maintaining the resulting two or more sets of near-identical information is cumbersome and most certainly will lead to errors in which some are updated and the others are not.

Even those traditional tools that provide some features to support reusable text do not provide a mechanism to enforce correct or consistent use. The result is that most authors continue to "reuse" content in the same way they traditionally have: by cutting and pasting from one document or section to another, moving all their styles—and any mistakes—along with the copied text. When content changes, the technical writer or editor making the change must use either a manual process or a semi-automated process using "search and replace" to update the text. Because the text has now proliferated to many different documents in many different contexts, both processes are resource intensive and pose a high risk of incomplete or inconsistent updates or inappropriate updates for a particular context.

Structured Authoring Environments

A structured authoring environment, on the other hand, is well-suited for supporting content reuse and repurposing, generally in a topic-based architecture. The content in a topic-based architecture is realized as collections of topics that can be assembled and reassembled as appropriate for various delivery contexts. Structured authoring inherently separates the authoring process that creates the content from the processes of editing, formatting, and production, which govern style, format, distribution medium, and to some extent, organization. This separation allows more focus to be placed on the quality of the authored content. The first technical communicators to implement structured authoring have tended to be part of large organizations that publish multiple versions of a large amount of data. In these large organizations, the technical communicators involved often become specialists in a particular role such as information architect, tools specialist, content author, content editor, style editor, or production editor. As it becomes more cost-effective for medium and small organizations and as more author-friendly tools become available for authoring and managing content, structured authoring is being adopted by medium to small organizations. In many of these organizations, the same technical communicators who develop content often also design the styles, formats, and outputs for the documentation. Even so, in structured authoring, these tasks are separate from developing the content.

Research Focus

At the beginning of this research, I had this question in mind: What role do the attitudes and personalities of the technical communicators themselves play in the adoption or non-adoption of the new technologies and methodologies required for structured authoring? My curiosity was piqued when I repeatedly heard in conversations with my peers at other companies that they want to transition to structured authoring using a topic-based architecture that utilizes XML, but have not yet made the move. Delving a bit deeper, it became evident that many of these technical communicators who continue to write in traditional technical communication environments know very little about this important new paradigm. Over the past few years, a knowledge and skills "divide" has developed in technical communication: the "haves" are literate in structured authoring technologies, methodologies, and skills while the "have nots" do not yet have this valuable paradigmatic literacy. This "technology divide" will widen over the next few years as structured authoring becomes the norm. More specialization of technical communication roles will take place and the technologically literate technical communicator will be in high demand.

As my research progressed, it became apparent that there is a much more basic question that first requires an answer before delving into the psyches of technical communicators who do or do not adopt the structured authoring paradigm. That question is: What is structured authoring and what is its importance to the field of technical communication? This thesis describes structured authoring in meta terms, providing an overview of the technologies and methodologies that are key to structured authoring and addressing the impact of this new paradigm on the field of technical communication.

Regardless of what terminology is used to refer to it, structured authoring as I define it here is already a burgeoning new technical communication paradigm. As large-scale structured-authoring projects continue and medium to small projects increase, a paradigmatic shift to structured authoring is occurring that is replacing the current traditional technical documentation paradigm. Structured authoring will result in further specialization of technical communicator roles and those technical communicators who learn the technologies and methodologies of this new paradigm will enjoy a host of opportunities in the job market.

The next chapter discusses the methods I employed to gather research data to support this thesis.

CHAPTER 2

RESEARCH METHODS

His aim was to furnish others with material for writing history...but men of sound judgement he has deterred from writing, since in history there is nothing more pleasing than brevity clear and correct.
Cicero, *Brutus* 262, tr. Hendrickson (Loeb)

A goal of this study of structured authoring is to provide an overview and a basic terminology of structured authoring and its key technologies and methodologies as well as to identify some of its more important potential benefits. In order to address my research question, I relied primarily on analysis and synthesis of secondary research. Therefore, a review of academic and professional literature and anecdotal research informs this basic research. Organizational narrative contributed a qualitative dimension to my review of secondary materials. Interviews with technical communicators who work or have worked in structured authoring environments complement the literature research. Their narratives and interpretations help to place the technologies and methodologies of structured authoring into context and their real-world evidence further emphasizes the importance of the structured authoring paradigm for the field of technical communication. In addition, many years of my own professional interest in this topic have yielded a wealth of experiential and anecdotal evidence, which I use to further contextualize my research.

According to Jane Ritchie in "The Applications of Qualitative Research Methods," a role of social investigation in qualitative research is to illuminate content (26). The research undertaken here is contextual in nature. That is, its purpose is predominantly to describe the "form or nature of what exists." The research findings contextualize not only the technologies described, but also certain aspects of the self-perception of the professional technical communicators who use them. The literature review, interviews, and anecdotal evidence show a changing landscape for technical communication as well as changing and expanding roles for technical communicators in this new paradigm.

Secondary Research

My foundational research was based on an extensive review of the available literature, both theoretical and practical, and personal notes from relevant conferences, seminars, meetings, emails, conversations, and interviews. From this secondary research, the concept and growing importance of structured authoring as a technical communication paradigm emerged. I reviewed both technological and social-based sources. The literature also provided illumination of the technologies and methodologies that structured authoring comprises. This review of relevant theories, methodologies, and technologies also served as a basis to inform and analyze the data gathered from interviews.

Organizational narrative played a large role in informing this research. In *Narrative and Research in Professional Communication*, Nancy Roundy Blyler argues for the value of narrative in the discipline of technical communication research as "one symbolic means by which reality is produced and reproduced." David Boje further proposes that organizational narrative is inherently fragmented, polyphonic, and collectively produced. According to Boje, what he refers to as "antenarrative" is constituted out of this otherwise disjointed "flow of lived experience." Boje further conceptualizes these fragmented slices of organizational narrative, or *antenarrative*, as "pre-narrative" that will become "world-changing" as it becomes narrative. According to Boje, organizations use antenarrative to create a collective memory. This concept of collective antenarrative provides a unifying thread for analysis of my research.

Interviews

I also conducted in-depth interviews of professional technical communicators who have experience in structured authoring and its associated methodologies and technologies. My sources for the interviews were practicing technical communicators, most of whom are fellow members of the Austin community of the Society for Technical Communication. Initially I was concerned about how my professional interest in structured authoring and close associations with my colleagues might bias the study. However, Lincoln and Guba point out that qualitative research is "confirmable," despite an inherent power variance (Bourdeau) or bias. I am very grateful to have had knowledgeable and willing research subjects who patiently tolerated my learning curve. Much like Cicero's conundrum in *Brutus* over the abundance that could be written about Julius Caesar, having studied such knowledgeable and accomplished subjects has provided me with a wealth of information that I cannot hope to completely explore within the limited scope of this thesis.

I interviewed colleagues who have or had a technical communicator role in completed, ongoing, and planned projects that fit my stated research criteria. Their stories provide a background against which the literature and anecdotal research can be qualified. The interview questions were designed to gather "real-world" information about the overall paradigm and the technologies and methodologies from professional technical communicators who work and operate in the thick of them. The study subjects also provided a wealth of information about the goals, challenges, expectations, successes, and rewards of the projects they embarked upon as well as how they perceive themselves as technical communicators. Learning the stories of these technical communicators provided insight into how they construct their workplace identities and perceive their worth within the context of this changing landscape, thus highlighting some of the more extrinsic benefits of the structured authoring paradigm, such as providing a technical career path for technical communicators (Silvi *et al.*).

In seeking subjects for this study, I chose a purposive approach. Purposive sampling is non-probabilistic, that is, not designed for either randomness or as a representative sample of the population. A *purposive* sample focuses on characteristics or circumstances relevant to the phenomenon being studied (Mays and Pope). I chose the three primary interview subjects based on the fact that, because of their backgrounds and experience, they each could provide a wealth of data relevant to this study. Before I began the formal interviews of these subjects, I performed theoretical sampling by first informally querying a number of people to narrow my focus and to decide which further data to collect. From analysis of this data, I chose the three primary subjects. From the initial queries, and then as I conducted each interview, I further refined my research questions in an iterative manner.

My primary interview methodology was narrative, and more specifically, antenarrative. I used Boje's concept of antenarrative to analyze the emergent themes from the in-depth interviews. In *Narrative and Research in Professional Communication*, Blyler (1996) argues for the value ethnography in qualitative technical communication research. I also used an ethnographic approach to gather data from these same research participants over the course of many months during informal conversations. In addition, I gathered supporting material from other colleagues in the form of antenarratives from conversations via email, at conferences and meetings, and in our workplaces.

Interview Methods

I recorded the formal interviews and then selectively transcribed them for review and analysis. The basic set of questions that I used as prompts are included in Appendix A. However, the goal was to ask open-ended questions that encouraged the subjects to speak openly in a narrative style.

Over a period of approximately seven months, I collected small ante-narratives from numerous colleagues. These insights were gathered over lunch, in meetings, from emails and phone conversations, and during social interactions. Both while transcribing and *in situ*, I attempted to reference and network the story fragments into narrative maps in order to understand the intertextual aspects of the respondents' stories and the dynamics of the stories across our intertwined social-professional networks (Boje). The goals in attempting to determine the connectedness of the stories across the social network were twofold. First, the connected stories bring together the antenarratives into a unified "lived experience" (Boje). Further, they provide insight into whether technical communicators in structured authoring environments have different perceptions about their roles and "voices" (Roberts, Lowry, and Sweeney; Korsgaard, and Roberson; Roberts and Lowry) in their organizations than do those in traditional environments.

Backgrounds of Research Participants

Rather than use pseudonyms to protect the anonymity of my research participants, I chose to use a metaphor from the concept of voice effect. Voice effect provides a theoretical underpinning for this research, in that the study subjects' interactions with me all involved the participants expressing their professional *voices*. All of the participants, both in the formal and informal interviews, were eager to have their say, whether they were relating essentially neutral information or positive or negative perceptions. The *value-expressive voices* that they shared with me showed a desire to express their experiences and opinions in this forum. According to Roberts *et al.*, people seek to have an expressive voice as a symbol of their value to other group members even though such expression may have no direct effect on their personal lives. This *relational model* of voice (Avery and Quiñones) states that these individuals' views are worth hearing and, thereby, convey a measure of social status.

The voices of three participants, **Voice 1**, **Voice 2**, and **Voice 3**, have been invaluable to me in this study. These three research participants, whose voices I labeled according to the sequence in which the interviews were conducted, are all experienced technical communicators. All are very active in the Austin community of the Society for Technical Communication and have served as board or committee members of that organization. Each has either done structured-authoring projects in the past, currently works in a structured-authoring paradigm, or is beginning a new structured-authoring project. Each of these three agreed to participate in in-depth interviews, which lasted from one-and-a-half to three hours. Their voices provide context for this research and illustrate the importance of the structured authoring paradigm for the field of technical communication.

Voice 1 is a 38 year-old, who describes himself as a "technical writer and tools hack, the tools guy, you know, anything sort of generally geeky, I'm the guy who does that." He has been a technical writer for about five years, and has been doing "tools" work along with technical writing for much of that time. He holds a B.A. in English Literature, an M.A. in Latin Literature with a minor in Greek Literature, and a Ph.D. in Classics. He works for a medium-large software development company in Austin, Texas. His title is Staff Technical Writer on what he refers to as the "wad of people" on the company's "unstructured" organization chart. He is part of an eight-member team of technical communicators.

Voice 2 is a 37-year old technical writer for a small software company in Austin, Texas. He holds B.A. in Plan II Liberal Arts Honors. He has been a technical writer for approximately fifteen years. He began his technical writing career in hardware technical writing and moved to software writing after about a year. His main focus of late has been creating online help. He is the sole technical writer at his company and is in the early stages of implementing "XML-based documentation."

Voice 3 is a 48 year-old "technical writer and manager" who describes himself as "specializing in software." He holds an undergraduate degree in English and took courses in programming, formal logic, and rhetoric as part of his course work. He is self-employed and "not currently engaged." He has been a technical writer for twenty-five years. He has worked on a major markup language documentation project, a dozen or so enterprise software projects, and served as “evangelist” for moving an entire organization to a structured-authoring environment.

