THE LANGUAGE OF TOOLS: EXAMINING THE RHETORICAL IMPLICATIONS OF SOFTWARE CHOICE

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

Jeremy Tate English, B.S., B.A.

San Marcos, Texas
May 2012

THE LANGUAGE OF TOOLS: EXAMINING THE RHETORICAL

IMPLICATIONS OF SOFTWARE CHOICE

Committee Members Approved:

_____
Deborah Balzhiser, Chair

_____
Pinfan Zhu

_____
Craig Hanks

Approved:

_____
J. Michael Willoughby
Dean of the Graduate College

FAIR USE AND AUTHOR'S PERMISSION STATEMENT

Fair Use

Duplication Permission

ACKNOWLEDGEMENTS

The author would like to thank his graduate advisor, Deborah Balzhiser, as well as his thesis committee, Pinfan Zhu and Craig Hanks, for their guidance. Additionally, he would like to thank his friend and colleague Mandy Grover for her support. Finally, he would like to thank his ever-supportive partner Rachel Patterson for her patience and understanding.

This manuscript was submitted on April 8, 2012.

TABLE OF CONTENTS

Page

CHAPTER ONE: INTRODUCTION

Background

When I began this project, I found myself faced with many choices. I had to decide how to research the issue I wanted to explore, how to frame and develop my argument, what theoretical framework to use—the rhetorical choices that anyone who has written academically has to make and will be familiar with. However, as not only a graduate student but also as a professional technical communicator, I was faced with the additional decision of what tool to use to write these words: should I draft my text using standard word processing software or in the XML-based structured-authoring system I favor for my professional work? Each technology presented certain conveniences and certain drawbacks.

Even once I decided on a technology—word processing software—I had to decide on a program. Would I use the ubiquitous Microsoft Word, the sleek Apple Pages, or the powerful but complex LibreOffice? Would I prefer familiarity, flash, or control? Being an academic, a professional, and something of an obsessive, these were not decisions I took lightly. While it may seem that such decisions are mere matters of preference or convenience, and indeed often seem to be made based simply on what is available, I had a strong sense that my decision would have an effect on my final product; by affecting even slightly the way that I thought, I felt that it would affect my choice of words and perhaps even my argument. This was not just a choice of which tool was easiest to use or

most convenient, or which technology had more interesting features, but a choice not unlike choosing a tone, a style, or even a research method. My choice, ultimately, was rhetorical.

My goal in this project is to examine the ways in which technical communicators rhetorically relate to their software tools. While a great deal of research has been done on the broad social effects of networked culture and the ways in which digital writing in general differs from traditional composition, my interest is in the specific pieces of software with which technical communicators work on a regular basis. These programs are generally concerned with the digital creation of texts—word processing, image manipulation, presentation, and web authoring. It is important that we more fully examine these tools specifically because they have come to play such an important—and common—role in our day-to-day lives.

The Ubiquity of Software

Ask any computer user—not just a professional technical communicator—what program they are most familiar with for creating text, and the most common answer will be Microsoft Word: a component program in the Microsoft Office suite, which also contains the spreadsheet program Excel, the presentation program PowerPoint, and other productivity programs. North Americans encounter this suite seemingly everywhere: at home, at work, and at school.

Common perception holds that Microsoft Office is by far the most widely-used of the so-called "office" genre of software, but actual numbers about market share are difficult to come by. The most concrete numbers I encountered, presented by Stan Liebowitz and Stephen Margolis in their book *Winners, Losers & Microsoft: Competition*

and Antitrust in High Technology, are over a decade old; nonetheless, they are striking. In 1989, MS Office held zero percent of the market; by 1997 it held over ninety percent while its nearest competitor, WordPerfect, held less than ten (181). A 2006 article in *Bloomberg Businessweek* reinforces this picture of the suite's market dominance, stating that it "has a 95% market share and 400 million copies in use" ("More to Life Than the Office").

The actual numbers, however, are less important than the simple fact that, regardless of hard evidence, people seem to *know* that Microsoft dominates the market. Other office suites exist—Apple's iWork and the open-source LibreOffice being two of them—but even when their virtues are being extolled in the press, they are usually described as alternatives to MS Office. Apple touts compatibility with Microsoft's suite on the main page of its website advertising iWork ("iWork"). This phenomenon is not restricted to office software: the digital image manipulation program created by Adobe is so ubiquitous that "to Photoshop" has entered the popular lexicon as a verb for retouching or altering an image.

Any alternative to a piece of software with such wide adoption as Photoshop or the MS Office suite certainly has a difficult road ahead of it. It must not only address the technical challenge of overcoming an established and fully-featured tool, but it must also overcome the inherent preference of the average consumer to use what is familiar to them. Computers, for many users, are merely tools to accomplish a job; there are those who would no sooner think about the differences between office software suites than they would investigate the relative stress tolerances of ballpoint pens—if it writes, it works, and that is enough. I will argue in this piece that such a manner of thought, which values

only the final discreet product of the tool and not the manner of its creation, is inappropriate for the technical communication field. Our rhetorical background, our positions as experts, and our responsibility to apply critical insight all demand that we look more closely at the software that affects the lives of so many people.

Research

To this end, I sought to examine the leading journals of the field to determine how we interact with, and what we expect from, our software. Specifically, I sought out articles that discussed one or both of two topics: word processing or presentation software tools (most often the Microsoft Office suite) or XML and single-source software tools. There has recently been a great deal of discussion about the introduction of single-source tools, causing some anxiety in the field due to the perceived problems that such a change will bring; this conflict between an old and a new way of doing things represented a remarkable opportunity to examine the community's values.

The ubiquity and dominance of Microsoft Office is being challenged; the fact that this change is, in many ways, being “forced” upon the field makes it an excellent chance to study not only our relationship with this particular suite, but also our attitudes toward software in general. If, for instance, MS Word was simply being replaced by a competing word processing program of similar functionality, there would not have been as much opportunity to examine the deep-seated beliefs and biases present in the community toward their program of choice. The transition would have been based on minor functionality, with the core expectations and beliefs simply transferring to the new program. In defending the so-called traditional ways, the field expresses what had been implicit; in being forced to accept a new paradigm, they articulate precisely what they

valued about the old one. By observing the field in a period of stress, we are better able to see the basic beliefs of its members.

After stating the results of my research and discussing the trends I observed, I will turn to concepts from rhetorical theory to make an argument that we as a field must realign our relationship with our tools. This argument will be based on an understanding of the ability of software to affect our thinking, much like language, and of our unique position as experts of rhetorical theory. I will argue that our decisions regarding software are rhetorical and must be approached with rigor; rather than select our tool because it is cheap, or common, or because it writes and that is good enough, we must select our tool based on a full understanding of its rhetorical implications.

CHAPTER TWO: LITERATURE REVIEW

Overview

My primary research consists of a review of articles published in four leading journals of the technical communication field: *Technical Communication*, *Technical Communication Quarterly*, *The Journal of Technical Writing and Communication*, and *The Journal of Business and Technical Communication*. I selected these journals as a representative cross section of the conversation in the field, with focus ranging from academic issues to workplace issues; further, in an attempt to narrow my scope, I restricted my research to four of the most influential journals. I included the past eleven years of issues in my research, examining all articles published from January 2001 until December 2011.

My motivation in selecting the specific time frame I did was not to determine whether or not we as a field have ever discussed this issue but whether or not we are discussing it now. While interesting conversations concerning the rhetorical nature of software may have occurred in these journals at some point before this time frame, given the rate at which technology moves, I feel that if we have not discussed it recently we might as well not have discussed it at all.

I was specifically looking for the ways in which the field's attitudes toward tools were expressed in the context of the conflict between “traditional” word processing/presentation software and XML/structured authoring, so I focused on those

two topics. I included an article if it discussed political, rhetorical, or organizational implications of one of the two technologies or if it discussed practical advice for the use of a particular tool that implemented one of the technologies. While many articles may tangentially mention the effect that digital writing has on the author or the organization, I focused on those articles that explicitly discussed either the tools or the theories behind the technology.

By these criteria, I identified forty-one articles that discussed either word processing/presentation or XML/single-source. Representation was roughly equal between the two categories, with twenty-one articles about word processing/presentation software represented and twenty about XML/single-source represented. Organizing the articles by technology allowed me to more easily identify trends within each group and contrast the attitudes toward one tool with the attitudes toward the other. These trends exist across journals and throughout the time period represented. In the sections that follow, I will discuss the trends identified in those articles in the word processing/presentation category and the trends in XML/single-source articles.

Trends in Word Processing/Presentation Software Articles

Discussion of word processing and presentation software tools tends to emphasize practical advice, discussing methods for manipulating software tools to achieve specific effects either in the resulting document or in the immediate business/cultural context. Compared to the articles that discuss XML/single-source, there is less discussion of underlying theory or effects on the writing process itself. I identified two key trends in these articles: *using tools to achieve an end* and *defending the tool*.

Using Tools to Achieve an End

Examples of word processing or presentation software—primarily Microsoft Word or Microsoft PowerPoint—are usually discussed in terms of how best to manipulate them. They are viewed as a means to an end; the "end" can be either a discreet document/presentation or a measurable social/political effect in the workplace. Any rhetorical discussion focuses on these textual or political end products—not the software tool itself.

In "Word-Processing 'Efficiency'——By Means of Personalized Word-Frequency Lists," David Coniam outlines a method for taking advantage of the *Autotext* feature in Microsoft Word to increase typing efficiency. Directly speaking to academics in university situations, Coniam argues that users can become more efficient by creating keyboard shortcuts for frequently-typed words. He describes how to do so with one of Word's "many preset hot key combinations, many of which are generic text manipulation features" (177). For Coniam, the desired goal is more efficient production of documents —a goal met simply by effective use of the tool. In terms of rhetorical focus, there is little in the article to suggest anything more than a purely instrumental relationship with the technology; it focuses only on mechanical typing efficiency and could be described as "arhetorical."

A slightly more rhetorically-focused viewpoint is espoused by Eva R. Brumberger in "The Rhetoric of Typography: The Persona of Typeface and Text," and by Jo Mackiewicz in two articles: "What Technical Writing Students Should Know About Typeface Personality" and "Audience Perceptions of Fonts in Projected PowerPoint Text Slides." Brumberger, arguing for the importance of visual rhetoric in document design, points out that word processing tools "offer ready-made design templates, which often

appear to have been created without any understanding of the principles of document design" (206). Mackiewicz echoes this line of thinking and carries it into a similar study of the visual effectiveness of fonts in PowerPoint presentations ("Audience Perceptions"). For Brumberger and Mackiewicz, the desired goal is a document that most efficiently conveys information to an audience without the shape of the text itself getting in the way; this is achieved by properly selecting the text styles available in each respective tool. Both authors acknowledge the capabilities of text layout and design to affect a reader, but they engage with it at the level of the final product—the document or the slide presentation—discussing the tool only as it is used to create that final product.

Mackiewicz is not the only author to speak about the use of slide presentation software. I initially intended to focus solely on word processing software for this project, but in the course of my research I repeatedly came across references to software of this type—almost always referencing a single program: Microsoft PowerPoint. The frequency of these types of articles convinced me to expand my initial research category to include this type of software as well. Expanding my scope in this manner led to a wealth of material to examine: discussions about the effective use of PowerPoint occupied more than half of the articles I identified in this category.

