A Hypertext-Based Annotation System
For Electronic Scholarship

John McLean Whittet

May 5, 2008
# Table of Contents

- Autobiographical Sketch
- Final Thesis Report
- Thesis Proposal
- Thesis Pre-Proposal
AUTOBIOGRAPHICAL SKETCH

John McLean Whittet is a BS Computer Science major with a minor in Engineering Business and a graduation date of May 2008. His interests lie in the future of the Internet to develop and deploy desktop-quality applications with the addition of networking, delivery, and information provided by the Internet itself. He hopes to apply this interest to business development though a technology consulting position at Accenture, a global management consulting, technology services and outsourcing company. He boasts more than five years of web design and development experience in academic, non-profit, and commercial areas though Basseq Technologies, a self-run web design and development company. Additionally, he also has both academic and commercial experience with traditional software development processes. Applicable academic classes include most notably CS 340, Advanced Software Development, as well as the rest of the established BS Computer Science curriculum at The University. Professionally, John has worked for and with Fortune 500, non-profit, and private companies in the areas of defense, consumer products, and internet technology: all on projects related to the development of inter- or intranet applications.
A Hypertext-Based Annotation System
For Electronic Scholarship

A Thesis
in STS 402

Presented to

The Faculty of the
School of Engineering and Applied Science
University of Virginia

In Partial Fulfillment
Of the Requirements for the Degree

Bachelor of Science in Computer Science

by

John McLean Whittet

May 5, 2008

On my honor as a University student, on this assignment I have neither given nor received unauthorized aid as defined by the Honor Guidelines for papers in Science, Technology, and Society courses.

___________________________________________
(Full signature)

Approved __________________________________________________________________________ (Technical Advisor)

Thomas B. Horton (Signature)

Approved __________________________________________________________________________ (STS Advisor)

Patricia C. Click (Signature)
# TABLE OF CONTENTS

Table Of Contents........................................................................................................... ii  
Glossary Of Acronyms....................................................................................................... iii  
Abstract................................................................................................................................ iv  
I. Introduction............................................................................................................................. 1  
   Background & Problem Statement.................................................................................... 1  
   Social Context.................................................................................................................... 2  
   Technology Concepts......................................................................................................... 2  
   Scope ................................................................................................................................... 3  
   Going Forward.................................................................................................................... 4  
II. Literature Review................................................................................................................ 5  
   Annotation in Electronic Text......................................................................................... 5  
   Ajax and JavaScript Libraries....................................................................................... 6  
III. Impacts & Ethical Implications....................................................................................... 8  
IV. Materials & Methods........................................................................................................ 11  
   Client-Side Functionality............................................................................................... 11  
      Vanilla JavaScript, DOM Manipulation & AJAX Communications....................... 11  
      Markup & Presentation............................................................................................... 14  
      Usability....................................................................................................................... 15  
   Server-Side Processing...................................................................................................... 15  
      Database Schema....................................................................................................... 15  
      User Authentication.................................................................................................... 16  
      XML Markup............................................................................................................... 17  
V. Results & Discussion.......................................................................................................... 18  
   Shortcomings.................................................................................................................... 19  
   Successes............................................................................................................................ 20  
VI. Conclusion........................................................................................................................ 21  
   Interpretations................................................................................................................... 21  
   Recommendations & Future Research........................................................................ 22  
Bibliography........................................................................................................................... 23  
Appendices............................................................................................................................ 25  
   Appendix A: Technical Algorithmic Explanations....................................................... 25  
      A-1: Tree Walking........................................................................................................ 26  
      A-2: Deleting an Annotation...................................................................................... 27  
   Appendix B: Database Table Descriptions................................................................. 28  
   Appendix C: Screenshot & System UI Description.................................................... 29  
   Appendix D: Source Code and Test Application....................................................... 30
GLOSSARY OF ACRONYMS