In the next chapter, I elaborate my findings from this research in order to describe the structured authoring paradigm and its key technologies and methodologies.

CHAPTER 3

FINDINGS

Pelopidas the Theban is better known to historians than the general public. I am in doubt how to give an account of his merits; for I fear that if I undertake to tell of his deeds, I shall seem to be writing a history rather than a biography.; but if I merely touch upon the high points, I am afraid that to those unfamiliar with Greek literature it will not be perfectly clear how great a man he was. Therefore I shall meet both difficulties as well as I can, having regard both for the weariness and the lack of information of my readers.

Nepos, *Pelopidas* I

In this chapter, I describe the structured authoring paradigm along with its most significant intrinsic technologies and methodologies. I present findings that show the importance of this burgeoning structured authoring paradigm in the field of technical communication. Further, I describe findings that highlight the benefits of adopting a structured authoring paradigm.

In Chapter 1, I presented a working definition of structured authoring. I will use that definition as a jumping off point here to further elaborate the definition. Like Cornelius Nepos, writing the first "lives" in Latin in the first century B.C., I too grappled with the difficulty of adequately situating my topic without overwhelming the reader with details. For that reason, I have chosen to include in Appendix A more detailed descriptions of certain of the concepts mentioned herein that are associated with structured authoring.

What is Structured Authoring?

From Chapter 1, we have this working definition of structured authoring. *Structured authoring* is a technical communication paradigm for planning, creating, and managing technical information in a structured manner according to a defined and enforced model, utilizing a markup language to create discrete information topics that are stored and managed independently from their format, organization, and structure in order to optimize the information content for reuse in other formats, for other purposes, and within other content.

In the next sections, I will elaborate my findings about the principal points of this definition to more fully define structured authoring as a new technical communication paradigm.

The Model for the Structured Authoring Paradigm

Structured authoring is a technical communication paradigm for planning, creating, and managing technical information in a structured manner according to a defined and enforced model.

The key point here is that the structure is both defined and enforced within a model. The model for specific content itself—the *content base* or *knowledge base*—is its *architecture*. The outputs from the content—the "documents"—derive their structure both from the underlying architecture, which determines what information is available, and the model for the output, which determines what information is included in the output and how it is included.

Enforcement for adherence to the model can be at a holistic level or at a very granular level.

Structured writing as it was defined by Robert E. Horn is the precursor to structured authoring. Many of the principles that govern "good" technical writing today emanate from Horn's information mapping and structured writing model, which emphasizes using a systematic approach, which Horn calls information mapping, to "chunk" text into information blocks that are labeled and categorized by type ("structured"). The modern approach to structured authoring is similar in that it seeks to create information within a prescribed architecture. Content analysis serves to categorize information. Information is structured and cross-referenced in discrete topics according to its content and its categories.

The model for structured authoring is a *topic-based architecture*. As Robert Horn argues, a structured approach facilitates reader usability (2001) by presenting information in topic hierarchies—what Horn refers to in information mapping terminology as taxonomies (1993)—that quickly can be scanned and skipped through rather than in huge blocks of linear information that obscure the information that the reader is actually trying to find.

Like structured programming before it, structured authoring employs a top-down, systematic approach to design or "architect" information at the highest level of abstraction and then to create the details in manageable pieces. By implication, such writing necessitates the use of a standardized methodology for creating the model and creating content within the information architecture. DITA [pronounced 'Di-Tuh'], the Darwin Information Typing Architecture, provides a standardized methodology for creating a knowledge base in a topic-based architecture. The DITA model supports three primary *topic types*: concept, reference, and task.[2] A particular task, for example, generally has an associated conceptual topic and reference topic (Hackos). Each topic can have multiple related topics. Although these three types are standardized, the model can be collapsed or extended with specialized types.

Why Topics? Components, Topics, and Objects in Structured Authoring

The concepts of writing in topics and topic-based architecture are not new. Writers of all ilk, not only technical writers, have long written with a topical focus. From its use in Aristotelian rhetoric, *topos* (literally, 'place' or 'location') referred to a category or relation, an implication, from which many arguments and enthymemes (deductions or inferences) derive (Burnyeat). Conclusions of a certain form can be derived from premises of a certain form (Rapp). The topics and topic-oriented architecture of today's advanced technical writing technologies are not decidedly different.

A *topic* in a topic-based architecture represents a single concept, task, or reference item. On the website, *Cover Pages: Online resource for markup language technologies*, Robin Cover describes topics in this sense as easily

2. Robert Horn uses the term "information type" in information mapping (structured writing) and includes seven types: structure, concept, procedure, process, classification, principle, fact. Some of these additional types no doubt have merit. ISO 9002 and TL 9000 standards, for example define a process, procedure, and task or instruction as three related but different entities. The three broad types defined for a topic-based architecture, task, concept, and reference, are commonly used and understood by technical communicators and adequately define the types of information for the purpose of this study.

managed, "reusable, stand-alone units of information." Borrowing from the concept of reusable components in software engineering, a synonymous term for software data models is "components." Other terms frequently used for this concept are "modules" and "objects."

In order to minimize confusion about these terms, I will use "topics" to refer to items typically in the realm of information developers and "objects" to refer to items typically in the realm of software developers. The term "elements" and its generic form, "components," will be used to refer to units that are individually "tagged" in the content. An *element* is a component as small as a single character or as large as an entire document. The appropriate granularity for elements is a matter for much debate. Using too small a granularity wastes valuable time spent managing extraneous components. Using too large a granularity negates the benefits of component reuse.

Structured authoring almost inherently utilizes extensible markup language (XML) as its foundational technology for semantic tagging. In the next section, I discuss semantic markup and XML.

The Role of XML and Semantic Markup in Structured Authoring

A formalized encoding system is inherent in providing a standardized model and methodology for structured authoring. In order to conceptualize and effect the separation of the content from its format, structured authoring uses a markup language and semantic marking. *Semantic marking* refers to tagging or "marking" text semantically. Semantic markup improves the maintainability and reusability of information by identifying *elements* of the information by their function or content rather than by their presentational format or style.

The four most popular tagging "languages" currently in use are SGML, HTML, XML, and XHTML. HTML are XHTML are actually presentation applications that include some semantic tags and some presentational tags. They are neither extensible nor customizable. Documents tagged with HTML or XHTML largely become static documents. XML and SGML are the two most popular semantic tagging languages. Both use semantic tagging. Based on SGML, XML is a newer language that is easier to use and is also extensible, customizable, and can support data interchange. Using XML for semantic markup facilitates content reuse and content repurposing, two often-cited primary objectives of structured authoring. Although HTML and XML tags may appear similar, they are actually quite different. Table 1 shows a simple comparison of HTML and XML markup. Note that the HTML tags are largely stylistic, to determine the presentation of the text (font, bold, italics). The XML tags (step, cmd, p, def) are semantic, to define the content. The formats are determined by a separate stylesheet, itself an XML document, and applied to the step, command, paragraph, and definition when the document is "built." This comparison also illustrates one of the more simplistic enforcement aspects of XML. In the second example, the <p> tag is opened but not closed in the HTML. Although this is not valid, most browsers will assume the close tag for the paragraph. In the XML markup, the tag is closed: </p>. XML markup must be *well-formed* and valid in order for an XML parser to read it. This validity checking helps to ensure that the element thus tagged is structured as intended and that errors are fixed before the presentation files are created.

TABLE 1. HTML and XML Markup Examples.

HTML Markup	XML Markup
Markup	Markup
<font face=arial>Use the <b>start</b> command to start the program.</font>	<step>Use the <cmd>start</cmd> command to start the program. </step>
Result	Result
Use the **xyz** command to start the program.	Use the **xyz** command to start the program.
Markup	Markup
<p>This concept is called <i>structured authoring</i>.	<p>This concept is called <def>structured authoring</def>.</p>
Result	Result
This concept is called *structured authoring.*	This concept is called *structured authoring.*

In order to understand why XML is a foundational technology for structured authoring, it is helpful to know some of the history behind it. The older, more complex ISO 8879 standard from which XML derives, known as Standard Generalized Markup Language (SGML), can be used for structured authoring. In fact, because it is used in a large number of government and older publishing applications, SGML likely will continue in some form as a legacy publishing technology. However, XML is superior to SGML for structured authoring applications for three reasons. First, the complexity of SGML makes it difficult, and thus expensive, to learn and implement. Second, SGML is applicable to publishing only. By comparison, XML has been widely adopted by the software development community. The advantage of a common technology platform for software and publishing and the availability of technology expertise on the software side increases the likelihood of adoption of XML by the technical communication community. Third, as further new user tools, partner technologies, implementation methodologies, and uses for XML are developed, SGML and XML will continue to diverge. XML is growing in popularity in technical communication and publishing applications because of its relative ease of implementation as compared to SGML, its broader international appeal, its application for data exchange, and its widespread support within the software development community.

The XML recommendation was published in 1998 by the World Wide Web Consortium (W3C). The W3C Core Working Group that originally designed XML comprised industry experts from Sun, Microsoft, HP, Adobe, and others as well as researchers and experts from the University of Chicago, the NCSA, and the international Text Encoding Initiative to represent academe. According to the W3C, XML was originally designed to "meet the challenges of large-scale electronic publishing" and it has grown to provide a central role in data exchange. XML is an improved version of SGML that is easier to use, widely available, rich in features, and extensible.

Although XML began as a simplified subset of SGML for publishing applications, it has been widely adopted by software developers in new applications for data exchange on the web and between a variety of devices. Embraced by the software development community for web-based applications, this data exchange capability inherent in XML is also a compelling motivator for using XML to create technical documents. This cross functionality of XML results in a greater likelihood that companies will be willing to purchase the tools necessary and will have the technical capabilities available to support structured-authoring projects that are implemented using XML.

Another related markup language is the hypertext markup language (HTML). A common misconception is that XML it is an extension of HTML. Actually, both are developed from the SGML standard. However, HTML and the newer eXtensible HTML (XHTML) are similar to XML in outward appearance only. Like XML, HTML and XMTML were standardized under the auspices of the W3C for the purpose of publishing on the web. Each of these uses markup tags that have standardized meanings and that can be applied according to standardized rules. However, three very important differences exist. First, HTML and XHTML have a limited set of tags whereas XML can use an unlimited set of tags. Second, HTML tag names are defined by the HTML standard whereas XML uses semantic tags defined in the data model to suit the content. Finally, HTML tags force specific behaviors that are almost exclusively stylistic in nature (fonts, colors, layouts, and so forth). Link tags and scripting tags provide some structural features for HTML. In contrast, XML tags have no defined meaning at all until programs, scripts, or style sheets are applied. Because it does not provide semantic markup and does not have an externally enforceable data model, HTML can be used only as a presentation (output) format in the context of structured authoring. Text created using XML markup is *transformed* to HTML or XHTML.

In fact, the philosophy behind both XML and SGML is a separation of content, structure, and format (Dobratz). It is this separation that facilitates content reuse and repurposing, including translation. Because the format is thus separated from the content, a data model is required in order to provide the format and the rules for each type of document. In the following section, I discuss the advantages of a using a model.

Model-Based Documentation: Advantages and Disadvantages

As discussed earlier, the concept of the *data model* is foundational for XML and for SGML. The data model equates to a methodology. It is what sets these markup languages apart from other technical publishing technologies. The data model and its extensibility is what makes XML interesting for use in the programming world, thus providing a two-way collaborative environment for technical communicators and developers. In order to facilitate reuse and repurposing of documents, the XML model separates content from format. Further, XML enforces the model by requiring that a valid structure is specified for a document. The XML data model defines a valid structure for an XML document by specifying which XML tags, and by extension which elements, can be included and how they can be used. The two methodologies for data models are the DTD, which is shorthand for document type definition, and the schema. A defined DTD or schema specifies the *grammar* for an XML based document. DITA, introduced earlier, is a type of DTD. It is useful for a technical writer to know how a DTD or schema affects content. Except in "lone writer" situations, it often is not necessary for a writer to know how to actually create the DTD or schema or even how it will be applied in detail.