Authors discussed the use of graphical layout in slides (Mackiewicz), the most rhetorically effective way to write slide headlines (Alley and Neely; Alley, Shreiber, Ramsdell, and Muffo), the proper outline structure for content (Farkas), and the best way to craft handouts from slides (Lee). Even more explicitly than with word processing, authors expressed tremendous interest in the best ways to manipulate PowerPoint to achieve an end product that conveys information clearly and is aesthetically pleasing. Just

as with Mackiewicz and Brumberger, these authors acknowledged certain rhetorical effects that different uses of a tool could convey to an audience on a textual level; however these rhetorical effects are discussed as they relate to the final products and not the tool used to create them.

Some authors more directly addressed the rhetorical effects of software tools, as Clinton Lanier did in two articles concerning the implementation of e-editing at a government lab ("Creating Editorial Authority", "Electronic Editing and the Author"). "Technological innovation, essentially the dissemination of new technology within an organization," Lanier states, "restructures the organization in such a way as to create *new* structures, including power structures" ("Creating Editorial Authority," 471). He goes on to describe how this dynamic can be used to the advantage of the editors in the organization, who historically have suffered from what he describes as a "low-status" position: "e-editing should be closely examined as a mechanism for changing that status and for acquiring at least a portion of authority" (478). For Lanier, the desired goal is an improvement in the immediate working conditions of writers; he advocates a method of achieving this based on taking advantage of the features of a software tool for political, not textual, effect. While this represents a much higher-level discussion of the effects that text can have not only on audience understanding but also on the creation and maintenance of power, Lanier's discussion still relegates the tool to a secondary role—it facilitates the political effects, but his rhetorical discussion lies with the political effects themselves.

This tendency to discuss tools as means to an end, whether that end is textual or political, suggests two things. First, there is a level of comfort with the software tools in

question. The articles seem to assume that the basic operation of these tools is well known to the audience; freed from describing the basic functionality, the authors are able to instead focus on refined methods of operation and interesting applications of the tool. The tools of word processing and presentation software seem to have been internalized by the field: they are standard, accepted, and in some cases even mundane. Second, this tendency suggests that while software tools are recognized as being instrumental in achieving rhetorical effects in end documents, they are not directly discussed as rhetorical objects in and of themselves; in a sense, these authors discuss software similarly to the "windowpane" concept of language: its should clearly and unobtrusively convey the purpose of the author.

Defending the Tool

The significant number of articles concerning Microsoft PowerPoint in this category contributes to the second trend I identified: defending this single program's usefulness or suggesting modifications to correct its perceived flaws. While the articles concerning word processing tended to use Microsoft Word as their example but spoke in slightly more general terms, these articles are striking in that they much more specifically speak to a single program.

This defense of PowerPoint all seems to stem from the publication of Edward Tufte's article "The Cognitive Style of PowerPoint: Pitching Out Corrupts Within," which issued a direct challenge to the underlying workflow encouraged by the default settings in PowerPoint. Tufte's essay argues that this default structure—an emphasis on bullet points over fully elaborated sentences and a forced hierarchical structure on data that is best presented in parallel—encourages an oversimplification of ideas and a presentation style

that is counterproductive to an audience's understanding of an issue. He argues that it "allows speakers to pretend that they are giving a real talk, and audiences to pretend that they are listening" and declares it a "prankish conspiracy against evidence and thought" (185). Each of the articles I identified in this category presents a response to Tufte's argument. Some authors disagree with his basic premise, that PowerPoint is by its nature counterproductive, and attempt to show that PowerPoint is useful for conveying information. Others seem to begrudgingly accept his conclusions about its default settings but reject his argument that the program is beyond repair.

At the most passive end of the spectrum, Nicole Amare's "To Slideware or Not to Slideware: Students' Experiences With PowerPoint vs. Lecture" presents survey data about which method of instruction—slide- or lecture-based—is more highly regarded by college students. While Amare states that she personally does "not agree with Tufte that PowerPoint is making us and our students stupid," she admits that her study "reveals unpredicted results" (305). Students' enthusiasm for the more dynamic style of PowerPoint did not lead to the increased data retention over standard presentations that she expected to see (305), though she says the data is not conclusive and "larger studies are needed to assess the link between PowerPoint, cognition, preferences, and performance" (306).

Jo Mackiewicz undertakes a similar study in "Comparing PowerPoint Experts' and University Students' Opinions about PowerPoint Presentations," in which she seeks to determine how effective the advice of experts on slide authoring is when tested against student expectations. She found that the two groups largely agree on what works and what does not, but that they differ on specific details such as the use of animation.

Mackiewicz's article, more so than Amare's, represents an attempt at coming to a theory of effective PowerPoint use based on usability studies and research; both studies, however, proceed from a skepticism about Tufte's basic premise: that PowerPoint is inherently counterproductive to the conveyance of information.

Responses to Tufte that seem to more begrudgingly agree with his basic point, but not with his concept that PowerPoint is beyond help, are presented by authors that advocate for specific composition practices. Michael Alley and Kathryn A. Neely, in "Rethinking the Design of Presentation Slides: A Case for Sentence Headlines and Visual Evidence," argue for a more thoughtful design of slide headlines to aid in information retention. This includes the use of full sentences as slide headlines instead of the staccato topic chunks encouraged by the tool's default settings. Alley and Neely identify this as *intelligent use*: "The advocates of intelligent use seek to maximize the potential advantages of projected slides while also calling attention to the need for thoughtful design of slides and presentations" (419). They admit, however, that their strategy "requires much effort to overcome the defaults of presentation slide programs, such as PowerPoint" (423); intelligent use works in spite of, not in concert with, the default settings of the tool. Alley, along with co-authors Schreiber, Ramsdell, and Muffo, presents an experimental study describing the effectiveness of his and Neely's recommendation in "How the Design of Headlines in Presentation Slides Affects Audience Retention." This study gives evidence that overriding the default structure of PowerPoint in favor of the recommendations made in the earlier article leads to a measurable increase in information retention. In both of these studies, the rhetorical effect of the tool is discussed in as much as it aids information transmission, but only in terms

of the final product—the slide presentation.

David Farkas, in "Managing Three Mediation Effects that Influence PowerPoint Deck Authoring," identifies certain ways in which the default standards of PowerPoint tend to affect the composition process: the tendency for authors to cut content based on slide layout restrictions, the tendency for authors to distort hierarchical organization by splitting content over slides, and the tendency to underplay important distinctions due to the uniform nature of slide titles. In this way, he also agrees with the basic substance of Tufte's criticism but is unwilling to go as far as Tufte does in attacking the program itself; he opens the article with an explanation that, "given the enormous prevalence of PowerPoint in business, government, education, and other areas of life," it is important to "identify and address" its flaws (28). After offering methods for overcoming these effects, Farkas concludes that "by carefully identifying and investigating PowerPoint's many mediation effects, we can better understand PowerPoint and promote better PowerPoint use" (37). It may be broken, but we should not throw it out yet.

Notably, there were no articles in any of the journals that I reviewed that did *not* take issue with some aspect of Tufte's argument in "The Cognitive Style of PowerPoint." Several of them expressed an agreement with some of his basic ideas, but ultimately each of them either defended PowerPoint outright or at least offered suggestions for improving it. No articles were undertaken with the purpose of reinforcing Tufte's view, and none offered an alternative to PowerPoint.

Similar to the articles identified in the trend of viewing tools as a means to an end, these articles focused on the effective manipulation of software tools in order to achieve desired rhetorical and practical products. However, the distinction between discussing the

products of a tool versus discussing the tool itself is clear: faced with a criticism of the underlying assumptions behind a tool, the authors responded by devising more complex and elaborate ways to use that tool. They were unwilling to discuss the tool as a rhetorical object in and of itself, remaining faithful to the "windowpane" concept: Tufte's criticism is only valid if you do not use the tool correctly, and the field's discomfort with this criticism is evident in the explosion of response it garnered.

Each of the trends identified so far reinforce two particular viewpoints: that the measure of success for a software tool is in its usefulness at creating a textual product or achieving a political end in the workplace, and that the solution to cognitive or rhetorical problems with the operation of a tool—provided you agree that these problems exist—is simply new and better ways of using the tool. Authors engage in rhetorical discussion, but only with the final products of their tools and not with the tools themselves; in fact, faced with a rhetorical challenge to a tool, they responded by attempting to draw focus back to the products.

Trends in XML/Single-Source Software Articles

XML stands for "extensible markup language." It is a standard that defines a basic method for marking text with its purpose or intended use instead of specific formatting instructions. For instance, instead of writing a book synopsis and formatting it to appear in italics, an author would write the text and mark it as a synopsis; the formatting of that section would vary based on the way in which the text is meant to be used. The same underlying text can be used to create multiple kinds of documents; a printed document and a webpage, for instance, could be created at the same time, from the same text, and feature completely different formatting. This is called "single-source" authoring—the

"source" text is written once and processed multiple times according to its intended use. Text is identified by what it is structurally instead of how it should look aesthetically. The act of writing and the act of formatting are separated.

In contrast to the articles concerning word processing and presentation technology, the articles I identified as being about XML/single-source technology exhibited less of an emphasis on specific usage and more of an emphasis on the cognitive, political, and social effects of adopting the technology. While it was common to find articles on word processing and presentation that advocated for certain "tips and tricks," such as using the Autotext feature in Word (Coniam, "Word Processing 'Efficiency'") or using the Slide Sorter view in PowerPoint (Mackiewicz, "PowerPoint Presentations"), the articles in this category dealt less with workflow recommendations and more with overall explanations of the technology and examinations of its effects on the workplace and on the author. I identified three trends in these articles: *explaining the technology*, *anticipating changes in the workplace*, and *describing changes in the writing process*.

Explaining the Technology

One of the most common trends in the XML/single-source articles was an explanation of the basic underpinnings of the technology. As it requires authors to think of their texts in a different way than the traditional word processing and presentation tools with which the field is familiar and comfortable, authors devoted considerable time to explaining exactly how it works—and at a significant level of technical detail. It is not enough for these authors to discuss the technology in general terms, as I have done; detailed explanation of the files, workflow, and even markup syntax appears to be the

priority of these authors. Audiences are being taught the new technology much in the same way that one would be taught a new language; instead of assuming basic competence and focusing on expanding vocabulary, as was the case with traditional word processing tools, these authors start with the alphabet. In this sense the discussion is not exactly rhetorical, but grammatical.

Writing in 2001 about the then-new XML standard, Lars Johnsen spends the entirety of "Document (re)Presentation: Object-orientation, Visual Language, and XML" presenting examples of text written in XML, describing the process of creating style sheets for rendering the text, and presenting examples of identical pieces of text rendered in different ways. In fact, most articles written in the early years of my research period present detailed descriptions of the technology. These include Fillip Sapienza's "Does Being Technical Matter?" (2002), John T. Battalio's "Extensible Markup Language" (2002), and J.D. Applen's "Technical Communication, Knowledge Management, and XML" (2002). In each of these early articles, the authors go so far as to provide samples of text written in Extensible Markup Language, provide charts describing workflow processes, and generally orient the audience to the technical underpinnings of the changes they are discussing.

However, the trend of describing the technology continues past these early years well into the second half of the decade. It is evidenced as late as 2008 in Charlotte Robidoux's "Rhetorically Structured Content," which does not go to the lengths that the other articles do but still includes a basic explanation of the technology and an example of what XML text looks like (114). This trend suggests a preoccupation on the part of the authors, and by extension the journals themselves, with understanding and coming to

terms with a new technology. Rather than focus on any specific tool that implements XML or single-source technology, they devote themselves to describing the fundamental methods by which the technology works even years after its introduction. There is a definite discomfort with the technology that stands in direct contrast to the familiarity with which the field views word processing and presentation—XML/single-source is seen as strange and new, and it is received warily. This trend does not describe a rhetorical relationship with a tool so much as grammatical one; however rhetorical issues are expressed in the other next two trends I discovered.