Ajax  Asynchronous JavaScript and XML
CrUD  Create, Update, Delete
CSS   Cascading Style Sheets
DOM   Document Object Model
GUI   Graphical User Interface
HTML  HyperText Markup Language
HTTP  Hypertext Transfer Protocol
MVC   Model View Controller
SQL   Sequential Query Language
TAPoR Text-Analysis Portal for Research
W3C  World Wide Web Consortium
XML   Extensible Markup Language
ABSTRACT

This thesis will discuss the development and analysis of an electronic text tool allowing users to annotate specific portions of a source text via a web browser and Web 2.0 techniques and methodologies. The desire for such a system has been present in the humanities for at least the past decade, and only recently has the state of web browser technology progressed to the point where a seamless application is possible within the constraints of hypertext and its associated languages. Academic literature cites the need for electronic text tools, specifically annotation as a way for researchers to point out and share information with one another, while other texts explain the technological side of the equation, specifically relating to asynchronous hypertext requests. The ethical and societal impacts of this project are widespread, and could have a profound positive impact on the area of humanities computing with very little downside. Fundamentally, the system is a simple web application that relies on asynchronous communication and electronic document manipulation with JavaScript. The system provides a unique service to many different types of users, ranging from humanities researchers to professors to developers. An initial release indicates that such a product would be well received to all these users. The state of the Internet is such that an application allowing the sharing of data worldwide is a boon to researchers and developers alike, though improvements, suggestions, and ideas for this system in the future are diverse and exciting and could well provide an interesting alternative to traditional methods of annotation and sharing ideas.
I. INTRODUCTION

By creating a browser-based annotation system to aid users in marking up and sharing comments on electronic texts, this project aims to apply Web 2.0 techniques to humanities computing. Using cutting-edge technologies and methodologies, this system was written and analyzed to provide an insight into the successes and shortcomings of such an approach.

BACKGROUND & PROBLEM STATEMENT

More than seven centuries of written word have left humanity with a wealth of knowledge and information. The advent of the computer—and, later, the Internet—have helped humanities researchers analyze text faster and more effectively; the use of text tools to discern patterns, similarities, and usage provide those studying such literature an abundance of statistics. This data is used to provide insights into our language and culture, comparisons between genres and authors of literature, and an understanding of our literary progression from the works of Aeschylus to the words being penned or typed by an unknown author right now. Once connected to the Internet, collaboration becomes not only a possibility, but also an academic imperative. As such, the ability to both discuss aspects of an electronic work and make these thoughts available to others worldwide has topped the wish lists of many a humanities researcher. As such, annotation, or extra information asserted with a particular point in a document, has become a necessary, and lacking, aspect of electronic text functionality.
**Social Context**

Ray Siemens—Canada Research Chair in Humanities Computing at the University of Victoria and a member of the Text Analysis Portal for Research (TAPoR) project—has noted that electronic text, while useful, does “not yet offer text-analysis features commensurate with expectations even at the beginning of the last decade” (Siemens 93). This project aims to address some of these expectations, specifically as they relate to annotation and the sharing of ideas. Simply put, annotating a source involves marking up a portion of text for the purposes of notation; the digital equivalent of highlighting or notes in the margin. Annotation on the web, especially as it relates to Ajax and interactivity, is all but non-existent, but could provide a shared environment for researchers to collaborate with other experts across the globe, no matter their geographic location.

**Technology Concepts**

As a technical paper, the context of this project goes beyond mere humanitarian pursuits. Software engineering as a profession is quickly moving toward the delivery of applications across a network, often with a shared code base. These applications, entirely Internet-based, allow users to access a variety of data, interact with other product users, and store client-specific settings in such a way that moving between workstations requires no user-specific profile or data acquisition. From a deployment view, such applications are easier to support—as patches and updates are applied across a customer base instantly—and run almost regardless of normal hardware and operating system (OS) metrics, current browser quirks notwithstanding. Software advances and programmer ingenuity have been combined to bring previously desktop-specific features—such as interactivity and rich media—to web applications. Through the software investigations in this project, the author examined
applications of traditional software engineering techniques to more dynamic combinations of client-server architecture. These investigations become more technical, and as such will be discussed in the following sub-section.