The following sections describe some of the advantages and disadvantages of the structured authoring paradigm, such as the separation of format and style from content, enforced content validation, and creation of reusable components.

Visual Discontinuity: Separation of Content and Format

Because XML is a markup language and not a presentation language, what appears on the screen depends on how the XML content is *rendered* based on the applied DTD and stylesheet. The trend is away from what-you-see-is-what-you-get, or WYSIWYG [pronounced 'wizzy-wig'], authoring to what Chris Lilley of the W3C calls what-you-see-is-one-possibility, or WYSIOP [pronounced 'wizzy-op']. What appears in a printed document or in help or on the web or in a different document may appear completely differently, depending on how it is rendered. This flexibility of presentation is a direct result of the separation of the content from the format. This may seem like a radical concept, but we actually have been working in WYSIOP for quite some time. One example is when a Postscript or other specialty font appears perfectly on the screen but does not render correctly in the printed or PDF document.

This separation of presentation and content is often cited as a problem within the structured authoring paradigm. Authors have become accustomed to formatting their own documents and seeing the immediate results. Their reliance on the format is so strong that they can have difficulty creating content independent of its format.

The fact that early XML editors lacked even the WYSIOP interface compounded this problem and left authors to write in what essentially amounted to code that looked more like the example in Table 2 than a technical document. **Voice 1** cited this as the "biggest pain point for the longest time—the most unfortunate thing" in the XML-based project that his company undertook. Many of the writers complained about the XML editor interface. In his retrospective analysis, he considered the project assumption that these technologies "could be used right out of the box" suggested "naïveté" in planning.

The dramatically different authoring environment that **Voice 1** and his peers experienced generally is not the experience in more recent projects. Virtually every modern XML editor includes both a WYSIOP interface and a code interface to ameliorate the effects of this potential writer disorientation. Using these newer-generation tools will help to protect one of the major strengths of an XML-based project: the separation of content and format that facilitates multiple uses in multiple outputs.

Using Schemas for Advanced Features

“Schema” has a specific meaning in the world of XML and structured authoring. It is not merely a model, as one might think of it based on the word’s semantic meaning. The XML *schema* data model not only defines a valid structure for the content, but can validate that the content itself is conforming. For example, where a DTD for a contact list can enforce that a phone number appears and even occurs in a certain location in a “document,” a schema can enforce that the phone number contains a country code, area code, exchange, and line number. This XML code example defines a ten-digit phone number as a string of three characters, a period, three more characters, another period, and four characters.

TABLE 2. XML Example: Contact Data Base.

Markup

```
<xsd:simpleType name="Phone">
   xsd:restriction base="xsd:string">
      <xsd:length value="10"/>
      <xsd:pattern value="\d{3}.\d{3}.\d{4}"/>
   </xsd:restriction>
</xsd:simpleType>
```

Result

512.555.1212

This simple construct ensures that the phone number has all the requisite parts, but it does not ensure that they are numeric values. A more complex definition might specify an optional county code, numeric values for each part of the phone number, and the acceptable ranges for each. This *data typing* present in the schema constrains the data thus defined. Advanced data typing is one of the most compelling arguments for using a schema for documentation. Data typing constrains data that becomes a part of the document and thus provides a level of validation for the document. It furthermore facilitates applications such as online ordering or customer support applications that use the validated information from the document, thus creating a kind of document-based application.

While they are beginning to be used extensively in software development applications that utilize XML, schemas in purely information development applications are not yet as common. The schema-based model does facilitate a greater level of validation and a more sophisticated data model. However, schema also significantly add to the complexity of the model, require a much steeper learning curve, and result in a much more verbose model as compared to DTD and DITA (Houser). Technical communicators who are not interested in the data typing and validation features of schema often regard the additional features as irrelevant for document-driven XML. Furthermore, the wide acceptance, availability, and long-term use of DTDs and the emerging emphasis on DITA further detracts from the appeal of schemas. The push to move to schemas will likely come from software developers for two-way data sharing between applications and documentation. Because of their complexity, schemas are not likely to be embraced for purely document-driven XML until they are implemented within the major technical publishing tools.

Specializing Elements, Topics, and Domains

Another important advantage of architected documentation is specialization. *Specialization* is an extension or derivation of a DITA content *element. Topic specialization* is the process by which new topic types are defined. A corporate web site, a targeted marketing data sheet, a technical product document, and a training document, for example, may all show reference information about a particular piece of equipment. Although there is some overlap in details, the focus, depth, and breadth of each of these is different.

Let us consider reference material for a telecommunications gateway. The web site might identify very high-level, "executive summary" type information about the gateway, such as the applications it supports and the standards with which it complies. The data sheet would include "speeds and feeds," the key marketing and sales points about the gateway, such as characteristics of the gateway and a diagram and details of the gateway in a given application. The data sheet may also include the executive summary information, expanding upon some, condensing some, and leaving out some. The product document would provide a broad base of technical reference information about the product, such as would be required by someone using the product or preparing to use it. Much of the same information that is required on the web and in the data sheet is required, but in much more detail. And some of the marketing-based information will be too high-level, and thus not applicable for the product document. Pricing is one obvious example. And, finally, the training material might include large portions of the technical information mixed with related industry and technological details.

While each of these represents a reference topic about the same telecommunications gateway, produced by the same company, and, possibly, by the same technical writer, each of the topics is specialized for its own purpose. They may require customized output formats or media and their depth and breadth of content varies, but they all maintain consistency in naming conventions, base content, corporate "look and feel," and development processes such as tagging conventions.

Domain specialization, closely related to topic specialization, provides a means of utilizing different vocabularies within different domains, such as for a programmer audience versus an executive audience or for an American English-speaking versus a British English-speaking audience. Specializations in DITA represent only the extensions, deltas, or constraints relative to existing topic types or domains, thereby reducing the work necessary to specialize.

Defining Document Ontologies and Discovering Data: Metadata and Topic Maps

Metadata is an important concept in classification and management of data. Metadata is often defined as "data about data," which is a simplistic but functional way to abstract it. Metadata in the context of information development has two purposes: as a means for discovery of particular data during information retrieval and as a means for classification of data.

Metadata is a fundamental concept for structured authoring, XML, and single sourcing or reusable components. Metadata tagging of individual articles or topics provides a taxonomic structure for the data. Determining the granularity and categories of content for metadata is a basic architectural task. The easiest and generally the most useful approach for metadata granularity is to use a variety of facets as opposed to attempting to use large, monolithic categories (Daniel and Burman). It is also important to balance granularity with maintaining a manageable number of unique metadata tags. Too many tags can lead to metadata not being applied or applied inappropriately.

Identifying the categories of metadata and then defining the acceptable metadata content for every category across the entire data set as a whole must be accomplished before any specific metadata is applied. The key to effective metadata use is consistency. Separating the definition of metadata from its application and beginning with a basic metadata specification facilitates consistency. The most widely-known specification for metadata is the Dublin Core. The Dublin Core standard specifies general, *horizontal*, metadata categories. Definition of domain-specific, *vertical* metadata will also be required. While metadata is extremely useful and currently is the prevalent method of managing data, topic maps enhance metadata and potentially can supplant it.

Topic Maps

The newer *topic maps* are a subject-based classification technique. Topic maps comprise only three categories: names, occurrences, and associations. While on the surface the fewer categories inherent in topic maps may make them appear to be more simplistic, their strength (and potential difficulty of application) is that each subject[3] can have multiple *topic mapping names* and multiple subjects can have the same topic mapping name. Furthermore, each topic mapping name can be assigned a limiting *scope*, without which it assumes unlimited validity.

Topic mapping occurrences function in topic mapped subjects much like an index in the book paradigm. Topic mapping occurrences use strings or URIs[4] to identify connected information resources. *Topic mapping associations* represent relationships between topic-mapped subjects. Associations such as "used-with," "based-on," "about," and "creator-of" facilitate the creation of an integrated network of subjects with clearly-defined associations.

Intrinsic Benefits of Structured Authoring: Single Sourcing

According to the STC *Single-Sourcing* web site, *single sourcing* is "using a single document source to generate multiple types of document outputs" or

3. The "topic" in "topic maps" is used in the general sense of the term, which refers only to the content value rather than the DITA-specific sense, which implies a specific architecture. Hence, the concept of topic maps uses "subject" in its nomenclature.
4. URL (uniform resource locator) is a familiar term to most. URI (uniform resource identifier) may be less familiar. Both specify the address of an Internet resource. While a URL specifies a specific file location, a URI is merely a unique name used to access the resource. It can represent a call to an application or a call to a data base instead of a fixed file location.

"workflows for creating multiple outputs from a document or database source." Single sourcing provides great "economies of scale" and is often the primary or at least one primary technological objective of structured-authoring projects. Single sourcing as it relates to structured authoring implies automatic management of a single source of data for use in multiple contexts, such as a reference guide, embedded help, and marketing materials. Single sourcing also includes automatic formatting of a single source of data for use in multiple outputs, such as PDF, Help, and HTML. Generally, a configuration management system (CMS) or some other data base system is necessary to store and manage the data in order to effectively single source all but the simplest of projects.

The ultimate goal of single sourcing is to save time and money. An added benefit of single sourcing done well is that it results in more consistent and potentially more accurate end documentation. Although difficult to quantify without specific comparison data, the cost benefit of single sourcing derives from three sources: quicker revisions of existing documentation, quicker introduction of documentation for new products, and quicker and easier translations and customizations to enter new markets (Hamilton).

Using a structured authoring, architected model of discrete topics that are well identified facilitates information reuse. During a new software release or other maintenance exercise, only the new data requires update and re-publishing for a given revision. The single-sourced data for multiple revisions leads to lower maintenance costs and quicker turnaround times. For new products, single sourcing provides economies of scale by precluding re-authoring or reformatting of information that has not been modified.

Many new products marketed by a company share a host of commonalities with existing products. Concepts, procedures, and interfaces are often the same or quite similar across product lines. Again, the combination of using well-architected discrete topics and single sourcing these topics from the CMS for multiple contexts provides the cost benefit.

The third typical scenario of single-sourcing cost savings is for translation or customization. Translation is a language and culture focused customization. The process as it relates to single sourcing is essentially the same whether the desired output is translated documentation or customized documentation. Therefore, the discussion here applies equally to both.

Cost benefits from single sourcing imbue translation projects. As in the other two instances, it is the integration of technologies and methodologies that benefits translation. In the initial translation, the structured documentation architecture and the XML marking itself clarify which text is and which is not to be translated, thus saving planning and management time and expense. For example, in our example on page 26, we may identify system commands, tagged with <cmd></cmd>, as text that is not translated. Thus, the translators would not only skip all occurrences of commands, but also could use the semantic tag to understand the context of the text. In subsequent translations, the amount of text to translate is reduced to only that text that has changed from a previous iteration.

In all three of these scenarios, using XML and separating formatting from authoring can dramatically reduce the amount of time and expense associated with revising and re-formatting text each time the content is revised. Some estimates are that as much as forty percent of translation cost is for reformatting text (Hamilton). Using XML to separate the content from the format virtually eliminates this cost.

Intrinsic Benefits of Structured Authoring: Faster Turnaround Time for Content Revision and Reuse

As budgets shrink and technical communication needs expand, there is little argument among professional technical communicators that we must do things in a different way than we have been in order to keep pace. In 1975, the Society for Technical Communication reported a five hours per page average time required to produce technical information. Based on nineteen-hundred and twenty working hours in a year, a technical communicator writing five pages per hour can produce almost four hundred new pages in a year. In a 1991 conference paper, "Developing a Business Case for an XML Authoring System," Jean Mercedes Hamilton uses a figure of one thousand pages in a year per author. Whether or not this figure is realistic is arguable. However, even divided in half, the resulting five hundred pages per year would translate to less than four hours per page. Clearly, expectations are for a quicker cycle to produce technical documents.