Anticipating Changes in the Workplace

The transition to single-source authoring methods is widely expected by scholars to lead to political and social changes for technical communication professionals; identifying what these changes will look like and addressing them is the second trend that becomes apparent when reviewing the articles in this category.

In “The Impact of Single Sourcing and Technology,” Ann Rockley presents one of the earliest represented examples of this tendency to look forward and describe the changing role of the technical communicator. She identifies the many different roles that she believes technical communicators will play in a single-source environment, describing the changing responsibilities of each. Speaking about writers for instance, she says that they “will become more proficient communicators and rely less on the tools that are used to display the final information”; about editors she says, “many organizations have reduced or eliminated the role of the editor. However, single-sourcing makes this role an important one to ensure that information can be reused effectively” (192-193). Writing in 2001, Rockley's wide variety of new roles and responsibilities highlight an

underlying belief that the field is in for significant changes. Battalio, in “Extensible Markup Language,” presents another example of this tendency to look forward and project the changes that are coming: “The new process will no longer be a self-contained procedure. Instead, software documenters must work closely with a wide range of specialists from knowledge managers and information developers to information technologists and computer programmers” (235). The relationship between writer and tool expressed in these articles is different from that more commonly found in word processing/single-source articles; instead of dealing with the “low-level” rhetorical aspects of a tool's textual output, it deals with the greater rhetorical questions of a writer's relationship with his or her audience.

There are a number of articles that continue in this manner, attempting to determine the new role of the technical communicator, but there are many different perspectives on the issue. Article topics range from speculation on the ability of technical communicators to create new environments of collaboration (Applen, “Technical Communication”), to opinions on single-sourcing's effects on the teaching of technical communication (Eble, “Content vs. Product”), to studies attempting to determine the degree to which single-sourcing has been adopted by the community (Dayton and Hopper, “Single Sourcing and Content Management”). In each case, a rhetorical relationship with a tool is expressed, but purely in terms of how it will affect a writer's environment. More precisely, the important relationship seems not to be with the tool itself, but with the new reality that tool will create.

Locke Carter, in the introduction to the special issue that *Technical Communication* devoted to the subject (volume 50, issue 3, 2003), directly addresses the

reason that this new reality demands discussion: “One might reasonably demand to know why this topic needs a special issue of *Technical Communication*. After all, I do not recall a special issue on the impact of 'cut-and-paste,' 'drag-and-drop,' or other hot topics of the past” (317). Carter argues that many in the field see the technology as especially disruptive and threatening: “When production techniques are changed radically, workers often perceive this change as a threat to their jobs and their way of doing things” (318). This trend, then, suggests a tendency on the part of the academic and business community to attempt to divine the ways in which their professional world will change. While it is acknowledged that the new tools will bring about this change, and the mechanics of how it could happen are discussed, there is still a sense that the tool is merely a means to an end. We discuss the reality it will create, not necessarily why it is able to create that reality.

Describing Changes in the Writing Process

The third trend that is visible in the XML/single-source category seems to be a natural extension of the previous one: a specific interest in the ways that structured, topic-based composition will affect the rhetorical aspects of technical communication. In addition to being concerned with how the workplace will change, authors are interested in how the writing process itself will change.

The general concern in these articles with XML/single-source technology is its perceived tendency to strongly restrict authors, firmly imposing arbitrary limitations on style and segmenting the writing task. As Sapienza puts it, “Given the predominantly restrictive rules of logic that guide computer programming and hypertext markup, it would appear that XML schema would exemplify a formalistic genre process” (“Does

Being Technical Matter," 162). Forecasting changes in the teaching of technical communication, Michelle Eble states that "Single sourcing is more than a complex software package or XML tags; it is a way of thinking, a reconceptualization of the relationship between audiences, purposes, and documents" ("Content vs. Product," 345). The message is persistent and clear: writers must accept and deal with fundamental changes to the processes of composition and presentation.

Of the articles included in this review, the most sustained treatment of this topic is found in Stewart Whittemore's "Metadata and Memory: Lessons from the Canon of *Memoria* for the Design of Content Management Systems." Whittemore acknowledges the confusion and dread that accompanies such a drastic shift in composition style: "while the demise of the unified writer might open interesting new avenues for researchers, for practicing technical communicators tasked with composing in a content database, it can engender a sense of powerlessness and purposelessness" (90). He offers a solution, however, that should appeal strongly to the classically-trained writer: look to the past.

Relying on what he calls the "Ciceronian and Quintilianic traditions" (95), Whittemore argues that the practice of breaking a composition into constituent parts and re-ordering it based on necessity is not much different than the methods advocated by the ancient masters of rhetoric: "the orator would commit her speech to memory by breaking it down into constituent parts of varying lengths and then associating these parts with mnemonic images" (96). He advocates a similar method for organizing content management systems: they "should indicate both visual and spatial relationships" and "augment file folder browsing with methods for finding content that leverage human

visual-spatial sense-making capabilities" (101). Finally, writers must "be given larger design views of their texts-in-progress so that they can keep track of their larger discursive goals" (106), based on Quintilian's advice to visualize the text on a physical space like a page in order to aid in composition (104). Whittemore is not the only author to speak about this paradigm shift, but he is the author who does so most systematically and with the strongest emphasis on classic rhetoric.

Of the trends I identified in these articles, this one most directly deals with rhetorical issues and addresses a rhetorical relationship. This is a discussion of the very ways in which the tool itself will affect the way we write. However, the weight of the discussion still is on how a writer can deal with this change, as if the change is both a given and an unavoidable product of the tool. Even if such a view is warranted—that there is no way to escape this tool changing our writing process—it is not explained. As such, we discuss what to do about our writing process changing without actually discussing why that change will happen.

These three trends: continually describing the technology, anticipating the changes that will come about in professional settings, and discussing adjustments that must be made to the rhetorical process, create the impression that the field is highly anxious about these developments. Authors and readers seek to get all the information they can on XML/single-source technology because, unlike word processing or presentation tools, it worries them. It promises not only to restructure their professional lives, but most importantly, the very way in which they compose. They acknowledge that, at least on some level, their relationship with their tools is rhetorical; however, they still avoid talking about the tools as rhetorical objects directly even though doing so may be

beneficial in many cases.

Key Differences Between the Article Categories

In the previous section, I identified trends within the two article categories: word processing/presentation technology and XML/single-source technology. Now, I will turn my attention to the key differences that can be drawn between the two categories based on these trends. I have identified two differences in the articles that bear discussion: the authors' *attitude toward tools* and their *attitude toward power*.

Attitude Toward Tools

There is a strong distinction between the two article categories with respect to their attitude toward software tools. The key difference seems to be that, while articles in the XML/single-source category rarely if ever mention by name any tools used for writing or maintaining single-source documents and instead speak in generalities about the affordances of the underlying technology, the articles in the word processing and presentation category speak almost exclusively in terms of specific tools. Furthermore, there is a remarkable homogeneity evinced in their discussion of these tools; in almost all cases, the Microsoft Office suite is spoken of exclusively or used as a stand-in for all programs of the genre.

Jo Mackiewicz, for instance, uses Microsoft Word as a stand-in for all word processing programs in her article about the rhetorical effects of typeface, citing “the increased graphics and design capabilities of word processing software--like Microsoft Word” (“Typeface Personality,” 115). David Coniam spends an entire article suggesting the ways that writers can be more rhetorically effective by using a software option available in Word (“Word-Processing ‘Efficiency’”). Throughout the journals, either

when speaking directly about the genre or casually mentioning it in the course of an article, for all intents and purposes Microsoft Word is word processing.

When discussion shifts to presentation software, Microsoft PowerPoint is even more directly used as a representative. For instance, in Amare's "To Slideware or Not To Slideware," PowerPoint stands in for all presentation software; its flaws are used as shorthand for the flaws of the entire genre: "*PowerPoint* is accused of encouraging us to 'think in bullets,' a cognitive activity that may affect our quality of ideas, not just our presentation style" (298). Mackiewicz echoes this generalization in "Opinions About PowerPoint Presentations," an examination of a single program that does not explicitly claim to represent the entire genre but makes no effort to dissuade the idea.

Even in articles that are not primarily about word processing or presentation, tools are casually spoken of as "given". When Battalio explains how different an XML workflow is, he explains that "employees would not simply open a Word document and start typing a letter to a client" ("Extensible Markup Language," 217). Joe Williams, echoing Ann Rockley and describing the increasingly sophisticated levels of single-sourcing, states that some level of sophistication is possible "even using FrameMaker or Word" ("Implications of Single Sourcing," 321). Ultimately, in all of these instances, it is implied that Word and PowerPoint have been internalized to the degree that they are used as generic terms. We seem in these cases not to distinguish between our relationship with a tool and with the genre it represents.

The only significant discussion of XML/single-source tools, however, is found in an article that is critical of their business practices: "The Rhetoric of Enterprise Content Management (ECM): Confronting the Assumptions Driving ECM Adoption and

Transforming Technical Communication" by Rebekka Andersen. In it, Andersen systematically defines and challenges the assumptions made by the sales strategies of vendors selling content management software: "technical communicators cannot afford to remain recipients of ECM technologies, allowing IT departments, software vendors, and management to make critical communication decisions" (81). That there could be enough competition among vendors to inspire such an article suggests that no one of them has yet taken hold of the market. The ambivalence toward tools expressed in other articles reinforces this; unlike word processing, there is no ubiquitous equivalent to Word. We are not so comfortable with this technology to have internalized it to the degree that we have word processing or presentation. Our wariness about the new technology may be the result of a realization, to a small degree, of its rhetorical importance; our acceptance of the previous tools may come from a failure to realize their rhetorical importance.

Attitude Toward Power Creation

The authors of the articles in the XML/single-source category seem well aware of the concept that technology affects power dynamics and reflects specific values. As Jason Swarts states, "Technologies are purposeful; they are designed by someone, and being designed, they embody the values and actions of the designers" ("Information Technologies," 304). This topic is addressed in both categories, but with strikingly different attitudes. In both cases, the creation or restriction of power is discussed as an end—another product similar to a printed text or a presentation—but the articles concerning XML/single-source express considerable anxiety about the creation of power while the articles about word processing/presentation software actively campaign for it.

The primary example of power dynamics being discussed in terms of word

processing/presentation is in Lanier's articles on using MS Word to obtain power in an organization. As stated, this is a description of a rhetorical relationship, but one that does not explore the tool as a medium of power creation. The tool is interchangeable, and would be discussed the same way no matter if it were a computer program or a magic spell: it is only rhetorical in that it is useful in facilitating a new dynamic.

Lanier's optimism for the re-casting of power roles in an organization makes an interesting contrast to the attitudes expressed in the articles concerning XML/single-source technologies. Carter's introduction to the special issue of *Technical Communication* dealing with the new technology relays a story in which a technical communicator stated that, if single-sourcing was implemented at his workplace, the writers would all “walk” (317). In a single-source environment, it is believed, the individual writer will need to surrender a degree of power to those who organize and control the information databases. Communicators, the articles tell us, will need to learn to master this new system and affect its development if they wish to keep their power.