The exploration of this subject area included the creation of a web-based textual annotation tool and related features; as discussed above, such web applications are fast becoming the de facto standard for simpler software. This tool provides a set of operations to be performed on a repository of information by several users. This system is presented to the user with a logical and interactive user interface utilizing semantic HTML and Asynchronous JavaScript and XML (Ajax) requests. Additionally, markup and presentation—the design and structure, respectively, of the comment information—is itself an important topic, as to facilitate usability, simple maintenance, and graceful degradation to screen readers or mobile devices.

SCOPE

This project delivers a web 2.0 system to explore annotation in a web-based environment in order to address the seemingly immense shortcomings in digital text analysis libraries. The nature of the work implies a symbiotic relationship with other browser-based electronic text analysis tools; where projects such as TAPoR deliver marked-up electronic text with various algorithmic transformations applied to the content, this system would provide a layer on top of the text for researchers to mark and discuss aspects of the work in an atmosphere as simple as the proverbial notes the margin or as complex as a full-blown social networking site. The eventual applications of this work are widespread, though the purpose of this project is merely to analyze the success of a simple annotation system written using Ajax techniques and applied to a browser environment. This project opens doors to research in the
areas of semantic and standardized markup of electronic text, source control and mechanisms for updating annotated libraries, and community-based linking and reference to other annotations, selections, or media.

GOING FORWARD

The following chapters will expand on the technical and social ideas laid out in the introduction and discuss their relevance and relationship to this project, specifically refereed sources relating to the idea of annotation and text analysis, along with technical literature relating to browser-based software development, e.g. JavaScript, HTML, CSS, and the Document Object Model (DOM). As should accompany any engineering endeavor, a thorough analysis of the social and ethical implications of this project will be presented. The bulk of this paper discusses the materials and methods of the project, along with an analysis and interpretation of results. Finally, the interconnection of this project with other aspects of humanities computing provides an opportunity to discuss research opportunities going forward, including related technologies, software improvements and suggestions, and general thoughts on the state of electronic text analysis.
II. LITERATURE REVIEW

The humanities community has been excited about the advent of electronic text—and the analysis possibilities inherent with the backing of a computer—since the widespread adoption of the Internet. As this paper has already demonstrated, a pluggable web framework built with Ajax technology will provide a solid foundation for text tool development and deployment via a web browser. This section of the proposal briefly reviews academic thought on the development of electronic text tools, specifically annotation. Additionally, it will provide an overview of Ajax methodologies and technologies.

ANNOTATION IN ELECTRONIC TEXT

The problem at hand revolves around the analysis of digital text. With more than a million dissertations already translated to digital versions and the increase in the amount of research both going online and using digital tools, shortcomings in the use of text analysis tools have become apparent. As if to compound the situation, the state of electronic text is far behind the features desired by academics and researchers. Despite such shortcomings, the computer has, since the early 1990’s, provided what Siemens refers to as “the natural companion to such electronic texts: useful, integrated text analysis software” (Siemens 91).

Siemens then goes on to define dynamic text as the combination of electronic text and incorporated textual analysis software, and discusses the idea that such tools need to be integrated, seamlessly, and should be offered via an interface that is intuitive and straightforward. The dynamic edition is then a scholarly text that implements a dynamic interaction with the text while maintaining traditional hypertext navigation. This dynamic
interaction could replace some of the traditional annotation via hypertext links and “transfer… the principles of interaction allowed by a dynamic text to the realm of the full edition” (Siemens 94). The Internet takes this interaction to a new level with the addition of networking, social computing, and standardized markup. Daniel Cohen believes that the research environment provided by the Internet is easy to use in that it “takes advantage of semantic bibliographic information on the Web” (Cohen 3).