Extrinsic Benefits for Technical Communicators: Money

According to Jason Schweitzer, a Central Texas consulting company account manager, technical communication contractors with structured authoring skills (such as XML, DITA, schema, and CMS) can command from $40 to $56 per hour. According to the 2006 STC Austin Salary Survey[5], the average agency contractor rate in Central Texas is $30 per hour. Also, if we extrapolate from the median annual contractor earnings ($55,000) reported in the STC Austin Survey, we can derive a figure of approximately 1833 hours worked annually. Thus, using Schweitzer's figures and 1833 hours, we get an annualized range of $73,320 to $102,648. Because we have only these two figures, we cannot calculate a useful median; however, the average is $87,984. Reported median salaries in the surveys are $62,860 (USA) and $67,400 (Central Texas) for a technical writer; $70,660 (USA) and $66,900 (Central Texas) for a technical communicator with a Master's degree; and $85,090 for a technical communicator whose role is as a usability expert. Figure 1 shows this information in chart form. Clearly, industry places a high value on structured authoring and other specialized experience.

5. The 2006 Salary Survey reflects 2005 data self-reported by technical communicators in the Austin (Central Texas) area.

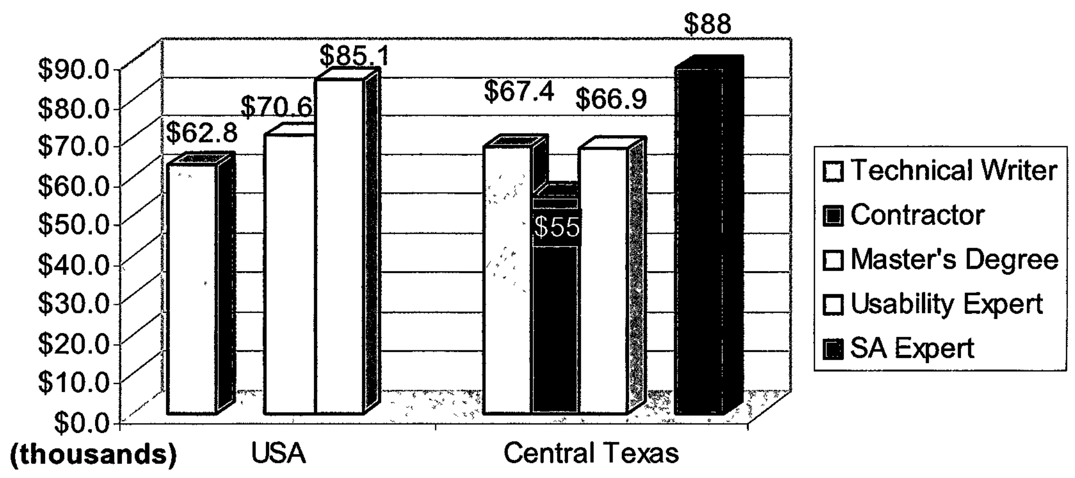

FIGURE 1. Compensation: USA and Central Texas.

Through formal and informal interviews, I discovered that structured authoring is also valued among technical communicators themselves. The next sections address what some technical communicators had to say regarding how structured authoring and its associated technologies affects them, their organizations, and their colleagues.

"Voice" and the Interview Data

The formal and informal research participants in this study are my colleagues from the STC and from industry. They are associates and friends. As such, they are not merely study subjects, they are collaborators in producing this study. They provided their inputs in formal interviews, at lunches, in email chains, during meetings and seminars, during networking times, and over cocktails. During many of the informal conversations, as in many before I began this present study, the conversation turned to the technical communicators' perceived lack of "voice" in their respective companies.

Voice refers to the concept of individuals in an organization being allowed to provide relevant input into decisions that affect them or affect the organization (Roberts, Lowry, and Sweeney; Korsgaard, and Roberson; Roberts and Lowry). Perceived voice in an organization has been shown to improve employee attitudes, commitment, output, and increase what Roberts *et al.* call "organizational citizenship." Further, perceived voice and participation positively affect employees' perceptions of interactional and procedural fairness, or "justice" (Schminke).

A lack of voice generally had not been my overall experience as a professional technical communicator. However, I was aware that my experiences could be unique in a number of areas: 1) Much of my technical communication career has been as a consultant. 2) For many years I have functioned in what I call "setup/startup" roles. In these roles, I helped to design "the system," not only for the technical communication departments, but also for corporate processes that affect those departments. 3) Much of my technical communication career has been in management.

According to Marshall Schminke, who closely relates voice effect in an organization with a sense of justice, leaders in organizations tend to believe that they have a voice and that the "system" is fair, as is evidenced by the fact that it "worked" for them. Schminke also posits that people who help design and implement the system also believe in the inherent fairness of the system. Further, as a consultant, I am specifically engaged with my clients to have a "voice."

Regarding the formal interview subjects, one of my interests was whether these participants had the same experiences of "voice" as I had or whether they would, as Schminke describes, "see the organization as inherently less fair" and their "voice" as smaller. During the interviews, each of the three participants described at least one instance of their *instrumental voices* in their projects and in their workplaces. **Voice 2** talked excitedly about a current situation in which he has an opportunity for design input into the user interface of the software program. **Voice 1** and **Voice 3** mentioned several examples of efforts leading to changes in the processes used by the entire group. **Voice 3** also talked about his role as the "proponent" for moving to the same structured authoring system across several writing departments.

Resistance to Change

In completing this research, I expected to uncover some of the major obstacles to completing a structured authoring project—and I fully expected a large number of them to stem from people's resistance to change, resistance to let go of creating a "book." This preliminary research bears out that premise anecdotally. This is certainly an area for further research. I often hear technical writers lament not having any "influence" in the "process" at their companies. When these conversations turn to specifics, what often they are lamenting is not having any say, any "voice" in the style of their documents.

Extrinsic Benefits of Structured Authoring: Job Growth and Security

It is not enough for a technical writer to be a good writer. Although this may gain one entry into the field, neglecting the "technical" part will inhibit advancement or worse. As **Voice 1** points out in his interview, one of the former writers on their project "resisted" the new tools and technologies and was later fired. When the field of software programming began, programs were created in text on teletype machines. It was not necessary to know sophisticated programming. A few commands and sequences were all that was necessary. Yes the programming task was complex, but that was the programmers' *raison d'être*. Now, programmers must be proficient in a myriad of technologies, languages (including XML and XSLT), and tools just to do their jobs. And those are all items outside of *core competencies*, programming, and their *domains*, the areas in which they program.

Technical communicators, too, must move outside their core competencies and their domains. Unlike engineering or technical marketing, it is very rare for a technical communicator to be hired solely for domain expertise. It is possible for a technical communicator to enjoy success by having an extremely broad and deep domain knowledge combined with excellent writing and organizational skills and an *aptitude* for tools. As **Voice 2** suggests: "They ought to be able to learn whatever tools we're using." However, like **Voice 1**, who says he "tricked [his] way into a tech writing job" by reading the first chapter of "every book on the technology in the bookstore," moving into a new domain or into an new technology environment will be difficult. Remember, also, what **Voice 2** suggests about a technical communicator with good tools experience. "I would expect to pay more for someone with a tools background."

In order to gain a competitive edge and to maintain job security, it is necessary for technical communicators to have robust skills and knowledge in three areas: writing, domain, and technology. By *writing* here, I mean writing, editorial, organizational, stylistic, analytical, and associated rhetorical implications as well as writing processes. *Domain* includes one or more areas of expertise such as telecommunications, computer science, finance, or medicine. *Technology* includes the basic technical publishing tools, office automation tools such as spreadsheets and presentation tools, the technologies and tools within their domains such as testing equipment and graphical user interfaces, and tools and technologies that are current or imminent in the field of technical communication. Currently, this points to structured authoring and an XML-based tool chain.

Consider the response of **Voice 3** when asked "What would you tell someone else embarking upon a similar project?"

> Take courses. I consistently steer people toward an education. Take writing courses. I believe you get a better and more complete picture of what technical writing is all about. Is it necessary?...[It is] unless you are extremely intuitive and...unless you are so smart that you can basically get a degree from your experience.

Here is where academe has a significant role. A technical writer cannot rely on writing skills and rhetoric alone. Academe must prepare students to enter the workforce and move directly into the world of the new tools, technologies, methodologies. Or as **Voice 2** says, they must have "used something besides Word." Universities and colleges must provide environments in both undergraduate and master's programs that are seeped in these technologies so that technical communication students are ready to enter the workforce when they graduate.

Extrinsic Benefits for Technical Communicators: Technological Literacy

One question that I asked every person I interviewed, including several whom I interviewed very informally at lunch over curry or at the cocktail hour over a chocolate martini, is: "Do you consider yourself to be a technical person?" Without fail and only in a few instances with any hesitation or reservation, all respondents answered in the affirmative. **Voice 1**, clearly the most technical of the three in-depth interview subjects, originally said that he was "not really technical going into it." He modified his answer later in the interview to "I always liked that kind of stuff [technology] and had a computer at home." This kind of qualitative research, of course, cannot uncover whether all of the twenty or so technical communicators I queried are in fact "technical," or to what extent if so. But it is interesting to note that they all unequivocally construct themselves as such.

Voice 1 defines himself as a "technical writer and tools hack, the tools guy, you know, anything sort of generally geeky, I'm the guy who does that." Throughout the interview, he emphasizes the tools and technologies and downplays the writing, particularly when asked about his writing skills.

> **A:** That [writing skills] needs to be something that you just have. It's important but less rare [than technologies skills, for which this subject considers himself an expert]. And expectations on the part of the audience are so low. They don't care about passive voice. They just want to get the goddamned thing to work. There are certain ways that we can make writing better, but we shouldn't spend too much time on that. On the other hand, some writers don't take ownership of content and make it their own in the doc.

Identity Formation: "Technical" People

Psychological essentialism has lent weight to the agency side of the socio-psychological structure-agency debate. According to Kenneth Gergen, psychological essentialism says that individuals possess specific mental processes or mechanisms that are central to the understanding of human action. If we are to believe this, one's social environment (structure), plays little role in their identity formation. However, according to Côté and Levine in *Identity Formation, Agency, and Culture*, "modern media technologies" are facilitating wider, more collaborative, and more "democratically institutionalized" constructions of self. This view swings the pendulum back and forth between structure and agency.

It is with this model of identity formation in mind that I formulated certain interview questions to get at how the interview subjects construct their professional lives. I asked each a series of questions to get at their perceptions of their technical selves. ("Do you consider yourself to be a technical person?" "Were you always technical?" "Were you mechanical as a child?") Some of the more interesting answers resulted from these questions, intended to determine the technical communicators' perceptions of themselves.

When asked if they were always technical, not all answered with quite as resounding a "yes," as they had when asked if they are technical (now). Nonetheless, after some self-debating, most decided that they always had a technical leaning. I also asked about the educational backgrounds of the respondents. Only one of seven thirty-something and younger technical communicators I asked this question had pursued a technical or technical communication education. That one person had also qualified her answer that she did consider herself to be a technical person but had not always been so. When questioned about why she had embarked upon a career in technical communication, she answered that she was "doing it [technical writing], so [she] may as well understand it." None of this group majored or minored in engineering or programming.

I asked four technical communicators who have each been practicing at least twenty five years all three questions (technical person? always technical? education?). Interestingly, all four (three men and a woman) answered that they are technical. All answered that perhaps they had not always been so but that they always liked to solve "problems" or "puzzles." Two of the men and the woman either hold engineering degrees or minored in engineering in their academic studies.

It is difficult to assess the value of these *hybrid* results, which mix probabilistic and non-probabilistic classifications of the data (Mays and Pope). However, they hint at the value that these technical communicators place on their various technical communication skills. Questions about what they deem to be their own greatest strengths, the most important characteristics for a technical writer, and the most important aspects of their projects clearly showed an emphasis on the technological literacy aspects of technical writing. Their answers also surface some interesting questions for further research. What role does age, generation, or gender play in technical communicator self-identities, technological literacies, choices for career preparation, and career domain choices?

Does Structured Authoring Replace Writing?