In both cases, the capability of the tool to create or restrict power is approached somewhat uncritically. Authors seem to accept that software will give power to someone or take it away from someone; the trick is not to question this process or engage with it, but to find a way to be on the beneficial end of it. There is a rhetorical relationship expressed here, but one that seemingly ignores one of the most important elements in the process: the software tool itself.

Observations on the Literature

Having identified the trends that run through each set of articles, and the key differences between them, I will now offer three observations on the body of texts as a

whole.

Status Quo and Impending Chaos

Word processing tools in general, and the Microsoft Office suite in particular, are widely seen in the journals as a safe status quo. Familiarity with the basic operation of these programs is assumed; it is unnecessary to explain how they work to the reader. Instead, the authors are free to suggest new and innovative uses for the tools—much in the way that one would suggest new chord structures to a musician who already knows how a guitar works. Many articles, such as Whittemore's "Metadata and Memory," even refer to word processing and presentation tools as "traditional." These articles suggest that practitioners are comfortable with these technologies and have adopted them into their everyday life and work routines.

Furthermore, this status quo is important enough to be vigorously defended, as is the case when Tufte published his scathing criticism of Microsoft PowerPoint. The industry's reaction was swift and loud, immediately publishing many articles that sought to defend or rehabilitate PowerPoint. It was not that PowerPoint "made us stupid"—we just weren't using it right, they told us. If we only learn to manipulate the tool in better ways, our problems will be solved and we will not have to surrender it. That a well-designed tool should not need to be defended is never even addressed.

The greatest challenge to the status quo of what Whittemore called "traditional" programs, however, does not come from the criticisms of Tufte but from the advent of a new set of tools: XML and single-source. The community initially reacted with intense curiosity and anticipation; the early articles describing the technology are full of technical detail and lengthy explanation. Once the full details of the situation became clear, the

community began focusing on ways to adapt to the change and to stay ahead of what they felt, as Carter identified, was a threat to their livelihood. The same community that accepted advice to use Microsoft Word to wrest power back from the engineers in an organization (Lanier, "Creating Editorial Authority") suddenly began warning each other of the dangers of letting software and software vendors take power away from them (Andersen, "The Rhetoric of Enterprise Content Management").

My research into the conversation in the academic journals suggests that on one hand the field understands the way that software tools are social, political, and rhetorical devices—however only insofar as their final products have these qualities. Further, they are happy to exploit the rhetorical power of these objects when advocating using them to their advantage. On the other hand my research suggests that the field only discusses the negative aspects of this power when it appears to threaten their jobs.

Tools in the Background

It is obvious from my research that the articles in each category have different attitudes toward software tools: those discussing word processing and presentation deal almost entirely with the specifics of tool use while those that discuss XML and single-source barely mention the tools at all. However, the superficial difference between these two viewpoints masks an underlying similarity—that in the technical communication journals, software tools are unworthy of discussion as rhetorical objects in and of themselves. Tools are either means to an end, or they are unimportant altogether.

For the word processing and presentation articles, this tendency is best expressed by the articles that defend PowerPoint. Tufte's criticism is remarkable in that it is a direct rhetorical challenge to the underlying values of a program; it is an attack not on the final

slide presentation “product,” but on the tool itself and the ways of thinking that it engenders. The community's response is telling in that rather than engaging Tufte on his own terms, by arguing that PowerPoint's default settings do not in fact encourage irresponsible presentation of data or by laying out the underpinnings for a better slide presentation tool, they opt instead to insist on different and more elaborate ways of using the tool. Tufte points out a flaw in the foundation, and the community repaints the house.

This reluctance to engage in a direct rhetorical discussion of the tool itself and instead focus on its products is addressed by Michael Knievel in “Technology Artifacts, Instrumentalism, and the *Humanist Manifestos.*” He argues that it is a result of the field's humanist background and historical alignment with the academics of English departments. “Technology in its material form,” he says, “is indirectly addressed, tangentially included—implied rather than centrally located and directly treated” (72). The field's discomfort with addressing their tools head-on is displayed in this concerted effort to change the terms of the debate: the tool is not important, only the final product. This attitude is taken to its logical conclusion in the XML/single-source articles, as the tools are usually not even mentioned.

Knievel, citing W.J. Ong, further suggests that “many humanists have developed a sort of dualism, constructing a false dichotomy between the realm of the physical and that of the mind” (78). Because of this dualism, he claims, they have drawn boundaries between new technologies and those that have become familiar: books and pens are technological artifacts and they are used every day, but because they are familiar and normal they are not thought of as technology (78). The community's failure to engage in discussion concerning the rhetorical shortcomings of PowerPoint may be a consequence

of this mindset: because PowerPoint and Word are familiar and normal, they are not thought of as tools in quite the same way, and therefore an attack on their underpinnings is taken instead as an attack on their final products.

Acceptable and Unacceptable Creations of Power

As is made clear by the observed trends in the articles, the authors are well aware of the ability of software and technology to disrupt existing power structures and create new ones. My observation about the articles, however, is that the community is only critical of this ability when it manifests in the specific environment of their workplace, and then usually only when it works against them.

This is apparent in the difference in attitudes between the article categories when discussing the power-creation process. The articles concerning XML/single-source express a definite uneasiness with the changes that are to be brought about by a shift in production and authoring techniques, even if this uneasiness is couched in a language of reassurance. To remind practitioners that they will still have a place in the new world is an acknowledgement that many believe they will not; rarely will a field spend so much time talking about the future if it is not worried about that future. By comparison, while it is acknowledged that word processing and presentation tools can be disruptive to existing power structures, the only discussion this warrants concerns ways in which to use this to the advantage of writers and editors. There is no discussion of how such technologies could possibly disenfranchise technical communicators.

Word processing and presentation tools may have been seen as being extremely disruptive at their introduction, but my research only covers the past ten years. It is outside the scope of this project to discuss the attitudes expressed by the community

twenty to thirty years ago. Whatever trepidation met the older technology at its adoption is irrelevant, ultimately, in the face of its widespread and apparently unquestioned acceptance today. The power afforded to the community by these tools has long been accepted, and is seen as beneficial. The tools may no longer *create* power structures, but they *reinforce* them every day; the community is not concerned because the reinforcement works in its favor and we are so used to this reinforcement that we barely notice it—if we notice it at all.

Thus, the community considers certain actions of power creation to be acceptable and others to be unacceptable (or at least troubling). The power created and reinforced by word processing and presentation software is acceptable, and therefore receives significantly less attention in the journals than does the problematic power creation of XML/single-source. In terms of its rhetorical relationship with software tools, the community seems unwilling to engage with its tools directly as rhetorical objects. We discuss the rhetorical effects of the products we create with our tools, but not the tools themselves. As I will argue in following chapters, this inequality of discussion in the literature allows other important issues to go unnoticed and undiscussed.

CHAPTER THREE: THE IDEOGRAPHIC STRUGGLE

Ideographs

In my research, I identified a conflict in the journals between the familiar technologies of word processing and presentation tools on one hand and the new XML/single-source tools on the other; one is seen as safe and traditional, the other as dangerous and threatening. I argue that this conflict is evidence for the fact that software plays a specific rhetorical role in the world of technical communication and in society at large. To do so, I turn to Michael Calvin McGee's concept of the "ideograph," which he outlined in a 1980 article in the *Quarterly Journal of Speech*. While specifically meant to apply to terms of language used in political discourse, McGee's definitions are nevertheless quite illuminating when applied to the world of software.

In "The Ideograph: A Link between Rhetoric and Ideology," McGee begins by explaining that many theorists, such as Burke, Dewey, Mead, and Lippmann, have attempted to distance themselves from the idea of "ideology" as an explanation for "public" or "mass consciousness." Presented with the "brute, undeniable phenomenon" that "Human beings in collectivity behave and think differently than human beings in isolation" (425), these theorists prefer to work from a framework that sees this collective behavior as a mass acceptance of a myth—a collective decision to agree to a fiction.

Different philosophical camps, however, define the this collective decision differently. Marxist authors, according to McGee, tend to characterize this acceptance as

the public being tricked by those in power; authors in the vein of Kenneth Burke reject a political explanation and instead focus on a “philosophy of myth,” emphasizing the reasons that members of a society subscribe to these concepts (425-426). The conflict between these two viewpoints, according to McGee, obscures a more useful way of thinking about the situation.

He counters that the two concepts - “myth” and “ideology,” both contain elements of truth and that the “political” and “mythological” arguments should not be viewed as alternatives to one another. They should be viewed as complimentary: “The Marxian asks how the 'givens' of a human environment impinge on the development of political consciousness; the symbolist asks how the human symbol-using, reality-creating potential impinges on material reality” (426). In short, the truth of the situation draws equally from both sides, and McGee offers a vocabulary with which to discuss this middle ground.

Arguing that political power and persuasion are at their core rhetorical, McGee states that “ideology in practice is a political language” (427) and that the proper heuristic for understanding the motivations of “public” thought are the very terms that society uses to discuss its social values and decisions: “a vocabulary of 'ideographs' easily mistaken for the technical terminology of political philosophy” (427). In other words, we use a set of terms that represent the way we think about the world, these terms carry strong connotations and resonate strongly with the members of a society, and they are used as justifications in and of themselves even though their definitions are not extremely precise. These ideographs are words and phrases commonly used in political discussion, such as “equality” or “rule of law,” whose definitions are thought to be commonly understood by all members of the society but that are actually vague and shifting; they

are defined by consensus, change over time, and produce controversy and conflict when they are challenged with alternate definitions. For instance, while U.S. society has always espoused a strong admiration for the concept of "equality," and the term has often been used to justify legislation and social behavior, the actual meaning of "equality" is constantly shifting and certainly has been interpreted differently throughout history. The nation's long history of, and many battles with, racial and sexual discrimination illustrate this.

McGee makes two important distinctions about ideographs. First, they operate on all members of the society including both the ruled and the rulers. In this view, popular thought is not dictated by the ruling class to the proletariat, but instead it is a dialogue in which all parties participate. There remains an element of power and submission of course, but it is not strictly one way: "We make a rhetoric of war to persuade us of war's necessity, but then forget that it is a rhetoric—and regard negative popular judgments of it as unpatriotic cowardice" (428). In essence, we may create our ideographs but we cannot escape them; our leaders are just as beholden to our ideographs as we are, and must act in accordance with the common conception of them or risk alienation. They are, basically, the rules of the game. We may allow some players to have more advantages, and some of us have little choice but to play, but we are all still playing the game.

Second, he states that our cultural identities are inescapably defined by our ideographs. They are used both to unite and to separate. One can attempt to divorce oneself from society and engage in an unbiased examination of these terms, McGee states, but one can never be certain that one has fully escaped cultural bias. As the President of the United States is a member of the society of the United States, he can no

more free himself from the influence of its particular set of ideographs than can any other U.S. citizen.

McGee states that, at any moment, a culture has two "ideologies:" one, which he defines as a "grammar," consists of the collective historical definitions of ideographs (434). This functions as a historical precedent that gives a rough accumulation of and guideline for acceptable understandings of the terms. The second, he says, is a "rhetoric" that he defines as "a synchronic structure of ideograph clusters constantly reorganizing itself to accommodate specific circumstances while maintaining its fundamental consonance and unity" (434). In other words, at any moment, an ideograph must be at least partially defined by its relationship to other ideographs, which are themselves always shifting. McGee claims that, in order to properly understand a society's way of thinking, one must be aware of both dimensions and how they interact.