In order to discuss annotation as it relates to humanities researchers, a basic understanding of John Unsworth’s idea of scholarly primitives is helpful. These primitives—“basic functions common to scholarly activity across disciplines, over time, and independent of theoretical orientation” (Unsworth 1)—form the basis of all higher-level scholarly systems and include, among others: annotating, discovering, referring, sampling, and comparing. These functions for projects, arguments, statements, and interpretations provide the basics of many existing text analysis tools; this project focuses primarily on the idea of annotation, though incorporates other aspects as a matter of necessity or as a byproduct of the application and users.

AJAX AND JAVASCRIPT LIBRARIES

A fundamental aspect of this project will be web page development with Ajax. “Ajax… turns the Web on its head, fundamentally altering the idea of what a Web page is supposed to be by changing user interactions with online content” (West 1). Ajax, coined by Jesse James Garrett in 2005, defines a set of client-server applications without the need for page refreshes, which adds a level of interactivity to web applications not possible with traditional hypertext. As Garrett stated in his article that began the Ajax revolution, “an Ajax application eliminates the start-stop-start-stop nature of interaction on the Web by
introducing an intermediary—an Ajax engine—between the user and the server” (Garrett). This process actually improves page responsiveness rather than diminishing it, despite the increased communications overhead, as the user is able to work during network activity and only small portions of the page are updated at any given time.

Ajax is a set of tools for agile web development rather than a specific piece of software or set of instructions, and as such is constantly evolving in terms of techniques, libraries, and applications. As Paulson notes, “Ajax applications take advantage of dynamic HTML, which consists of HTML, cascading style sheets, and JavaScript glued together with the document object model” (Paulson 14). The core technical component of an Ajax application is the JavaScript object XMLHttpRequest, which makes HTTP requests and receives responses quickly and in the background, without visual interruptions. West adds, “Ajax operates like a Portuguese man-of-war. It’s a colony of technologies that work together towards one goal: delivering content and processing data on the Web without the reload, reload, reload tango that we’re used to with Web-based applications” (West 1).
III. IMPACTS & ETHICAL IMPLICATIONS

As with any technology, the potential impact, both positive and negative, can be extensive. This project aims to prove that a web-based text-analysis application based on web 2.0 principles will be a boon to humanities researchers. Computers are able to compute text-analysis algorithms more accurately and more quickly than even well trained humans, and this project hopes to provide a solution that will free up researchers’ time for other ventures. This solution will be built on a platform that provides fast extensibility and explores academic avenues of web 2.0 applications. From an implementation standpoint, Ajax “helps to increase the levels of interactivity, responsiveness and user satisfaction” (Mesbah 1). This makes a more successful application, and one users are more likely to use simply because it provides a better experience.

Additionally, trends in software engineering are swinging toward thin clients, or software such as web applications that keep a large amount of information in a central location. This paves the way for this project to incorporate aspects of social computing, where users are able to share sources, results, or the discussion thereof. Social computing sites such as MySpace or Facebook may be seen as teen hangouts, but these online communities “are changing the fundamental way people share information and communicate. They are profoundly affecting the global economy, social interaction, and every aspect of our lives” (Wang 81). This connection of this tool to the network of professionals on the Internet accelerates the rate of progress in the humanities field via a connection to other researchers and their work.
The ethical concerns of a project whose primary focus is software revolve more around the use—or more appropriately, misuse—of that application. In the case of software to aid the field of humanities research, there exists an inherent warning about the overuse of computational resources; specifically, the concern that software will take away from the field rather than add to it. If a computer application transforms a process that is currently a manual process requiring scholarly expertise into a *pro forma* activity, then its addition to the field may be more a detriment to researchers than the aid it was intended to be. Richard Johnson even warns against the dangers of subjective language in a computer application, noting, “Any time there is communication between a computer and a human, the information presented by the computer has a certain style, diction, and tone of voice which impact upon the human’s attitude and response toward the software” (Johnson 447). As such, the project must maintain a neutral research to allow the researcher to interpret his or her own conclusions from the computer’s results.