Although structured authoring can provide an excellent framework for creating content, it does not address the quality of the technical writing itself. Each of the technical communicators interviewed for this study emphasized the need for a technical communicator to have good writing skills. However, they each represented this need as being satisfied innately. When asked about required academic experience for a technical writer, **Voice 3** weights both writing and technical strengths.

> I'd be much more interested in their project work than their educational background. I would probably have bias toward someone with some kind of liberal arts training. I'd be suspicious of someone who didn't have that—whether or not they could write. But that's going to depend on if you're hiring someone who's solely tools oriented or if you expect them to write. If they're solely a tools person, an engineering background would be an advantage.

Anecdotal evidence suggests that technical communicators themselves subscribe to the "art" view of the writing part of technical communication—that good writing is innate. The interview responses and further anecdotal evidence reflect that technical communicators view the technology side to have components of both an art and a skill. The facilitating technologies and the technologies in the domains in which they write can be learned, but it is necessary for a technical communicator to be "technical."

Is Structured Authoring "Just Another Tool"?

A common argument against structured authoring is that it is "just another tool" or the technology *de jour*. Although it is true that the XML language could be replaced in a generation by a "bigger, better, faster" (or more likely, a “smaller, sleeker, more integrated”) technology, the basic construct will remain. XML is a simplification and extension of the earlier SGML definition that is still in use after three decades—a very long time, indeed, in technology years. In their overall concept and architecture, the two languages are not very different, which facilitates learning of XML for those familiar with the earlier technology. Several good interface tools are available for authoring topic-based content in XML, which has become a proven technology. XML also is becoming a *de facto* standard technology for content that is to be translated (Hertz).

Because of its extensibility, XML also has been adopted by programmers for content manipulation. This provides an excellent framework for writer-programmer collaboration in automating the production of some of the more repetitive and prosaic technical writing tasks, such as updating a parameter listing for every software release. Furthermore, the proliferation of related articles, classes, blogs, special-interest groups, and web postings are a testament to the fact that many professional technical communicators believe that modularly-architected XML is the correct methodology to achieve goals such as content reuse and repurposing for multiple contexts.

Current Status: Hyper-Interest and Hypo-Adoption

Large audiences for structured authoring and XML classes, SIGs, and so forth would lead one to believe that technical communicators are ready and eager to embrace the move from traditional authoring environments, at least in principle. However, even with an appealing business case, a plethora of readily available tools and technologies, and a desire to make the leap, relatively few technical communication departments outside of very large or very innovative companies have embraced structured authoring, XML, topic-based architecture, and true reusable components.

And, although there is a wealth of technology available to us, the vast majority of technical writers continue to write linear, static documents. And then we rewrite virtually the same documents over and over again. Relatively few technical communicators outside of large enterprises have actually embarked on authoring in the new paradigm.

It is widely accepted in the technical communication profession that some methodology is necessary that ultimately produces content that can be reused in other "documents." And we generally agree that structured authoring and some XML-like technology is required to do this efficiently and effectively.

Overall Benefits of XML-Based Structured Authoring

For the organization embarking upon a structured-authoring project, the potential benefits are threefold: better content management, reusable content, and economies of scale. Chunking of information into topics and the addition of an information hierarchy and metadata to the topics allows the content itself to be identified and categorized. Such categorization facilitates management of the content. Because these documents are readily accessible (and, ostensibly, written for reuse), they can more easily be reused for multiple media output and for multiple purposes.

Economies of scale are realized when the content is reused in multiple areas, say, as embedded content in a software product's user interface as well as in a technical document. This is the purpose of single-sourcing. By creating multiple articles of differing depth for a topic and carefully applying metadata to appropriately tag each level of content, an author can create content for a help system, a user interface, a marketing document, and a technical document practically simultaneously. This approach fine-tunes single-sourcing to a greater level of applicability. The benefits side of a business case for such an approach is compelling, but it must be weighed against the resources required for such an approach.

Effect on Academe

In the interviews I asked each person what characteristics he considers most important for a technical communicator, what strengths they bring to their jobs, and what characteristics they would look for when hiring a technical writer. Although almost every respondent answered differently on these questions, the common thread was that each considered himself to be a technical person and considered "technical depth" important for a technical communicator. Particularly telling was **Voice 1**'s answer to the question "What is the greatest strength that you bring to this job?"

> **A:** Technical depth and the ability to [pause]—depends on the aspect of job—the ability to have a fairly deep understanding of what I'm doing and to see opportunities for automation and pull it off. The mundane stuff. Take the human out of it. One of my favorite tricks is to find a way to get it out of the code and into the docs without a human re-typing it. When writing is "cut-and-paste," I hate it.

As he continues, he emphasizes writing and rhetorical skills.

> When it is describing, organizing, figuring out what the end user really needs, making it exist is satisfying. When it comes to automating a ref [a cross-reference], I see that as freeing up cycles to spend time on the higher level descriptive content that they need as well as the reference content that they also need.

Voice 3 also emphasized "writing and rhetoric" as necessary components of academic preparation for a career in technical writing. When asked about required work experience for a technical writer, **Voice 3** instead focused on the benefits of a technical education.

> Formal logic, which is typically taught in either a computer science or philosophy department is very helpful. And programming languages at a general level. The interesting thing is not just understanding one

> programming language, but understanding programming languages at a general level. It is a similar thing to understanding not just one religion, such as Christianity, but understanding...the common underlying themes.

The antenarratives provided during informal conversations and other anecdotal evidence also points to the necessity for a technical communicator to exhibit both technical skills and writing ability. Technical comprehension is necessary both in the domain that is the subject of the writing and in the technologies of modern writing. In a query at an STC meeting, all the technical communicators present in the conversation agreed with **Voice 3** that an understanding of programming languages is useful.

The implication of this for academe seems clear. In order to adequately prepare students for the workplace, technical communication curricula must focus on both writing and technology. An example might be to utilize writing exercises that both use the current "tools of the trade" and are based on students' domains of interest. Another example is requiring a technical cognate or minor.

CHAPTER 4

IMPLICATIONS: WHAT THE FUTURE MAY HOLD

We shall not cease from exploration and the end of all our exploring will be to arrive where we started... and know the place for the first time.

T.S. Eliot

The structured authoring paradigm promises a changing landscape for technical communication as well as changing and expanding roles for technical communicators. The term that the World Wide Web Consortium (W3C) XML Working Group, a team comprising software engineers and "experts in structured documentation and electronic publishing" (Connolly), chose to describe the final step in the process to publish an XML document is "transformation." More than a mere "transition" or "conversion," "transformation" indicates that the results are more than the sum of their parts. The technical communicators who take the time to learn and use these tools and technologies undergo no less a transformation. Using these tools, technologies, and methodologies to perform structured authoring does not result in technical communicators merely adding more tools to our toolkits, more technologies to our résumés, and more methodologies to our knowledge bases. All of these certainly occur and are important. Using these tools, technologies, and methodologies to perform structured authoring often changes the very way we think about information, the book paradigm, and our own professional lives.

Trends and technologies that hold promise to reduce the "time to market," increase the ease of content reuse for multiple contexts, or help maintain or reduce "head count" are of particular interest in business. As this study shows, structured authoring and XML are ready for adoption in mainstream technical communication to achieve these ends. Adopting structured authoring requires technical communication to move from a traditional, linear or "book" environment to a topic-based architecture in which technical content is authored utilizing technologies such as XML that separate the authored content from its format and output medium.

Such a move has far-reaching implications, both in theory and in praxis. Decision makers and practicing technical communicators must collaborate to plan and implement projects using these technologies. Herein are some of the implications of this study for technical communicators, for industry, for academe, and for further research.

Changing Technologies and Methodologies

Structured authoring is not a goal in itself. The goal is to be able to reuse and repurpose components. Or rather, that is a primary objective. The goal is to do more with less: fewer people, less time, less money. Reusable components, whether in documentation, software, hardware, tooling, or even people (which we soften to "cross training"), is one very common way of doing more with less.

In a presentation about single-sourcing, Candy Wong emphasizes that one of the limitations in single-sourcing may be a loss of applicability in some of the targeted outputs. In her example of authoring web content that can be presented across multiple devices, Wong talks about "high consistency" and "high customization." A benefit of single-sourced documents and user interfaces is that they are highly consistent in every user instantiation, which also results in what Wong calls "high user learn-ability." That is, they are easy for the reader or user to learn. The trade-off can be customization for the targeted audience or medium. By contrast, highly customized documents (or user interfaces) that are not single-sourced may more specifically target the intended audience or medium, but they lose the benefit of (guaranteed) consistency and present a "low user learn-ability" environment (Wong). A balance between all of these factors is required to determine which documents should be and which should not be single sourced.

There are countless specific vendor tools, both proprietary and open source, that are available to implement a structured authoring project. At its simplest, such a project can be undertaken in plain text, with manual chunking and identification of information and manually-placed cross-references to related topics. However, such a simplistic approach, while minimally satisfying a structured writing approach, surely would not accomplish the goal of doing more with less.

Further, writing team members would likely polish their résumés and head for the door. Tools that provide some level of sophistication, particularly in the area of user-friendly WYSIOP interfaces and technologies that are not egregiously difficult to implement are necessary to attract and maintain good technical communicators. Arguably, these tools may also provide efficacy for reducing errors and, thus, result in better documents. Tools that provide a certain degree of automation and verification are necessary to benefit from the underlying technologies and reduce the time to produce the documents.

The Need for Knowledge Management

I propose a companion term and concept for use with regard to structured authoring that is often overlooked, both conceptually and semantically. The underlying requirement for all of these technologies, tools, methodologies, and architectures is good *knowledge management*. I use "knowledge" instead of "information" because what must be managed is more than the information, the content. Further, the content management industry has co-opted the terms "information management" and "content management" to identify certain tools and services rather than for broad application to the concepts and processes. Knowledge management touches the information, the meta information, the model, the tools, the technologies, the people, and the processes.

Implementing a DITA methodology, attempting content management, or setting a goal for reusable content without understanding and properly managing the underlying information is an exercise in futility. The "architecture" part of DITA implies as much. However, for companies that do not have the massive "Machines" behind them as do IBM and other large corporations, this is the most difficult and most often overlooked part. I am constantly reminded of Blaise Pascal's quip, "I have made this letter longer than usual because I lack the time to maker it shorter." In order for any of these technologies, methodologies, or architectures to be efficacious, a primary emphasis must be placed on getting the underlying data under control, properly identified, and properly managed. A properly defined data model along with a version control system or content management system and strict adherence to standards is required for management of the content and the meta information.

Enterprise content management (ECM) proposes to provide a solution for knowledge management. However, implementation of ECM systems are priced out of reach for most small to mid-size companies. And, generally speaking, ECM focuses on objects of information (documents, for example), not the knowledge itself. A further complication is that content management currently exists in the realm of the information technology group, who certainly have a role, but perhaps should not be the arbiters of the information itself. A properly implemented ECM system that is always and consistently used can provide a means for management of the content, the meta information, the tools, and possibly, the model and the processes.

The final two items are the most difficult: technologies and people. People management here does not refer to personnel management. With regard to knowledge management, it refers to capturing their knowledge. The biggest difficulty is that people in an organization often do not want their knowledge captured. Knowledge is, after all, power. Open-standards technologies make this task easier in some ways, in that information about those technologies often is readily available from multiple external sources. These technologies also tend to encourage open communication among adopters, which becomes a source for processes and other information.

The subjects of this study have all functioned in roles as knowledge managers and it is clear from their repeated emphasis on proper modeling and organization that they have a clear understanding of these precepts. "Early adopters" in larger corporations have begun to speak at conferences and in other open forums. In Austin, the Central Texas DITA User's Group meets monthly and regularly features speakers who have implemented successful structured-authoring projects. It is imperative that individuals, user groups, and the STC, ATTW, IEEE-PC and other professional technical communication organizations share knowledge about these tools, technologies, and methodologies with their colleagues. It is also imperative that industry and academe encourage and support these efforts.

Effect on Technical Writer Voice

The lament of the technical communicator who does not have stylistic input will be exacerbated with a paradigmatic shift to a structured-authoring environment. In structured authoring, an architectural separation exists between developing content and developing format and organization.