This implies that an ideograph is best studied when it is in conflict. Until it becomes necessary—through discourse and often controversy—to explicitly define what it is we mean by something like "the rule of law," it is merely a vaguely-held common belief with varying interpretations. The resulting discussion, in which the community is forced to assert its understanding of the ideograph by examining not only the "grammatical" aspect of past definitions but also the "rhetorical" aspect of its relations to other ideographs, is the best opportunity to define and examine the ideographs themselves. With this in mind, I turn my attention back to the conflict between word processing/presentation tools and XML/single-source tools.

Ideographs and Software

Based on the literature I reviewed in the academic journals, and informed by

McGee's definitions, I have identified several ways in which software behaves in an ideograph-like way in the technical communication field. While it does not *entirely* fit the definition, as I will discuss in the sections that follow, the field's relationship with software is similar enough to its relationship with political terminology that a discussion in ideographic terms is warranted.

Why Software is Ideographic

I argue that the field's relationship with technology is similar to an ideographic relationship in three ways: conflict over new paradigms occurs in ways that echo McGee's description of the rhetorical/grammatical axes of ideographs, the community's relationship with software acts as a guide for the creation and maintenance of power structures, and these relationships are pervasive and not selectively used by one group or another. In this section I will elaborate on these similarities.

The most prominent similarity is one revolving around conflict. McGee describes the ways that an ideographic conflict would play out in a society, in which an accepted way of thinking about a concept or a value is challenged. In these situations society is generally asked to accept an expanded or altered definition of an ideograph it felt it already knew. This conflict causes the culture to reexamine its ideographic definitions, using historical interpretations as a guide, and either accept or reject the new definition. It is usually a question of relationships: ideographs, he argues, only have meaning as they relate to one another. He uses the example of the Watergate crisis, in which President Nixon committed actions many in the public considered illegal according to the ideograph “rule of law.” Nixon responded by claiming that the ideograph of “confidentiality” held more sway in the situation, essentially asking the public to alter the

definition they had for the relationship between the ideographs “rule of law” and “confidentiality” (433). The public, by way of the legislature, responded by examining its historical definitions of these terms: “No direct vertical precedent was available to support Nixon's usage” (433). The attempt was ultimately unsuccessful.

Similarly, technical communication professionals are being asked to reevaluate a relationship: that between the traditional way of working and the new single-source method. In McGee's terms, the new technology represents a “rhetorical” challenge to accepted definitions of what software is and how users relate to it; the removal of formatting decisions from the writing process, the breaking up of content into topic-based chunks, and the increased emphasis on context-independent documents represent considerable departures from business as usual for the average technical writer.

Because these writers feel compelled to adapt to these changes, due to both real and perceived pressure from management and the marketplace, they feel equally compelled to find an acceptable method from the existing canon of rhetoric and writing studies with which to view the technology. This is demonstrated particularly well in Stewart Whittemore's “Lessons from the Canon of *Memoria*,” which argues that adapting methods from well-respected rhetors of antiquity is an acceptable way of conceptualizing the new paradigm. Whittemore and authors working in the same vein are, in McGee's terms, searching for “grammatical” answers to a “rhetorical” problem; they are hoping that by finding a method the field already is comfortable with, they can make more palatable the uncomfortable realignment that single-source technology will bring to their writing methods.

Next, the community's conceptions of software tools act in an ideographic way in

the way that they relate to power. As defined by McGee, the phenomenon of the ideograph facilitates relationships of structure and authority: “It warrants the use of power, excuses behavior and belief which might otherwise be perceived as eccentric or antisocial, and guides behavior and belief into channels easily recognized by a community as acceptable and laudable” (435). Thus, at the same time that ideographs represent the way a culture views itself and the world, they create new power structures and solidify existing ones. They create the rules of the game, so to speak; for example, a culture that values “rule of law” as a primary ideograph may legitimize legal authorities as holders of power, and create situations in which it is acceptable to exercise this power. A culture that values “rule of law” less than it does, say, “tradition” will legitimize different uses of power and different individuals as holders of power. Even two cultures that ostensibly value the same ideograph can express it in different ways; “rule of law” may mean fidelity to centuries-old codified laws in one culture, but may mean adherence to unwritten legal precedent in another. The ways in which individuals or groups can acquire, maintain, or even lose power will be different according to the relationships their culture has with its ideographs.

Lanier's articles about electronic editing, which I encountered in my research, suggest a similarly ideographic relationship between software and power. In them, he advises using software tools (specifically Microsoft Word) to secure power for writers and editors in an organization. These articles offer a clear example of a piece of software —a word processing tool—that has not only warranted the use of power but has been used to create new power structures. Tellingly, much of the field's uneasiness with the transition to single-source tools comes from a perception that those tools are less

conducive to the creation of power for writers and in fact will take power away. Carter's anecdote, in which a writer threatens to “walk” if asked to adapt a single-source workflow, illustrates this viewpoint vividly.

Even when not explicitly speaking about the creation or loss of power, McGee's characterization of the ideograph as guiding acceptable behaviors is evident in the field's discussion of software tools. From David Coniam's article—which guides users into a specific way of interacting with Microsoft Word to achieve an organizational goal of heightened efficiency—to the many defenses and tips for improving PowerPoint—in which rhetorical debate over a software tool was eschewed in favor of similar but more elaborate methods of using the tool—the articles in my research suggest that the field's way of thinking about software tools guides its members into acceptable ways of using and talking about them.

Finally, an important characteristic of ideographs is their pervasiveness. They are not fictions invented by a ruling class for the specific purpose of creating power but are instead culture-wide constructs that affect all sides; even if one group has more power, all groups are affected. This trend is evidenced by the fact that the field's attitude toward software remains essentially the same regardless of whether the community is emboldened by software's capacity to grant power or frightened by its ability to take it away. That attitude can be summed up as the belief that agency exists with the software, not the individual. Whether discussing the most advantageous way to take advantage of Microsoft Word's power to change an organization or discussing the disturbing (to many professionals) world that XML/single-source will bring about, in which traditional writing practices are altered and new responsibilities are created, this underlying

viewpoint is consistent. While the level of comfort with the agency of software felt by individuals in the field varies depending on whether they are the beneficiaries or victims, their expectations for the mechanics of the process do not change. The software, not the user, is the change agent; the individual who controls the software controls the power.

This blanket understanding of the role of software in the community implies that, like McGee's description of how the ideographs of war operate, the community has created for itself a rhetoric of software and power. However, in many ways they have forgotten that it was a rhetoric and accepted it as self-evident truth. It is only natural to them that XML/single-source tools will take power away from someone, because they have created a rhetoric for themselves in which it can be used—it is even suggested that it should be used—to create power for writers either through organizational changes or heightened efficiency. This conception is so complete that they believe an unknown, and therefore unmastered, software can only be used by another group to wrest power away from them.

Why Software is Not Ideographic

For all the similarities between the community's understanding of software and McGee's description of ideographs, there remains a key difference that prevents a direct comparison. “An ideograph,” states McGee, “is an ordinary-language term found in political discourse” (435). It is part of a vocabulary that is “precisely a group of *words* and not a series of symbols representing ideas” (430). McGee is very clear on this point: ideographs are a phenomenon of *language*; they are well-worn phrases such as “pursuit of happiness” and “freedom of speech,” not generalized concepts such as a community's understanding of software.

This distinction would be moot if the articles in my research utilized such a familiar and repeated phrase to describe their relationship with software, but they do not. Instead, the characteristics of this relationship are indirectly discussed and act as a subtext to the concepts and issues directly being addressed in the pieces. This is not a minor difference; for McGee, the verbal and textual qualities of ideographs are the primary reason to study them. Quite simply, if there is no commonly-accepted term, there is no ideograph.

A New Definition

Because of this key difference, I cannot declare that the trends found in my research are purely ideographic in McGee's terms. Rather than presenting clear examples of his concepts, the articles I researched seem to imply the presence of an ideograph without making it clear what that ideograph is. We are left with all of the trappings of one —there is rhetorical and grammatical conflict, justification of power structures, and universal acceptance of a vaguely-defined definition—but no specific term to which a researcher can peg these beliefs; they are not words, but rather cultural texts that are more difficult to precisely define. Like a biologist seeking an animal in the wild, there is evidence for its presence—its prints, its distinctive habitat—but the beast remains camouflaged.

I believe that we will be able to see through the camouflage if we adjust our way of thinking. McGee's concept of ideographs is narrowly-defined, dependent upon a text-centric view of rhetoric that places the highest degree of value in words and language. Such a view becomes increasingly anachronistic as we move further into a world dominated by the digital. Our lives are controlled less and less by words on a page and

more and more by words on a screen; the power of the press has transferred to the ones and zeroes that lie behind those displayed words. McGee's assertion that ideographs are specifically textual served well to narrow his discussion and bring the focus back to rhetoric, but in this situation it is limiting: we now live in a digital world, and the language of that world is software.

When we expand our definition of rhetoric to include the means of digital production; when we come to think of language as not only the words we use to describe something but the tools that allow us to work with those words; we can come to an important realization. Just as it is not the words themselves that are important in ideographs, but the way we think about them, in the technical communication community *our conception of software behaves as an ideograph*.

Purely textual ideographs, such as the term "pursuit of happiness," have concrete meanings entirely apart from their ideographic definitions. One can use a dictionary to define each word individually and come up with a meaning that is objectively true while remaining unsatisfactorily vague; there will be no context for the term's social weight. This is due to the fact that we imbue terms with levels of meaning; words have definitions beyond their basic semantic senses. It is not enough to know that the "pursuit of happiness" means "chasing or striving for a feeling of contentment and joy," because such a definition does nothing to elaborate on the historically, culturally, and legally defined meanings for that term. A dictionary would do little to inform one that "pursuit of happiness" is often used in American society as a justification for the right to private property, for instance. The definition of a term and the concept carried by that term are often different, even though they are equally often conflated.

"Software," of course, is a term that carries its own dictionary definition. However, it is important to distinguish between *software-as-term* and *software-as-concept*. At no point does the community seem to disagree with their understanding of *software-as-term*; good or bad, ally or enemy, "software" is always indisputably a computer term meaning "The programs, routines, and symbolic languages that control the functioning of the hardware and direct its operation" (American Heritage Dictionary, "software"). However, the community has developed definitions for "software" that go beyond its literal meaning. Conflict over these expanded cultural definitions is what drives the ideograph-like qualities I have identified, not a debate over the *term* "software."

I argue that we must, as a field, embrace an understanding of software that acknowledges these expanded definitions and treats it as a rhetorical object in its own right. Continuing to view software only in its *software-as-term* form restricts us from seeing an entire spectrum of issues that are already at play in the community and in society at large. By adjusting McGee's rhetorical lens, we are able to see that it is possible to apply rhetorical thinking to software—the shoe fits, so to speak. In the next chapter, I will explain why it is not only convenient academically to adopt this viewpoint, but important socially and politically.

CHAPTER FOUR: SOCIAL IMPACT

Technology and the Humanities

As I described in my literature review, Michael Knievel argues that the field of technical communication has inherited a certain bias against technology. Perhaps the biggest challenge in asking the field to think of software as a rhetorical object in its own right, rather than as a useful but ultimately valueless object, is this bias. As Knievel describes, the field still identifies with the humanities, and "the humanities most typically defines its collective self against the sciences and technology, which it views as 'other'" (66). As a discipline that works intimately with technology, but that considers itself a subset of a school of thought that views technology as threatening, the field seems to have developed a rather rigid opinion of the ways it can and cannot discuss software.