Additionally, scholarly publications being utilized must be evaluated on their merits. Henk Moed suggests that researchers routinely answer questions—such as “How does one recognize a ‘good’ scholar? How does one recognize an ‘important’ scholarly contribution?” (Moed 500)—that a computer might struggle to answer. Moed goes on to define this approach to scholarship as bibliometric. A bibliometric method “aims at identifying characteristics of scholarly publications that can validly be assumed to reflect the ‘quality’ or ‘importance’ of a scholar or a scholarly work” (Moed 500). These bibliometric characteristics are, after a fashion, quantitative, but rely on a researcher’s intuition as much as the bottom line of an analysis worksheet. As this project incorporates scholarly analysis, it must not assume to determine the worth of the work in question or, if it does, be very clear about the steps it took to do so and give the user ample opportunity to make his or her own decisions.
This project must also take into account the electronic sources being utilized in the project, and recognize the potential for copyright or other abuse by users, should such prohibitions be applicable. Additionally, as this piece of software performs basic data mining, it may also take into account privacy concerns, should such a tool system be used on a data source containing personal information. However, this project breaks no barriers in the field of data mining, and as such the tools developed by this project will probably be considered rudimentary by the standards of that industry. Finally, one must take a step back and recognize far-reaching aspects of computer software. Its impact on the environment is but one example, as there exist “500 million obsolete computers in the U.S., and even today, only 10% of unwanted and obsolete computers are recycled” (Pratt 27). If engineers focus on software as the solution to many problems, this situation will only be exacerbated. At this time, software is a good solution to such problems, but both hardware and software manufacturers must remain good environmental citizens and take steps to improve recycling and other “green” efforts.
IV. MATERIALS & METHODS

Fundamentally, this project is a technical one, and while the motives and analysis of the thesis depend on more qualitative indicators, at the core lays several thousand lines of HTML, CSS, JavaScript, and PHP. The challenges and obstacles encountered during the development phase of the project—as well as the solutions and the choices made by the programmer—comprise the basic deliverable of the explorations into the area of browser-based humanities text analysis and annotation.

CLIENT-SIDE FUNCTIONALITY

This project is designed to be delivered over an Internet connection and used in a standards-compliant browser such as Mozilla Firefox or Apple Safari. With any web application, there are two distinct parts of the system: the front end (client-side), or the markup and semantic layout presented to the user; and the back end (server-side), or the processing power of the application, usually paired with a relational database. This application is based on Ajax methodologies, which adds more functional capabilities to the client side of the application, creates different avenues of information transfer, and further emphasizes the split between the two application halves. Nevertheless, each side is important and contains important elements that contribute in different ways to the system as a whole.

*Vanilla JavaScript, DOM Manipulation & AJAX Communications*

Simply put, Ajax methodologies and technologies allow web pages to load information “behind the scenes” and update the displayed information without the need to
for the user to wait for a complete page refresh. This project relies heavily on Ajax, and as such, a majority of the visual effects and GUI elements are created and defined using JavaScript alone. Without JavaScript, this application would be comprised of a very simple set of Create, Edit and Delete (CrUD) commands—not to mention simply not working with even a modicum of usability.

At the core of this system are two fairly simple code snippets: one to gather information about the currently selected text, including starting and ending anchor node references and character offsets and the selected text itself; and one to send and receive messages asynchronously to a PHP/MySQL back end. On execution of the system, the selected text is converted to a W3C Range object. On his website, quirksmode.org, Peter-Paul Koch explains Range as “an arbitrary part of the content of an HTML document. A Range can start and end at any point, and the start and end point may even be the same (in which case you have an empty Range).” For example, a Range may be, in colloquial terms, the text between the fifteenth letter in the second paragraph and the sixth letter of the fifth headline.