New technologies, tools, and the resulting dynamic "documents" themselves necessarily have a much greater degree of style neutrality than do "books." "Neutrality" in this case refers to the Bakhtinian sense of heteroglossia or "internal differentiation or stratification" (Honeycutt), in which the word [and, by extension here, document or style] alone is neutral: it is its interpretation within a cultural context that makes the meaning. In "Discourse in the Novel," Bakhtin explains his concept of neutrality.

> The word in language is half someone else's. It becomes "one's own" only when the speaker populates it with his own intention, his own accent, when he appropriates the word, adapting it to his own semantic and expressive intention. Prior to this moment of appropriation, the word does not exist in a neutral and impersonal language. (Bakhtin, from Honeycutt)

Such documents or topics must stand alone on their own merits. They must use language that is sufficiently neutral that it is appropriate for numerous scenarios and applications: technical documents, technical marketing documents, help embedded into a software program, and proposals, to name a few.

Effect on Academe

In the interviews, I asked what characteristics they consider most important for a technical communicator, what strengths they bring to their jobs, and what characteristics they would look for when hiring a technical writer. Although almost every respondent answered differently on these questions, the common thread was that each considered himself to be a technical person and considered "technical depth" important for a technical communicator. Particularly telling was **Voice 1**'s answer to the question "What is the greatest strength that you bring to this job?"

> **A:** Technical depth and the ability to [pause]—depends on the aspect of job—the ability to have a fairly deep understanding of what I'm doing and to see opportunities for automation and pull it off. The mundane stuff. Take the human out of it. One of my favorite tricks is to find a way to get it out of the code and into the docs without a human re-typing it. When writing is "cut-and-paste," I hate it.

As he continues, he emphasizes writing and rhetorical skills.

> When it is describing, organizing, figuring out what the end user really needs, making it exist is satisfying. When it comes to automating a ref [a cross-reference], I see that as freeing up cycles to spend time on the higher level

> descriptive content that they need as well as the reference content that they also need.

Voice 3 also emphasized "writing and rhetoric" as necessary components of academic preparation for a career in technical writing. When asked about required work experience for a technical writer, **Voice 3** instead focused on the benefits of a technical education.

> Formal logic, which is typically taught in either a computer science or philosophy department is very helpful. And programming languages at a general level. The interesting thing is not just understanding one programming language, but understanding programming languages at a general level. It is a similar thing to understanding not just one religion, such as Christianity, but understanding...the common underlying themes.

The antenarratives provided during informal conversations and other anecdotal evidence also points to the necessity for a technical communicator to exhibit both technical skills and writing ability. Technical comprehension is necessary both in the domain that is the subject of the writing and in the technologies of modern writing. In a query at an STC meeting, all the technical communicators present in the conversation agreed with **Voice 3** that an understanding of programming languages is useful.

The implication of this for academe seems clear. In order to adequately prepare students for the workplace, technical communication curricula must focus on both writing and technology. An example might be to utilize writing exercises that both utilize the current "tools of the trade" and are based on students' domains of interest. Another example is requiring a technical cognate or minor.

Effect on Industry

Multi-functional applicability—reusability—promises large economies of scale for structured-authoring projects. These economies affect "the bottom line" for industry. Capturing and sharing project data will allow others to better identify a realistic and attractive business model. Academe providing educational, modeling, and theoretical support will further serve to provide a broad-based body of knowledge and qualified practitioners. This combination will prompt industry to provide the financial, human, and time investments to implement structured authoring projects using XML and these other new technologies.

Embracing Structured Authoring

Structured authoring cannot be done in isolation. It is immanently and absolutely a collaborative endeavor. Writers who do not collaborate well and writers who insist on adding their own "personality" to technical documents will have difficulty with the structured authoring concepts. Although not inconsequential, mastering the DITA/XML concepts of elements, attributes, and hierarchical documents are the least difficult. The greater difficulty is in adjusting to the necessary workflow for structured authoring—the process for which the typical author has little or no say. It has been my experience that even enforcing a style guide and templates can prove difficult, with some writers always discontent that it doesn't "fit" and wanting to customize the templates and styles to suit what they have done before or what they would like to do differently. The structured authoring paradigm takes the need for consistency to a much higher plane.

Because structured authoring exists within a framework of an architecture, it is necessary for every construct within the architecture to comply with the overall design. This is not a new concept. The metaphor itself is from the building industry. Civil engineers, general contractors, builders, construction workers, and interior designers have long adhered to the constructs set out by the architects of a building project. The civil engineers verify that a particular design can be realistically implemented, and by what means. The general contractors manage the entire process. The builders decide how to implement it. The construction workers (the authors for our analogy) create the actual building, often in isolation from other tasks going on simultaneously, but always strictly adhering to a specific plan. When changes are warranted, they are communicated back up so that the plans can be modified. In structured authoring, the "worker" may similarly create content that he neither designs, nor engineers, nor manages, nor builds.

Although many authors welcome the removal of these "distractions" to writing, some see it as removing their ability to "really write." Again though, this concept is neither new nor radical. This has long been the premise for most software development. Furthermore, a large part of code that software programmers previously developed themselves is now available in componentized software "stacks" that are simply added into the design. Much like some technical writers' fear today with the new technologies, software programmers feared that the availability of these self-contained components would take away their jobs or make them less interesting. Quite the contrary, many believe that they now have more freedom to do more important and more interesting work. After their initial resistance to change passes, technical communicators surely will see the benefit of structured authoring in a topic-based architecture, as well. The need for technical documentation will not go away with structured authoring or reusable components; it will merely become more specialized. Clearing away some of the clutter in our current documents will help us identify those opportunities for specialization.

The Changing Role of Technical Writer

A framing question for this study is how transitioning from traditional to a structured authoring affects a technical communicator's identity, position, and practices? (Grodin) The corollary is how it affects his or her perception of other technical communicators. The responses of **Voice 3** are typical of all three of the interview subjects. When asked about his role as compared to the "average" technical writer, **Voice 3** provides the following response.

> The fact is, most writers probably are not as interested [in technology] as I am. Most writers probably don't know as much about the underpinnings of the software. And don't care as much. Most would just like to interview the developers and write something that's—[trails off]. [They are n]ot interested in automation really. Some percentage are. But not most. I don't know if that will change [trails off].

When asked about whether and how they would differentially value the role of technical writer or a "tools person," **Voice 2** and **Voice 1** provided very similar responses. **Voice 2** has this to say.

> I guess I would expect to pay more for someone with a tools background under the assumption that there's fewer of them out there. Especially since XML knowledge, I don't think is—in the writing community—spread all that far, generally. When I think about hiring a writer—discounting the whole tools idea, I'm not too worried about what tools they've used or have familiarity with. They ought to be able to learn whatever tools we're using. You've got to assume fairly primitive XML tools but rather

> advanced publishing tools. I'm going to want to know that they've used something besides Word. They know how to format a document for the long haul. If they've done tech writing projects. That they have some ideas about issues such as cross references. Doc maintenance. And then it's fairly standard issues such as: Can your rely on them? What are their samples like?

These responses beg the question: What is the role of a technical communicator? In order to be responsive to these demands, technical communicators must adapt to the changing environment and adopt the current technologies and methodologies. The ability to write well has become a given requirement in the new environment. With the separation of content and format that the new methodologies impose, a writer cannot rely on format; the writing must stand on its own. Further, writers who also master the subject domain and the XML-based structured-authoring along with its tools and technologies will be more employable and enjoy greater compensation.

Areas for Further Research

One thing that was clear when I began this research is that technical communicators, or their companies, have a certain hesitancy about deploying XML. Whether this is voiced or not, the facts are that of at least two dozen professional technical communicators that I approached, only a small handful are actually using XML or actively planning or starting an XML-based project. According to Debbie Wiles, in "Ethical Insights in XML and Single-Sourcing," this same attitude was present at the 2000 STC Conference.

When I attended the 2005 STC Conference, there were a number of sessions, panels, and workshops on XML and DITA. All were very well attended. However, as I talked to my colleagues in these sessions, most had not yet made significant progress in implementing XML in their information development, even if their software development group used XML to some extent. In almost one voice with many of my colleagues whom I approached as potential subjects for my current research, these technical communicators were still in the "analysis" or "research" phase for implementing XML. One has to ask: Why is this so?

The XML, DITA, Structured FrameMaker, and similar technical sessions, case-study sessions, and workshops at the conference had standing-room only and overflow room crowds. Other interesting and meaty sessions, such as a fantastic panel session about offshoring of technical communication jobs, had tiny audiences by comparison. Clearly, technical communicators are interested in deploying these technologies.

This question, in fact, was my original research focus. However, as clearer heads prevailed, I moved to the question herein presented. Nonetheless, the question stuck in my mind and continues, stubbornly, to reside there. Wiles, again, proposes that her tentative research indicates that the cause may lie in the fact that the planning and modeling aspects of implementing a successful XML-based project—the architectural design work, what she rolls up into a simplification of "intelligent XML tagging"—is overly daunting. And, as Wiles so eloquently points out, this process of identifying and defining the architecture for the content must be done on "all of your content—all of your content [sic]" before undertaking even a small project.

Although I agree with Wiles and my research further substantiates her premise, I am not sure that this is the only reason. Technical communicators are not known for turning away from such a challenge. Remember the similar answers of those four forty-something technical communicators when asked if they have always been technical? "I always liked puzzles." What greater puzzle could there possibly be than to model, characterize, and identify "all of your content"? I continue to believe that there is more to this hesitancy. It is not simply the tools, the technologies, the process. Although each of these may certainly be a factor.

This research provides only a sweeping overview of structured authoring along with some of its primary supporting technologies and key benefits. It broadly contextualizes technical communicators' roles, places, and identities in the structured paradigm. This topic offers many avenues for further exploration. Explanatory research to further characterize both the technical communicators' technologies and their lives is also in order. In the case of the technologies, particularly with regard to the apparently slow adoption of XML, additional evaluative research could determine the effectiveness of these technologies (Ritchie, 27).

Summary

The XML-based technologies, methodologies, and tools warrant recognition as the prevailing "tools of the trade" for the structured authoring paradigm of technical communication. Just as structured programming has been refined by object-oriented programming and reusable components in the past few years, refinements will also occur in structured authoring. However, the basic premise of structured authoring will remain valid even as the tools and interfaces become more plentiful and more mature.

XML is relatively simple to learn and use across any platform. It is also increasingly being used in the development of document-based and other applications. Furthermore, XML is a relatively mature language, greatly improved from its predecessors: the overly-complex SGML and presentation-oriented HTML. Because of this and because of its extensibility, XML will figure prominently as a technical communication facilitating technology for many years. Technical communicators who want to be in demand and expand their options in this environment will require solid writing skills, a firm technological foundation in one or more domains, and robust skills in XML and associated structured-authoring tools, technologies, and methodologies. In order to prepare students for such roles, academe must incorporate into their curricula relevant tools, technologies, and methodologies as well as writing opportunities within the students' domains of interest.

APPENDIX A

GLOSSARY OF RELATED CONCEPTS

The following paragraphs provide more information about topics presented in this thesis as well as closely related topics.

Component

In order to reuse information, you break it up into chunks that are variously called “components,” “modules” or “objects.” A component of information can be as small as you want it to be, although it’s usually of a size worth managing separately. For example, if you’re creating books, you could create components at the level of a section or sub-section, but you could also create components for warnings and copyright notices.

Darwin Information Typing Architecture (DITA)

The DITA data model is an information management technology that capitalizes on the reuse concept. Although the nomenclature implies that it is itself an “architecture,” the Darwin Information Typing Architecture is more of an architectural methodology for authoring, producing, and delivering technical information. DITA design principles are used to apply information types to "topics," which can then be used in multiple media outputs and for multiple purposes.

The goal of DITA is to provide a consistent methodology to create modular technical documents that are easy to reuse in multiple outputs for multiple devices (via topic and domain specialization) without excessive compromise on stylistic considerations.