Knievel's argument is that the academic thought that constitutes the field of "humanities," although largely informed by the philosophical concepts of humanism, diverges in key ways from humanism and has developed inflexible attitudes toward technology that are highly problematic to the field of technical communication. He argues that technical communication, by focusing on the effects of technology on individuals, has disregarded the more instrumental aspects of technology and neglected to discuss technology as a physical artifact. He says that such discussions, in which technology is discussed on its own terms, "operate at a less apparently rhetorical or textual level and thus, I suspect, are less readily identified as humanistic" (71). This has led, in his words,

to “a sort of dualism” between technologies that have and have not been “interiorized;” he uses the example of professors criticizing technology while not acknowledging the technology that allows them to do so in the first place, “such as books and pens—indeed, language itself” (78). Much as a scholar might decry the advent of online publication for work as being an unacceptable intrusion of technology on a traditional practice, not acknowledging that printing is itself a technology that has evolved steadily for centuries, technical communicators run the risk of accepting certain tools with which they have grown comfortable as “traditional” and forgetting that they are, in fact, technologies. This seems to be the prevailing attitude around the tools in the MS Office suite, as I identified in my research.

The authors have very clearly “interiorized” a set of technologies—word processing and presentation software—and are less apt to criticize them than they are to criticize the “new” technologies. While there may have been pages of ink spent discussing the difficulty in transitioning from pen-and-paper to word processing two or three decades ago, there is nothing in my research about these tools that approaches the level of anxiety and critique that the new technology engenders. The “interiorized” tools are, in my research, ubiquitous and accepted; the question is not what will happen when we start using PowerPoint, but how to use it better. However, unlike the academics of Knievel's example, which seem to have a fear of all things technical, they do not engage in blanket condemnation of technology nor do they fail to engage in an instrumental discussion of it. Instead, their dualism manifests in the way that they discuss instrumentalism: both sets of tools are powerful and can create texts, but one does so in a favorable way that reinforces their way of thinking and another one does so in a way that

seems dangerous.

I believe that much of Knievel's argument is on-target and I will draw considerably from it in making my own. However, I feel it important to discuss not only the strengths in his argument but also the flaws.

Strengths of Knievel's Argument

Knievel is very prescient in observing that the field's dominant approach to technology, in which it is viewed as outside the traditional humanistic experience and as a product of external forces, creates a situation in which "technology tools are seen not as loaded with *human* (a term that carries a clear value judgment) value but instead as a sterile threat to the richness of the humanities culture" (79). My research seems to clearly reinforce this opinion: software tools are discussed as go-betweens, as tools to create or remove power, but never discussed rhetorically on their own terms. They are not seen, as Knievel puts it, as "emerging from a shared culture" (79).

He makes the argument that the academic construction of "the humanities" has diverged from the actual thought of humanism as defined in *The Humanist Manifestos*, and that it should be possible to construct an approach to technology that "would rely less on the humanities' skepticism and more on humanism's combined optimism and inevitability: Technologies manifest human hopes and values" (79). The *Manifestos*, he argues, does not espouse such a negative view of technology; instead it views technology as an aspect of the human experience. Rather than see technology as an other, he advises, we should understand that it is inherently humanistic, even in its instrumental sense.

I believe that this is an obvious direction for the field. To do otherwise would be to engage in a self-deprecating cycle of consistently marginalizing the very thing with

which we are specifically trained to work. I believe the field already embraces this viewpoint to a degree; it certainly shows little hesitation to discuss its software and its tools in its journals.

Shortcomings of Knievel's Argument

Ultimately, for all the useful conclusions it draws and for all the important issues it brings up, Knievel's argument contains a flaw at its central premise: that software can accurately be discussed in an instrumental sense by the technical communication community. Such a concept requires one to ascribe to the idea that technology can be purely instrumental in the first place. Based on my research this does not appear to be the case; as already shown, software appears to perform a significant unspoken rhetorical function in the technical communication field. In fact, the shortcomings of the articles I have studied stem from an insistence on only speaking of technology in instrumental terms and in refusing to unpack *software-as-concept*—the meanings we give to software, not its technical definition.

Knievel is correct in observing that the field harbors a dualistic attitude toward technology that renders it reluctant to discuss technology as a central concern. The field seems interested in discussing the discreet *products* of technology—documents and presentations—but not in seriously discussing its software tools in the same way that it discusses literary genres or philosophical movements. Knievel's solution, to better embrace the instrumental definitions of technology and challenge the humanities to accept that such discussion is humanistic, will only perpetuate the attitudes I identified in my research: it is a call for more discussion of the products of technology, not a discussion of *software-as-concept*.

We as a field have customarily worked from one of two positions: that technology and software should be discussed purely in terms of their final products, or that technology and software should be discussed as the tools we see them to be. This is a similar situation to that discussed by McGee in his characterization of the debate between Marxists and the followers of Kenneth Burke over how to define the nature of the "philosophy of myth." McGee's answer, of course, was to point out that such a debate missed the point, that the truth was in the middle; his concept of ideographs developed from his observations. Similarly, we must realize that discussing software purely in terms of its products on one hand or purely as a tool on the other masks a more nuanced, ideograph-like understanding of software as a rhetorical object in its own right. Only when we embrace the fact that, like language, our software has the ability to guide our thought; only when we recognize that our anxiety over new ways of writing texts masks our willing participation in an existing power structure based on software; only when we are willing to engage in a *critical rhetoric* that seeks to affect the relationship between software and its users; only then will we begin to fully address the rhetorical nature of software.

CHAPTER FIVE: THE CRITICAL IMPERATIVE

Critical Rhetoric

While I believe strongly in the importance of reconceptualizing technology and software as ideograph-like rhetorical entities, it is not purely for the sake of academics that I advocate such a change. The rhetorical situation evident in my research is symptomatic of larger forces that, left unchecked, currently have and will continue to have serious implications not only for the field of technical communication but also for society at large.

The Need for Critical Rhetoric

Raymie McKerrow argues in "Critical Rhetoric: Theory and Praxis" that rhetoric itself can be a force for social and political change, untethered to a Platonic ideal of service to a greater truth. "The task," he says, "is not to rehabilitate rhetoric, but to announce it in terms of a critical practice" (441). I shall briefly explain McKerrow's vision for such a critical practice before describing how and why it should be applied to my findings.

McKerrow argues that Plato's arguments about rhetoric place it in service of "truth," which burdens all subsequent theorists with having to prove that rhetoric has worth at all. In order to do so, he argues, these theorists have all relied on universal concepts of truth that leave rhetoric as a secondary and marginalized field (441). Universal concepts of truth are problematic because they are essentially impossible to

prove; as already discussed by numerous scholars including McGee, what is “true” to one culture is not necessarily “true” to another, or even to members of the same culture at different periods of time.

Thus, a critical rhetoric must be “divorced from the constraints of a Platonic conception” and must recognize the relativistic nature of power structures. Such a rhetoric “seeks to unmask or demystify the discourse of power” (441). Power, McKerrow explains, is the result of an interrelated network of “ideologies” or “discourses”; whether or not a particular mode of thought is “correct” in any universal terms is not nearly as important as how it works to create and maintain acceptable ways of acting. This idea, he explains, draws from Foucault's concept of “orders of discourse”: social constructs that determine who can speak, about what, and when.

In this way, McKerrow agrees with McGee that power is created through rhetoric. McKerrow places more emphasis on the agency of the dominant class in creating this power—he describes the rhetorical conventions as “institutionalized rules accepted and used by the dominant class to control the discursive actions of the dominated”(443)—but he agrees with McGee that the current of power runs in both directions. “Those who are dominated,” he says, “also participate in the social structure and are affected by—and affect—the orders of discourse by which their actions are moderated” (443). His view on the ability of the dominated to affect change, however, is somewhat pessimistic. Arguing from within the power dynamic for change is self-defeating, he claims, because it acknowledges and validates the arrangement even as it criticizes it—in such a situation, there is at best an illusion of agency. “Actions oriented toward change will tend to be conducive to power maintenance rather than to its removal” (444). In McKerrow's

estimation, for instance, arguing for a new office software suite to replace MS Office as a "standard" is self-defeating because it validates the very concept that there should be a hegemonic standard in the first place.

A critical rhetoric, McKerrow claims, is the best option in this situation because it focuses not on the specifics of the power arrangement but on the rhetorical situations that allow such an arrangement to exist in the first place. Such a practice is freeing, he argues, not because it argues for any predetermined notions but because it offers a constant criticism; a critical rhetoric, echoing Foucault's argument for permanent criticism, has no particular goal other than a constant and skeptical examination of power. It acknowledges the difficulty in arguing from within a structure to change that structure, but offers an optimistic prospect for—if not radical and immediate change—at least a consistent incremental change.

Similarly, a critical rhetoric does not focus solely on "the question of the legitimacy of the state" (447). Power dynamics exist in all facets of life; it is counterproductive to only exercise our powers of critique on certain "acceptable" expressions of power. It is productive to examine all facets. Focusing on power purely in terms of legislation and jurisdiction, McKerrow states, "is a Western, democratic conception of power that is rational and orderly" but is ultimately insufficient to describe the full universe of power dynamics. This is the motivation that led Foucault to discuss power in relation to sexuality: "In this context, the analysis of power in terms of a juridical model, or in terms of a 'war' model, would be too far from the mark to be helpful" (448).

The Principles of Critical Rhetoric

McKerrow concludes his essay by offering a set of eight principles on which to base a critical rhetoric. They are not rules in the sense that one can use them as a checklist for analyzing a text; they are instead a set of concepts that he argues should guide critique. Indeed, the loose nature of these principles in which they act as guides and not rules is expressed in the first one: "*Ideologiekritik* Is In Fact Not a Method, But a Practice" (452). Referencing McGee, he argues that "creative insights are constrained by the systematicity of method" (452). Critical rhetoric is not a method but an orientation, a way of looking at texts and at research that proceeds from a basic perspective but is not rigidly controlled. This is "the least restrictive stage from which the critical act might be launched" according to McKerrow (452).

The other seven principles continue in this vein, affirming the immediate and material effect of rhetoric, the creative and interpretive nature of criticism as opposed to a precise and scientific endeavor, the importance of recognizing multiple interpretations of events, and a general respect for the fluid, performative nature of critique. McKerrow acknowledges that his rejection of Platonic universalism "raises a question: 'Have we abandoned the Platonic quest and embraced sophism?' The answer is 'yes'"(449). A sophisitic appreciation of the ways that rhetoric influences and creates power, and an understanding that meaningful change begins with a change in discourse, is more important to McKerrow than allegiance to a philosophically idealistic construction of "truth."

Implications for Technical Communication

McKerrow encourages, even challenges, us to see critical examination of rhetoric

not only as a worthwhile academic endeavor but one with direct social and political implications. I argue that, given my research findings and the conclusions that can be drawn from them, the technical communication field is in a unique position and should seize the opportunity to rethink its relationship with technology and software. Each of the three theorists I have discussed—McGee, Knievel, and McKerrow—have articulated viewpoints that inform my argument.

Applying McGee's concept of ideographs, I have demonstrated how software itself—not just the language we use to talk about software—behaves ideographically. Because the field argues over its interpretation and its use in the same manner that (according to McGee) communities argue over social and political terminology, it is clear that software fulfills many of the same functions that language does. Due to its ability to shape our thoughts and actions, to limit what we can and cannot do, and to create or realign power structures, it can be said that software is the language of a world dominated by the displayed word and carries with it all the same rhetorical importance.