But this information is not enough; the scientific adage regarding affecting the environment one observes is as true in cyberspace as it is in the African jungle. Specifically, in order to annotate a HTML document, you must first change it, adding markup to define the annotation comment and visually delineate the selected text. This is all well and good for the first annotation added to a system, but for any additional annotations, the Range object will reflect the updated markup, rather than the original document as retained in the database. The key here is the nodes returned via the Range object: both are text nodes, or nodes that hold textual content and no HTML. Every piece of text in a HTML DOM is a text node, and as markup is added after page load, text nodes are split and moved up or down in the
DOM tree. Thus, to find boundary points that are meaningful to the stored document, we walk up the tree to find the first permanent element, or element that exists in the repository. We then count characters in text nodes between the beginning of this element and the node/offset pair returned via the Range object, giving us information that is universal to the repository, rather than being tied to the current DOM structure of the annotated document.

From here, it is a simple matter of informing the server of the information associated with the new annotation via a XMLHttpRequest object—the JavaScript object responsible for asynchronous HTTP requests. The system sends all the information via a GET request, meaning encoded in the URL rather than sent as a separate header, to a PHP file, handling errors successfully. We use GET across the board rather than a mixture of GET and POST for simplicity; the World Wide Web Consortium (W3C) recommends using GET if the interaction is more like a question, and POST of the interaction is more like an order or changes the state of the resource, in this case the database. As such, ideally the application would send a GET request when getting existing annotations, and a POST request when adding, editing, or deleting an annotation. The difference is largely semantic, and makes little difference in a beta application, though the XMLHttprequest handles GET and POST requests very similarly. (Jacobs)

This exchange of information is handled on the client side by separate JavaScript functions. The remainder of the JavaScript associated with this system is largely visual, ranging from the simple—showing or hiding the annotation bar—to the more complex—wrapping every applicable text node between two references with a span with an arbitrary className, changing the appearance of that text. Indeed, most of the visual changes and layout of the user interface are controlled via the markup and presentation of the document. Technical algorithmic explanations may be found in Appendix A.
In semantic HTML, the markup and presentation of a HTML document are two separate but integral aspects relating to the display of information or content. The markup in this case refers to the code of the document, or the HTML elements and attributes that define the beginning, end, parent, and child elements of the DOM. On the other hand, the presentation defines how the document looks, ranging from font color to the location and size of objects in the browser. An important aspect of semantic markup is the separation of content and presentation, accomplished through the use of CSS, as the model and view (in the MVC sense) should be two separate but equal entities, not muddled together. From a utilitarian standpoint, concise, presentation-free markup allows for faster page loading, simpler system-wide stylistic changes, and better document degradation to text-only browsers or screen readers.

This system separates content and presentation and uses a combination of CSS classes to define much of the visual impact of the GUI. In practice, this means that the elements associated with a comment are physically located right before the selected text delineating the annotation, but are visually positioned on the right side of the browser window, lined up and colored according to user, leaving the reference document visually complete, with the addition of colored underlines denoting which text has been annotated. Much of these visual effects are accomplished with fairly simple CSS statements. Different attributes were applied to different classes, and JavaScript was used to change text appearance on the fly simply by changing the className of one or more HTML elements.
Usability

In any computer system, usability is an important consideration, and web-based applications are no exception, despite the prevalence of truly miserable computing experiences to be found on the Internet. Usability is not the explicit goal of this system, but there are several visual cues and helpful error messages to guide a user with no knowledge of the system other than, “It’s supposed to let me mark up this document.” Tabs and small images indicate that the sidebar may be shown or hidden, and the add annotation button has a fairly large area, as well as tool tips alerting the user to its functionality and keyboard shortcuts.

For a screenshot and description of this applications user interface, please see Appendix C.

SERVER-SIDE PROCESSING

The server-side of this system is responsible for permanent storage of data, as well as more complete and more secure processing of client information, not to mention data validation and the latter half of the Ajax communication family. This section focuses on details of the server system, specifically: database schemas, user authentication, and XML markup. These considerations take place in several PHP files, some of which deliver a document directly, others which accept a variety of GET data points, act accordingly, and return output as semantic XML.