DITA was developed by IBM and now is owned and managed by the OASIS standards organization. It includes its own set of abbreviations, acronyms, and specialized terminology. The DITA data model is based on the XML language, operates on topics, and utilizes other constituents of XML.

DocBook DTD

The DocBook DTD developed almost two decades ago for IBM to create SGML-based technical documents is not technologically different than the current DocBook DTD for XML. The massive size of DocBook, which has grown to more than three hundred tags, can make it difficult to use effectively. There are hundreds of DTDs available, such as MIL-SPEC and MIL-STD for the military, but DocBook is often used for technical documents.

Document Type Definition (DTD)

The document type definition is almost always referred to by its abbreviation, DTD. The DTD has its roots in the older SGML technology. A DTD enforces structure based on placement and correlation of tags. For example, a DTD for a contact information sheet may specify that name, address, phone number, and email tags must be present and that a company tag and title tag is optional. The DTD may also specify the sequence required or that any sequence is acceptable. Customized DTDs can be used to create multiple outputs for multiple contexts. It is typical to have a master DTD and additional DTDs to specify deltas from the master.

Dublin Core of metadata

The *Dublin Core* is set of fifteen elements for electronic resource discovery. The Dublin Core Network Working Group is a part of the Internet Engineering Task Force of the Internet Society. The elements fall into one of three categories that further define the data. *Content metadata* consists of seven elements to identify the content itself: title, subject, description, type, source, relation, coverage. Intellectual property metadata consists of four elements to identify who owns the data: creator, publisher, contributor, rights. *Instantiation metadata* consists of five elements that identify a particular published instantiation of the data: date, format, identifier, language. While metadata is extremely useful and currently is the prevalent method of managing data, topic maps enhance metadata and potentially can supplant it.

Editor

For XML based projects, the beginning of the tool chain (at least from the technical writer's point of view) is some kind of *editor*. Although it is possible to create XML-marked content directly in text files, a front-end editor with some kind of WYSIWYG capability is much preferred and easier to use for most technical writers. Editors also generally provide other useful features. *Content assistance* identifies valid XML elements constraints. *Syntax checking* or *highlighting* marks XML syntax errors so that they can be fixed before the file is processed. *Validation* validates the XML document based on an associated *grammar* (DITA or a DTD such as DocBook). The ability to flip between a WYSIWG *preview window* and a *source-code viewer* facilitates authoring by enabling continuous authoring rather than periodically transforming to determine the effects of editing. Further, some editors provide integration into a *version control system*, such as CVS or Subversion, which facilitates change management. Typical XML editors are XMetaL, Arbortext (Epic) Editor, Eclipse, WTP, and so forth. After content is created, marked up with XML (*edited*, in XML terminology), it must be transformed.

HTML. Hypertext Markup Language

HTML began as much simplified and limited version of SGML. In the context of structured authoring using XML, HTML is an output format. Text created using XML markup is *transformed* to HTML or Extensible HTML (XHTML).

Infotyped topics

Information typing and *topic typing* refer to particular information types that can be created. For the sake of simplicity, we will restrict these to the basic DITA topic types of concept, task, and reference, which are widely known and utilized in technical communication. This DITA feature provides a powerful structured means for extending and fine-tuning the topic types.

SGML

XML is based on the Standard Generalized Markup Language, SGML, an almost 30-year old technology. SGML is still in use today. However, the original interfaces to SGML were either too expensive for many companies or lacked author-friendly interfaces. As a result, only very large or highly technologically focused technical communication groups who had information architects and IT specialists adopted SGML until the advent of FrameMaker + SGML. FrameMaker + SGML is a commercial tool that provides both a familiar interface[6] and a reasonable "cost of entry." Although Adobe now sells only an XML version of FrameMaker, there continue to be many SGML documents created in FrameMaker and other tools.

6. FrameMaker was the tool of choice for technical communicators at the time FrameMaker + SGML was introduced.

Single-sourcing

According to the STC *Single-Sourcing* web site, single-sourcing is "using a single document source to generate multiple types of document outputs" or "workflows for creating multiple outputs from a document or database source."

Structured authoring or topic-based authoring

For the purpose of this thesis, the term "structured authoring" will refer the act of writing in some structured manner that results in topic-based documentation. By implication, such writing specifically utilizes a formal methodology such as DITA and some formalized encoding system, such as XML (or pseudo-XML). Where a specific tool, methodology, or system is relevant, it will be specified.

There are countless specific vendor tools, both proprietary and open source, that are available to implement a structured writing project. At its simplest, such a project can be undertaken in plain text, with manually placed cross-references to related topics. However, such a simplistic approach, while accomplishing the structured writing task, surely would not accomplish the goal of doing more with less. Some type of tool or tools that provide a certain degree of automation and verification are necessary to benefit from the underlying technologies.

Tool chain

Tool chain is a common buzz word for folks who use XML, particularly so for the "tools geeks." A *tool chain* is a set of tools used to create a more complex tool or product. Generally, the output of one tool becomes the input for the next. This concept is similar to "piping" available in Unix and, to some extent, the "redirected output" of MS-DOS.

Topic maps

The newer *topic maps* are a subject-based classification technique. Topic maps comprise only three categories: names, occurrences, and associations. While on the surface the fewer categories inherent in topic maps may make them appear to be more simplistic, their strength (and difficulty of application) is that each subject[7] can have multiple *topic mapping name*s and multiple subjects can have the same *topic mapping name*. Furthermore, each topic mapping name can be assigned a limiting scope, without which it assumes unlimited validity. *Topic mapping occurrences* function in topic mapped subjects much like an index in the book paradigm. Topic mapping occurrences use strings or URIs to identify connected information resources. *Topic mapping associations* represent relationships between topic-mapped subjects. Associations such as "used-with," "based-on," "about," and "creator-

7. The "topic" in "topic maps" is used in the general sense of the term, which refers only to the content value rather than the DITA-specific sense, which implies a specific architecture. Hence, topic maps uses "subject" in its nomenclature.

of" facilitate the creation of a network of subjects with clearly-defined associations.

Transforms

Interestingly, *transforms* is used as a noun, meaning the output of a transformation, as well as a verb, meaning the effect of the tool or process. Transformation utilizes the XML companion technology of Extensible Stylesheet Language (XSL). The XLS Transformation (XLST) technology essentially converts (although you'll be lambasted if you use a term other than "transforms") an XML-tagged document to another XML-tagged document, either to change its data model or, more commonly, to publish it in HTML or some other output format. XSL Formatting Object (XSL-FO, pronounced "FŌ") is one of the transforms that XLST produces. The interim "FŌ" file, which contains only formatting information for the XML, is then processed by a print or production engine to produce a PDF. XSL-HTML is essentially the same as XSLT, but performs a transform to HTML output in one step. XSL-FO and XSL-HTML can also be considered stylesheets, as they specify the styles for the XML. Other transformation tools include Ant scripts, XSLT engines such as OrangeVolt, and formatting objects processors ("FOPs") such as Apache. The output files that result from a transformation and that are published in a human-readable format typically include HTML, XHTML, and PDF, but can include transformations to formats for additional publishing or translation, such as into FrameMaker files.

XML. Extensible Markup Language

DITA currently specifies extensible markup language (XML) as its technological methodology. This is important, because XML provides powerful features applicable to the software design community as well as to the technical communication community. This cross functionality of XML results a greater likelihood that companies will be willing to purchase the tools necessary and will have the technical capabilities available to support structured writing projects. Although the XML encoding format may be replaced in a decade or so with a "bigger, better, faster" tool, the basic construct is here to stay.

APPENDIX B

INTERVIEW QUESTIONS

The following list comprises the questions that I prepared to query my research subjects. In composing the list, I endeavored to include any questions I might need to draw out information not provides in the answer to some other question. The result is that each interview included half to two-thirds of these questions. However, no two respondents answered the same and therefore the omitted questions differed for each.

In order to encourage narration, I asked many of these questions in the form "Tell me about how/when/why...." If a question was not relevant, it was skipped. If a respondent's narrative led naturally to a different question, that question was asked instead. Of course, the sequencing of the questions also depended on the respondents' answers and where their narratives led.

1. How old are you?
2. What is your gender?
3. Where do you work?
4. In what industry do you work?
5. What is your job function?
6. How long have you been a technical writer?

7. How long have you done similar work?
8. Do you consider yourself to be a technical person?
9. With what kinds of structured-documentation, XML, single-sourcing, content management or similar non-traditional technical writing projects have you been involved?
10. What kind of product(s) did this/these project(s) document?
11. How many times have you been involved in such a project?
12. Please describe these projects?
13. What was the overall motivation for this project?
14. How long did each project last? Chronologically? In person-months?
15. How many total people were involved in the project?
16. Who were the other people involved in the project?
17. What were their roles?
18. What do you recall about their experiences with the project(s)?
19. In what areas did they have trouble with the project?
20. What was your role before you started on this project?
21. What was your job title when you started?
22. What was your involvement in each project?
23. What was your motivation in being a part of this project? Why did you choose it/agree to do it?
24. Did your job title match your role at the beginning of the project? During the project?
25. Does your job title match what you do now?

26. Did your role(s) change over time? Why and in what way(s)?
27. Has working with these technologies/on this project/etc. been beneficial to your career? In what way?
28. Tell me about how this work has affected your work since? Your job opportunities? Your career path?
29. What technologies did the project utilize?
30. Which technologies did you use personally?
31. What tools did the project utilize?
32. Which tools did you use personally?
33. Did you complete the project(s) in phases or all at once?
34. Tell me about each of these phases.
35. What problems did you expect to encounter at the beginning of the project?
36. And what turned out to be the biggest challenge?
37. What were the most successful parts of the project? In what way?
38. What part of the project was the most critical?
39. Did what you do on a daily basis change from your previous role in technical communication projects?
40. In what way(s) did your daily activities change?
41. In what way(s) did your role change?
42. What technologies and tools did you learn and/or use that you had not previously used?

43. What were the functions of the other people on the team during each of these projects?
44. Did you enjoy your work on the project? Why or why not?
45. What obstacles did you encounter with the tools you used?
46. What obstacles did you encounter with the technologies you utilized?
47. What obstacles did you encounter with the other people involved in the project?
48. What problems did the other people in the project encounter?
49. What do you believe were the biggest challenges for this/these project(s)?
50. Tell me about one of these obstacles.
51. How did the project progress overall?
52. How do you feel about having been a part of this/these project(s)?
53. Tell me about how you would have approached this/these project(s) differently.
54. Tell me about the things that went smoothly with the project(s).
55. Did you have executive support for this project? Tell me about him/her?
56. Did you have an inside evangelist for the project? Tell me about him/her?
57. What kind of academic education do you think is important for someone involved in this type of project? Why?
58. What kind of work experience do you think is important for someone involved in this type of project? Why?

59. What kind of technological expertise do you think is important for someone involved in this type of project? Why?
60. What kind of tools expertise do you think is important for someone involved in this type of project? Why?
61. What are the three most important characteristics for a technical writer?
62. What was your academic background gong into this project? How did this help/hinder you?
63. What was your work experience gong into this project? How did this help/hinder you?
64. What was your technological expertise gong into this project? How did this help/hinder you?
65. What was your tools expertise gong into this project? How did this help/ hinder you?
66. Do you have any friends or colleagues that have done a similar project? How was his/her experience similar to yours? How was it different?
67. How has this project changed you?
68. What would you do differently?
69. What do you think your colleagues would do differently?
70. Did the project meet the goals set out for it?
71. What were the goals for the project?
72. Who determined those goals?

73. Did the team that actually did the work agree with these goals? Did you?

74. Do you consider the project successful overall? Tell me why/why not?

75. What is the key to its success/failure?

76. What would you tell someone else embarking upon a similar project?

REFERENCES

Best, Karl F. “Designing a Structured Authoring System.” Pages 41-46 in SGML/XML '97 Conference Proceedings. SGML/XML '97. “SGML is Alive, Growing, Evolving!” Graphic Communications Association, 1997.

Best, Karl F. “Just How Many DTDs Do You Need?” Pages 131-140 in SGML '96 Conference Proceedings. Celebrating a Decade of SGML. SGML '96 Conference, Boston, MA, Nov 18-21, 1996.