Knievel shows us how the field of the humanities has developed an institutional bias against discussing technology and software directly. He demonstrates how this has led to a dualistic manner of thinking in which certain technologies are treated as non-human “others,” unworthy of serious discussion, while other technologies such as printed books are “interiorized” and not discussed as technologies at all. This dualism rejects the inherent humanistic nature of technology, Knievel argues, and goes against the actual mission of humanism. Further, because technical communication draws much of its influence from the humanities, this has led to a kind of institutional identity crisis in which we must discuss technology but feel pressure not to. And, just as humanities

professors have interiorized the book and the pen, I argue that we have interiorized software—thinking of it as a value-neutral means to an end and largely avoiding talking about it as a rhetorical object.

McKerrow shows us that, once we realize that rhetoric creates power, continually engaging in a critique of that rhetoric is the only way to engage with and affect this process. For McKerrow, discussing rhetoric is not an abstract academic concern but an inherently active, meaningful, and political act. Furthermore, studying and critiquing any particular discourse should be a cyclical process, always doubling back on and continuously reevaluating itself.

It is my contention that these three perspectives in combination have clear implications for the field of technical communication. If we agree that our software has value beyond its application as a tool, behaving rhetorically as a kind of new language; if we agree that rhetoric is used to create and maintain power, and the only way to engage with this dynamic is through constant critique; if we agree that technical communicators, as experts in rhetoric, have been unwilling to acknowledge that software is rhetorical because of the strained relationship with technology they have inherited; then the argument is clear. Technical communication must realign its relationship with technology, specifically software, and engage with it as an inherently rhetorical medium with social and political implications.

It is clear that we as a field are uniquely qualified for this position, and it is equally clear that doing so is important not just to our academic peers but to all users of software. In order to do so, we must reevaluate our priorities and establish principles for studying, discussing, and teaching software that acknowledge its importance beyond

instrumentalism; software must become to us a language, not just a means to an end. In the final chapter, I will expand on this assertion.

CHAPTER SIX: CONCLUSION

What We Must Do

If we agree that our ability to contribute to the social and political debate surrounding software comes from our ability to engage in critical rhetoric, we must come to a decision about what that means for us as a field. In this chapter, I will lay out my recommendations for action, particularly in academic settings.

Knievel is correct in identifying the dual nature of technical communication's loyalties; it is a field that operates within the world of the academy, and receives much of its supporting theory from that world, but is it also has a much more direct relationship with the world of industry than almost any other field within the humanities. There are technical communicators in academics and technical communicators in business; the two may have much in common but they deal with very different day-to-day realities. Defining a course of action that would work for both is thus difficult at best.

I will focus on the academic side of the field for two reasons. First, as McKerrow explains, engaging in discourse and rhetoric around the creation of power is the most effective way to change the dynamic. Those working from within the academic sphere are, I believe, best equipped to begin this dialogue. I emphasize my use of the word "begin": those in the industrial sphere are just as capable of contributing to the dialogue as the academics, and will undoubtedly contribute in ways that the academy could not. However, given the need to establish a framework based on rhetorical theory drawn from

philosophical sources, it seems most natural that the academically-inclined among us should start the conversation.

Second, the students that we teach will be the industrial communicators of tomorrow. We have an opportunity not only to discuss the issues among ourselves, but to instill in our students a strong critical theory of technology and software. I look to the field of journalism for an example: by instilling values and ethics that derive not just from professional pragmatism but also from a sense of duty to society at large, including a preference for reporting that which is true above that which is beneficial to one's employer, the field of journalism not only participates in academic debate but instills in each generation of journalists a basic understanding of their professional and ethical obligations (even if some journalists choose to ignore these obligations).

The press can be used to control a population, to influence thought and suppress dissent; recognizing this, journalists idealize independence and fidelity to truth first and foremost. Technical communication programs could more strongly emphasize a similar ethic of independence that recognizes the capability of technological products to reinforce existing power structures. A government can make laws, but it requires the assistance of journalists to inform and educate the public; similarly, a technology producer can create a product, but usually requires a technical communicator to educate users about this product. Ours is a unique position of power, and we have an equally unique opportunity to craft an independent professional ethic. Existing codes of ethics, such as the one maintained by the Society for Technical Communication, speak to workplace behavior more than toward any critical or rhetorical imperatives.

The remainder of this piece will focus on suggestions for action in academic

settings, both to classroom practice and to overall curricular design at the college level. These are actions that are within our capability to achieve and draw from our background in rhetorical and critical practice. I suggest that we must take two general actions: we must address the rhetorical affordances of software and we must examine the influence of the market.

Address the Rhetorical Affordances of Software

We must recognize and address the way that specific software tools affect the production of texts. We already discuss the rhetorical effects of specific texts, such as the genre characteristics of workplace reports, the audience considerations of technical manuals, or the teaching effectiveness of slide decks. We are at least aware of the general ways that digital technology affects us, with important research existing on the subjects of non-linear text, new media, and networked writing. But when presented with direct criticism of a specific tool, such as when Tufte described the ways that PowerPoint affects the creation of texts, we seem to respond by defending the tool and trying to drive the conversation back to the product of the software.

In the world of traditional textual criticism, we do not shy away from discussing the ways in which language itself affects the composition process as well as the resulting expression. We cannot continue to behave as if the truth of this revelation does not apply to the software we use to express that language. What possibilities for rhetorical growth may we discover if we seriously begin to study the ways that our tools—our "language"— affect our texts?

For example: If I am composing a lengthy document in a word processing program, and I wish to insert a new chapter into the body with different formatting on the

first page, I have more than one way of doing so depending on which program I use. In order to do so within the standard workflow for Microsoft Word I must establish a new "section," utilizing Word's metaphor for organization that separates the text into separate chunks. LibreOffice, the open-source word processing suite included with many Linux distributions, accomplishes the same task by utilizing specific page styles within the text body and does not use the concept of "sections" in the same way. The functional effect of words on the page is the same—a new page with a heading and different formatting—but each program takes a different conceptual route to get to this destination. The Word method seems to encourage me to think of my text as the vertebrae in the neck of a giraffe: a set of discreet sections held together to create a larger structure; the LibreOffice method instead causes me to think of the document more holistically, with the differences in formatting throughout more like the mottled spots on the giraffe's neck. How does this affect my composition? What are the other ways that my software shapes me? How could this change the way we discuss software in general?

This is one side of the spectrum: the effect that the functionality of a particular piece of software has on us—the way its "grammar" affects our text. In our research and in our teaching, we must explore this issue. We must encourage our students to see not only the functions of the software—the convenient features that allow them to organize, format, and spell-check their texts—but also the assumptions behind those features. Why are they organized the way that they are? How do these features encourage one to look and think about at the text? Requiring students to use more than one example of a type of software tool, or creating assignments that focus on the assumptions that the authors of a software tool appear to have made about the users, will help our students engage with this

discussion. As made clear with Tufte's criticism of PowerPoint and the whirlwind of response it generated, it is a discussion that we have neglected for too long.

Examine the Influence of the Market

On the other end of the spectrum, we must address the rhetoric of software on a larger scale: the ways in which it intersects with dynamics of power and economic influence. While the specific functionality of a piece of software—its "grammar"—affects users in small but important ways, the market forces and business decisions of software creators affect users in ways more easily measured in societal terms.

In the world of software, we tend to use the term "standard" in different ways depending on context. There are standards developed by organizational bodies, such as the Hypertext Markup Language or HTML, that are codified rules for how a technology should behave. On the other hand, there are "standard" programs that, usually through market dominance, have come to represent what we expect from different kinds of software; they are the example by which all others are judged. There is Microsoft Word for word processing, Adobe Photoshop for image manipulation, Google for Internet search; in all but the first case, in fact, the name of the program itself has become a generic verb for the type of activity you are performing: regardless of the program you use to manipulate an image, popular usage describes the action as "Photoshopping" it. In both the business and academic worlds, we tend to accept these standards of judgment; they are empirically true based on market share and common experience. We perpetuate them in our research and our instruction; it makes sense to teach the norms of the industry to our students. They will, after all, need to eventually find employment in the industry.

Why are these programs the standards of judgment, however, and why did they

rise to prominence over their competitors to achieve the market dominance they currently enjoy? How much has to do with their ease of use and technical superiority, and how much has to do with the highly effective business acumen of the companies that produce them? Once we declare that we must teach a program because it is what everyone else uses, we engage in a feedback loop wherein we reinforce the market dominance of a particular software company. In some cases, this can even elevate a "standard of judgment" into an actual "standard"; the public embrace of Portable Document Format (PDF), originally an Adobe proprietary format, led to it becoming the *de facto* standard for platform-independent document transmission. This effect may be unavoidable and even desirable in some cases, but it should not be automatic.

We must begin to engage in a discussion of, and teach our students about, the ways that software producers use market forces to establish themselves as standards of judgment. Influential open-source software advocate Eric S. Raymond, for instance, famously published and commented on leaked internal documents from Microsoft discussing the threat posed by the GNU/Linux operating system to their market share. In the first of these so called "Halloween Documents," the author (an unnamed Microsoft employee) advocates a method for discouraging competition from open-source products that speaks directly to the balance of power in software. The author argues that, because the popularity of Linux and its ease of adoption are based on open standards, Microsoft should use its business capital and name recognition to first embrace those standards and ingratiate itself with the community. Then, slowly, Microsoft should "extend" these standards with their own proprietary enhancements such that users are increasingly required to use Microsoft products to take full advantage of them. Finally, once they have

"enhanced" them enough that a large number of users identify the standard with Microsoft, they will have essentially converted an open standard into their own proprietary standard and will have eliminated competition ("Halloween Papers 1").

If we perceive of software as language, then it is to some degree cultural. The Microsoft policy described above, called "embrace, extend, extinguish" by the open-source community, is then a kind of cultural imperialism. Business is business and we cannot expect corporations to behave otherwise—but neither can we ignore it. Word, Photoshop, or Google may very well be the best tool for the job; but we must be aware of the circumstances that brought our chosen tool to prominence and be capable of making an informed decision about adopting or teaching it. If it is possible to teach the works of the traditional literary canon while acknowledging that the canon is the result of cultural forces that favor certain viewpoints, genders, and cultural perspectives, then certainly it is possible to teach Adobe Photoshop while also teaching about Adobe's business practices. Courses in the business practices of major software corporations, or assignments that emphasize the history of "standard" tools, would help to give students a "macro" perspective; this perspective, compared with the "micro" perspective of the ways in which software functionality directly affects texts, will offer them a full sense of context from which to make informed professional and rhetorical decisions.

How We Must Do It

As I have stated, I believe this change must start from within the academy. It must take the form of published research and conversations in journals as well as lesson plans. I argue that these actions can be broadly grouped into three directives: *engage in research*, *teach the rhetoric of software*, and *assert software independence*.

Engage in Research

We must engage in, and publish, research on the comparative rhetorical nature of software tools. This should include comparisons of software tools within similar genres; such research will be in many ways similar to the research already undertaken by usability experts, but with a more explicit emphasis on the rhetorical assumptions made by and options given to the user. Research of this sort will describe, for instance, the ways that word processing tools ask users to conceptualize their texts, the ways that image manipulation programs organize and facilitate the user's visualization of the final product, and the ways that HTML editing programs simplify certain functions and hide others from the user. Such comparisons will enable us to identify trends in software design and engage in a sort of critical genre study of software. This will allow us to discuss the grammar of software.