Blyer, Nancy Roundy. “Narrative and Research in Professional Communication.” Journal of Business and Technical Communication 10:3 (Jul 1996): 330-351.

Boje, David M. *Narrative Methods for Organizational & Communication Research.* Thousand Oaks: Sage, 2001.

Boje, David M. “The Antenarrative Cultural Turn in Narrative Studies” Revised Sep 16, 2003 to appear in a forthcoming book edited by Mark Zachry & Charlotte Thralls *The Cultural Turn: Perspectives on Communicative Practices in Workplaces and Professions.* Amityville, NY: Baywood Publishing. Viewed at cbae.nmsu.edu/~dboje/690/papers/Antenarrative_Cultural_Turn.pdf, 2006.

Bosak, Jon. “Text markup and the cost of access,” *Nature*, Aug 2001. Viewed at www.nature.com/nature/debates/e-access/Articles/bosak.html, Feb 2006.

Bourdeau, Beth. “Dual Relationships in Qualitative Research.” *The Qualitative Report* 4.3&4: Mar 2000.

Bray, Tim. “Don’t Invent XML Languages.” Viewed at www.tbray.org/ongoing/When/200x/2006/01/08/No-New-XML-Languages, Feb 2006.

Burnyeat, M. F. “Enthymeme: Aristotle on the Rationality of Rhetoric.” *Essays on Aristotle’s Rhetoric*. Ed. Amelia Olsenberg Rorty. Berkeley: University of California Press, 1996. 88-115.

Caplan, Priscilla. “You Call It Corn, We Call It Syntax-Independent Metadata for Document-Like Objects.” *Public-Access Computer Systems Review* 6/4, 1995.

Carlson, David. "XML Documents Can Fit Object Oriented Applications [Objects & the Web]." Object Magazine 7/9, Nov 1997.

Chase, Susan. "Taking Narrative Seriously: Consequences for Method and Theory in Interview Studies." *Interpreting Experience: The Narrative Study of Lives*. Eds. R. Josselson and A. Lieblich. Thousand Oaks: Sage, 1995.

Coombs, James H., Allen H. Renear, and Steven J. DeRose. *"Markup Systems and the Future of Scholarly Text Processing." Communications of the Association for Computing Machinery* 30/11, 1987. Reprinted in The Digital Word: Text-Based Computing in the Humanities. Eds. George P. Landow and Paul Delaney. Cambridge/London: MIT Press, 1993.

Côté, James E. and Charles G. Levine. *Identity Formation, Agency, and Culture: A Social Psychological Synthesis*. Mahwah, NJ: Erlbaum, 2002.

Cover, Robin, ed. "IBM's Darwin Information Typing Architecture (DITA)." *Cover Pages: Online resource for markup language technologies*. Viewed at xml.coverpages.org/ni2001-03-16-a.html, Feb 2006.

Daniel, Ron Jr. and Linda A. Burman. "Why Metadata Affects Everything and How You Plan a Metadata Implementation." *XML Conference and Exposition*. Orlando: IDEAlliance, 2001.

Dobratz, Susanne. "SGML/XML Overview." *The Guide for Electronic Theses and Dissertations*. Ed. Joseph M. Moxley, www.etdguide.org: UNESCO, 2002.

Garshol, Lars Marius. "Metadata? Thesauri? Taxonomies? Topic Maps! Making sense of it all." *interChange*. 10:3 (Sep 2004): 17-30.

Gergen, Kenneth J. "Technology and the Self: From the Essential to the Sublime." *Constructing the Self in a Mediated World*. Eds. Debra Grodin and Thomas R Lindlof. Thousand Oaks: Sage, 1996.

Geisler, Cheryl *et al*. "Future Directions for Research on the Relationship between Information Technology and Writing." *Journal of Business and Technical Communication* 15:3 (Jul 2001): 269-309.

Giammona, Barbara. "The Future of Technical Communication: How Innovation, Technology, Information Management, and Other Forces Are Shaping the Future of the Profession." *Technical Communication* 51:3 (Aug 2004): 357-368.

Grodin, Debra and Thomas R. Lindlof. *Constructing the Self in a Mediated World*. Thousand Oaks: Sage, 1996.

Gulesian, Marcia. "The Economics of Web Service Development." XML.com, 2004. Viewed at www.xml.com/lpt/a/2004/07/07/econ-ws.html, Feb 2006.

Gurak, Laura J. and Sam J. Racine. "Guest Editor's Introduction: The Social Realms of Technology." *Journal of Business and Technical Communication* 14:3 (Jul 2000): 261-263.

Hackos, JoAnn. *Managing Your Documentation Projects*. New York: John Wiley & Sons, 1994.

Hamilton, Jean Mercedes. "Developing a Business Case for an XML Authoring System." *XML Conference and Exposition*. Orlando: IDEAlliance, 2001.

Hedtke, John. "Why We Don't Get No Respect." *STCAustin.org Blog*. Viewed at stcaustin.blogspot.com/2005/12/professional-development-why-we-dont.html, Dec 15, 2005.

Hertz, Brian. Remarks made at a CTDUG DITA user group meeting. Jan 25, 2006.

Honeycutt, Lee. *What Hath Bakhtin Wrought? Toward a Unified Theory of Literature and Composition*. Charlotte: University of North Carolina, 1994.

Horn, Robert E. "What Kinds of Writing Have a Future?" Stanford University: 2001.

Horn, Robert E. "Structured Writing at Twenty-five." *Performance and Instruction* 32 (Feb 1993): 11-17.

Houser, Alan R. "XML Schemas for Publishing Applications." *XML Conference and Exposition*. Orlando: IDEAlliance, 2001.

Ide, Nancy and Jean Véronis, eds. *The Text Encoding Initiative: Background and Context*. Dordrecht, Netherlands: Kluwer Academic Publishers, 1995.

Karben, Alan. "News You Can Reuse. Content Repurposing at The Wall Street Journal Interactive Edition," *Markup Languages: Theory & Practice* 1:1 MIT Press, 1999.

Korsgaard, M. Audrey and Loriann Roberson. "Procedural Justice in Performance Evaluation: The Role of Instrumental and Non-Instrumental Voice in Performance Appraisal Discussions." *Journal of Management*, Winter, 1995.

Lay, Mary and Laura Gurak, eds. *Research Methods in Technical Communication*. Boston: Allyn and Bacon, 2002.

Lincoln, Y. S. and E. G. Guba. *Naturalistic Inquiry*. Beverly Hills, CA: Sage, 1985.

Lindlof, Thomas R. and Bryan C. Taylor. *Qualitative Communication Research Methods*. Thousand Oaks: Sage, 2002.

MacIntyre, Peter D. "Variables Underlying Willingness to Communicate: A Causal Analysis." *Communication Research Reports* 11:2 135-142.

Mays, Nicholas and Catherine Pope. "Qualitative Research: Rigour and qualitative research." *The General Medical Journal.* 8 Jul 1995. Department of Epidemiology and Public Health, University of Leicester-London. Apr 2006. Viewed at bmj.bmjjournals.com, Feb 2006.

Patel, Tejas and Edward A. Fox. "SGML/XML Overview." *The Guide for Electronic Theses and Dissertations*. Eds. Joseph M. Moxley, Diane Masiello, and Edward A. Fox. 2002. Viewed at www.etdguide.org: UNESCO, Feb 2006.

Rapp, Christof, "Aristotle's Rhetoric." *The Stanford Encyclopedia of Philosophy*. Ed. Edward N. Zalta (Summer 2002). Viewed at plato.stanford.edu/archives/sum2002/entries/aristotle-rhetoric, Mar 2006.

Riel, Margaret. *The Internet and the Humanities: The Human Side of Networking*. Washington D.C.: US Department of Education, 1996.

Ritchie, Jane. "The Applications of Qualitative Research Methods." *Qualitative Research Practice: A Guide for Social Science Students and Researchers*. Ed. Jane Ritchie and Jane Lewis. London: Sage, 2003. 24-46.

Roberts, Tom L. and Paul Benjamin Lowry. "The 'Voice Effect' in Groups." *Proceedings of the Second Annual Workshop on HCI Research in MIS*, Seattle, WA: Dec 12-13, 2003.

Roberts, Tom L., Paul Benjamin Lowry, and Paul D. Sweeney. “An Evaluation of the Impact of Social Presence Through Group Size and the Use of Collaborative Software on Group Member ‘Voice’ in Face-to-Face and Computer Mediated Task Groups.” *IEEE Transactions on Professional Communication* 49.1 (Mar 2006): 28-43.

Schminke, Marshall. “Organizational Structure, Justice Perceptions, and Ethics.” Ethics Resource Center Fellows Meeting. Washington, DC. Jan 2006.

Silker, Christine M. and Laura J. Gurak. “Technical Communication in Cyberspace: Report of a Qualitative Study.” *Technical Communication* (Q4 1996): 357-368.

Silvi, Deborah, Susan Stotzer, and Jamie West. “A Non-Management Career Path for Technical Writers,” Proceedings of the STC 50th Annual Conference, Dallas, TX: May 18-21, 2003.

Snape, Dawn and Liz Spencer. “The Foundations of Qualitative Research.” *Qualitative Research Practice: A Guide for Social Science Students and Researchers*. Eds. Jane Ritchie and Jane Lewis. London: Sage, 2003. 1-23.

Sperberg-McQueen, C. Michael and B. Tommie Usdin. “Welcome to Markup Languages: Theory & Practice.” *Markup Languages: Theory & Practice* 1/1 (Winter 1999).

St. Laurent, Simon. *XML: A Primer.* Foster City, CA: MIS Press/IDG Books, 1998.

St. Laurent, Simon and Robert Biggar. *Inside XML DTDs: Scientific and Technical*. New York, NY: McGraw-Hill, 1999.

STC Single-Sourcing SIG. “Useful Links: Single Sourcing.” Single-Sourcing web site. Viewed at www.stcsig.org/ss, Mar 2006.

Streich, Robert. “Documents Are Software. A Focus on Reuse,” *SGML/XML 1997 Conference Proceedings.* SGML/XML '97. Graphic Communications Association, 1997.

Taft, Darryl K. “MS Researcher to Explore Human Side of Technology.” *eWeek.com*.Dec 19, 2005. Viewed at www.eweek.com, Feb 2006.

Vinoski, Steve. “The Social Side of Services.” *IEEE Internet Computing.* IEEE:2006.

Wiles, Debbie. "Ethical Insights on XML and Single Sourcing." *STC Conference Proceedings 2001.*

Usdin, B. Tommie. "When 'It Doesn't Matter' means 'It Matters.'" *Extreme Markup Languages 2002: Proceedings.* Wyndham Montreal Hotel. Montreal, Quebec. Aug 2002.

Wong, Candy. "Single-Authoring Technique for Building Device-Independent Presentations." *W3C Workshop on Device Independent Authoring Techniques,* 2002. Viewed at www.w3.org/2003/08/Workshops: W3C, Feb 2006.

VITA

Cheri Lou Mullins was born in Terrell, Texas, on October 22, 1957, the fifth child of Alpha Hazel Tate and Francis (Franklin) DeWitt Mullins. After completing studies at Kimball High School, Dallas, Texas, in 1975, she entered the University of Texas at Arlington, where she majored in Systems Analysis and Design and Management Science. She put her formal education on hold to raise a family and work as professional technical communicator in Dallas and Fort Worth. She continued to work as a technical writer, editor, educator, consultant, and manager in Fort Worth, Los Angeles, Silicon Valley, Maryland, and Austin, Texas. In Austin, she transferred to St. Edward's University, changed her major, and completed her undergraduate studies in Technical Communication and Publishing in 1993 and received the degree of Bachelor of Liberal Studies in 1994. She entered a post-baccalaureate program at the University of Dallas in Telecommunications Management in 1994 and completed it in 1996. In the Fall of 2005, she entered the MATC program in the Graduate College of Texas State University-San Marcos.

Permanent Address:	PO Box 200023
	Austin, Texas 78720-0023

This thesis was typed by Cheri Lou Mullins.