We must also discuss the context of software, including its background, the political and corporate cultures that enabled it to exist, and the social implications of its use. Such research would include historical studies of software tools, identifying perhaps the changes between versions of a tool as it became more popular or the management priorities of the software producers. Establishing a context for software along with a grammar, just as McGee stated in his discussion of ideographs, will allow us to come to a more informed position from which to discuss the rhetoric of software.

Teach the Rhetoric of Software

In addition to engaging in research, we must encourage our students to view software not as a means to an end but as a rhetorical artifact. This should include courses in the rhetorical nature of specific software tools or genres of software; a “rhetoric of word processing” if you will.

Drawing from our research in comparative rhetorical studies, we can introduce students to the ways that different software tools affect the way that they create their texts. Not only will this encourage students to adopt a more critical view of all software tools in general and assist in fostering a critical rhetoric of software, it should also strengthen their skills as users of software—it will improve their command of software "grammar" as well. A user who has been shown more than one way to accomplish a task will be able to learn future software tools more quickly; software is, after all, a language.

By incorporating a critical rhetoric into the classroom, we will fulfill not only our humanities-based mission of producing critically-thinking members of society but also our professional responsibility to produce competent professionals capable of solving problems with creative thinking. Such a method of instruction may be intimidating to students who are unused to thinking of a computer as anything other than a tool, or to instructors who are uncomfortable with the idea of needing to stay informed of developments in software. But the benefits that can be reaped from instilling a critical rhetoric of software in the classroom greatly outnumber the negatives of ignoring its rhetorical importance.

Assert Software Independence

While it is necessary that our field work closely with industry in order to stay involved in the technological conversation and to provide our students with employment opportunities, it is also necessary that our critical examination of software not occur in a vacuum. Software producers, both commercial and non-commercial, have a necessary interest in acquiring and maintaining users. They will aggressively seek the cooperation of academic organizations, including technical communication departments, in

establishing and maintaining a user base.

This can create a difficult situation for administrators, as software is often not cheap. A free or cost-reduced donation of software or hardware from a corporate entity may be seen as a windfall for a department; it may enable students access to technology they may otherwise not have. However, it carries its own political, social, and rhetorical consequences.

Settling on a particular program as a standard, either in making it the only available alternative, making all resources available only for it, or only teaching it in class, comes with social and political responsibilities that departments must consider. Distributing course documents in, for instance, Microsoft Excel format forces those students who do not own the software to either use the department's computers or purchase the software themselves; in either sense this can prove a hardship for distance students or those with financial difficulties. Departmental websites that do not conform to standard coding practices, for instance those that only work in Internet Explorer, similarly discriminate against students that do not use Windows products.

This does not mean that departments cannot settle on standards, even those that can be exclusive. It means that departments must consider not only the financial implications of a donation but also the social implications, and make every effort to be inclusive in their choices. Distributing departmental communication in an open format, such as PDF or standards-compliant HTML for instance, instead of a proprietary format will ensure that the highest possible number of students can view it.

By “assert software independence,” I mean that the software choices made by technical communicators at the academic level should only be made after significant

rhetorical and philosophical considerations. In other words, we should recognize and distance ourselves from the efforts of software producers (even open-source ones) to acquire users and, to the greatest extent possible, make decisions based on a critical perspective and not on expedience. Some may believe that this would sour our relationship with the software industry, and that we need their patronage in order to continue teaching. It is possible, I counter, to have friendly relationships with entities to which you are not beholden. We are, after all, important to them.

Final Thoughts

In writing this, I was affected by the tools I chose—even if it was in very subtle ways. For this reason, I feel it important to explicitly call attention to these tools: I wrote this using the open-source word processing program LibreOffice, version 3.4.3, on an Apple notebook computer running version 6.0 of the open-source Debian GNU/Linux operating system. Whatever the format of the document you are reading, it is important to remember that the final product was influenced by the tools I chose.

As a student and as a writer, I cannot help but think that my choice of software tool is more than a matter of economics or convenience. Nor do I think that it is a choice to be made lightly, without examination or critique. As I stated in the introduction, I believed at the start that it was a rhetorical choice. I only feel more strongly so now.

In this project I have attempted to show that by calling attention to our tools we are better able to engage in a discussion about the ways in which they affect us. My research suggests that we intuitively interact with software at a rhetorical level; I suggest that we take the next step and discuss this phenomenon outright. Surely there has never been a better time nor a field better prepared to begin this conversation.

WORKS CITED

Alley, Michael and Kathryn A. Neely. “Rethinking the Design of Presentation Slides: A Case for Sentence Headlines and Visual Evidence.” *Technical Communication* 52.4 (2005): 417-426. Print.

Alley, Michael, Madeline Shreiber, Katrina Ramsdell, and John Muffo. “How the Design of Headlines in Presentation Slides Affects Audience Retention.” *Technical Communication* 53.2 (2006): 225-234. Print.

Amare, Nicole. “To Slideware or Not to Slideware: Students' Experiences with PowerPoint vs. Lecture.” *Journal of Technical Writing and Communication* 36.3 (2006): 297-308. Print.

Andersen, Rebekka. “The Rhetoric of Enterprise Content Management (ECM): Confronting the Assumptions Driving ECM Adoption and Transforming Technical Communication.” *Technical Communication Quarterly* 17.1 (2008): 61-87. Print.

Apple, Inc. *iWork*. Apple, Inc., 2012. Web. 28 Feb. 2012. <http://www.apple.com/iwork>

Applen, J.D. “Technical Communication, Knowledge Management, and XML.” *Technical Communication* 49.3 (2002): 301-313. Print.

Battalio, John T. “Extensible Markup Language: How Might It Alter the Software Documentation Process and the Role of the Technical Communicator?.” *Journal of Technical Writing and Communication* 32.3 (2002): 209-244. Print.

Brumberger, Eva R. “The Rhetoric of Typography: The Persona of Typeface and Text.” *Technical Communication* 50.2 (2003): 206-223. Print.

Carter, Locke. “The Implications of Single Sourcing for Writers and Writing.” *Technical Communication* 50.3 (2003): 317-320. Print.

Coniam, David. “Word-Processing 'Efficiency' – By Means of Personalized Word-Frequency Lists.” *Journal of Technical Writing and Communication* 31.2 (2001): 175-187. Print.

Dayton, David and Keith Hopper. “Single Sourcing and Content Management: A Survey of STC Members.” *Technical Communication* 57.4 (2010): 375-397. Print.

Eble, Michelle F. “Content vs. Product: The Effects of Single Sourcing on the Teaching of Technical Communication.” *Technical Communication* 50.3 (2003): 344-349. Print.

Farkas, David K. “Explicit Structure in Print and On-Screen Documents.” *Technical Communication Quarterly* 14.1 (2005): 9-30. Print.

Farkas, David K. “Managing Three Mediation Effects That Influence PowerPoint Deck Authoring.” *Technical Communication* 56.1 (2009): 28-38. Print.

Johnsen, Lars. “Document (re)Presentation: Object-orientation, Visual Language, and XML.” *Technical Communication* 48.1 (2001): 59-65. Print.

Knievel, Michael. “Technology Artifacts, Instrumentalism, and the *Humanist Manifestos*: Toward an Integrated Humanistic Profile for Technical Communication.” *Journal of Business and Technical Communication* 20.1 (2006): 65-86. Print.

Lanier, Clinton. “Creating Editorial Authority Through Technological Innovation.” *Journal of Technical Writing and Communication* 39.4 (2009): 467-479. Print.

Lee, Chien-Ching. “Specific Guidelines for Creating Effective Visual Arguments in Technical Handouts.” *Technical Communication* 58.2 (2011): 135-148. Print.

Liebowitz, Stan J. and Stephen E. Margolis. *Winners, Losers & Microsoft: Competition and Antitrust in High Technology*. Oakland: The Independent Institute, 1999. Print.

Mackiewicz, Jo. “What Technical Writing Students Should Know About Typeface Personality.” *Journal of Technical Writing and Communication* 34.1 (2004): 113-131. Print.

Mackiewicz, Jo. “Perceptions of Clarity and Attractiveness in PowerPoint Graph Slides.” *Technical Communication* 54.2 (2007): 145-156. Print.

Mackiewicz, Jo. “Audience Perceptions of Fonts in Projected PowerPoint Text Slides.” *Technical Communication* 54.3 (2007): 295-307. Print.

Mackiewicz, Jo. “Comparing PowerPoint Experts' and University Students' Opinions About PowerPoint Presentations.” *Journal of Technical Writing and Communication* 38.2 (2008): 149-165. Print.

McGee, Michael Calvin. “The 'Ideograph': A Link Between Rhetoric and Ideology.”1980. *Contemporary Rhetorical Theory: A Reader*. Ed. John Louis Lucaites, Celeste Michelle Condit, and Sally Caudill. New York: The Guilford Press, 1999. 425-440. Print.

McKerrow, Raymie E. “Critical Rhetoric: Theory and Praxis.” 1989. *Contemporary Rhetorical Theory: A Reader*. Ed. John Louis Lucaites, Celeste Michelle Condit, and Sally Caudill. New York: The Guilford Press, 1999. 441-463. Print.

“More to Life Than the Office.” *Bloomberg Businessweek*. Bloomberg L.P., 3 July 2006. Web. 28 Feb. 2012. <http://www.businessweek.com/magazine/content/06_27/b3991412.htm>

Raymond, Eric S. “Halloween Document 1.” *The Halloween Documents*. Eric S. Raymond, 2010. Web. 28 Feb. 2012. <http://catb.org/esr/halloween/halloween1.html>

Robidoux, Charlotte. “Rhetorically Structured Content: Developing a Collaborative Single-Sourcing Curriculum.” *Technical Communication Quarterly* 17.1 (2008): 110-135. Print.

Rockley, Ann. “The Impact of Single Sourcing and Technology.” *Technical Communication* 48.2 (2001): 189-193. Print.

Sapienza, Filipp. “Does Being Technical Matter? XML, Single Source, and Technical Communication.” *Journal of Technical Writing and Communication* 32.2 (2002): 155-170. Print.

“Software.” *The American Heritage Dictionary of the English Language*. 5th ed. 2011. Print.

Swarts, Jason. “Information Technologies as Discursive Agents: Methodological Implications for the Empirical Study of Knowledge Work.” *Journal of Technical Writing and Communication* 38.4 (2008): 301-329. Print.

Tufte, Edward. “The Cognitive Style of PowerPoint: Pitching Out Corrupts Within.” *Beautiful Evidence.* Cheshire, CN: Graphics Press, 2006. 156-185. Print.

Whittemore, Stewart. “Metadata and Memory: Lessons from the Canon of *Memoria* for the Design of Content Management Systems.” *Technical Communication Quarterly* 17.1 (2007): 88-109. Print.

VITA

Jeremy Tate English was born in Cleburne, Texas on June 17, 1981. He attended Granbury High School in Granbury, Texas, graduating in 1999. He then entered the University of Texas at Austin, earning a Bachelor of Science in Radio-Television-Film in 2003 and a Bachelor of Arts in English in 2005. He is active in the independent film community of Austin, TX, and since 2007 has been employed in many capacities—including quality assurance, software requirements analysis, and information development—by Multimedia Games, Inc., a casino-gaming software company in Austin. He entered the Graduate College of Texas State University-San Marcos in August 2009.

Permanent address: tate.english@utexas.edu

This thesis was typed by Jeremy Tate English.