Database Schema

All information in the system is stored in a MySQL relational database in three tables: annotations, authors, and sources. The annotations table contains the information on
individual annotations, including node/offset information as discussed above and comment
time and content. Each annotation also contains a foreign key to the authors table, indicating
the author of the comment, and the sources table, indicating the document to which the
annotation is associated. The authors table contains information on each user, including
display name, e-mail address, and login information for authentication. Finally, the sources
table is a version-controlled repository of all source documents in the system, containing—at
this time—merely the semantic markup of each document. Further database schema
information may be found in Appendix C.

Sources are stored in the database for two reasons: to provide a repository for
information—making adding or removing sources simple—and to provide a relative link
between sources and annotations. Since annotations—specifically the Range information to
denote a comment’s anchor in the text—depend on the structure of the source to remain
exactly the same, a problem arises, as updating or editing a source becomes a necessity. This
feature was not addressed by the author in this project, as outlining and algorithmically
finding the differences between two document versions is a project in of itself, not to mention
the migration of existing annotations to the new version.

User Authentication

User authentication is needed on the server side to ensure, at this time, that the which
whom to associate each annotation. In the future, authentication may allow only those
associated with a project to view source documents and add annotations and other privilege
considerations. The authentication object included in this project was written before the
thesis project, and retains user login information via cookies and PHP sessions, allowing login
once a session or “remembering” login information on non-shared computers via browser
cookies. Authentication forces users to identify themselves, allowing the system to group
annotation together by user, and eventually—in a collaborative environment—for
researchers to converse with each other and know the other’s academic credentials.

XML Markup

In utilizing Ajax, it is important to note that all communications between client and server asynchronously are XML documents. The XML utilized in this project is very simple, comprised of, in most cases, annotations and associated information.

For information about the source code of this application and a test version of the system, please see the information in Appendix D.
V. RESULTS & DISCUSSION

This system was built using some of the latest techniques in web application development, including but not limited to asynchronous requests using JavaScript to accomplish an interactive interface in a web browser to allow for shared annotations on a source. While this system provides a new an intuitive way to mark up and share thoughts on a piece of electronic text, its usability and real-world applications must be examined to determine the success or failure of such an approach.

To assess the utility of this program from an end-user point-of-view, it was distributed it to several people—beta testers, if you will, though their expertise in humanities computer was valued over their ability to technically evaluate software. Their interests lie at the intersection of the humanities and technology, and this system would be relevant to them as a tool in their daily workflow, as an example of up-and-coming technology, or an idea they may integrate in their own projects. They include:

- Bethany Nowviskie—Director, Digital Research and Scholarship, University of Virginia Library, University of Virginia
- Paul Vetch—Senior Analysis / Research Fellow, Computing in the Humanities, Centre for Humanities, King’s College, London
- Thomas Horton—Associate Professor of Computer Science, School of Engineering and Applied Sciences, University of Virginia

Their comments and input provide an insight into the multitude of possible uses, as well as the many ways this application can be improved going forward. This section includes
comments from these contributors, and combines their thoughts with those stemming from
the author’s own experience in writing and testing this application.

SHORTCOMINGS

Most of the shortcomings noted by the beta testers of this application were feature
requests. In terms of the features implemented in the first release of this software, the
platform cries out for more advanced tools to help wade through a multitude of comments on
the same source. In particular, the ability to filter by user or subject, as well as a method to
visually discern the relationship between a comment and a span of selected text.

From a platform standpoint, a browser provides a few quirks to overlook or work
around. First is the necessity of extraneous markup to the application, and the overhead
involved in keeping track and modifying that markup to provide the user with a presentation
that is friendly and simple. The physical addition of markup an extension of the fact that the
application runs in an environment—the browser—designed to view documents and nothing
more. As such, the tools associated with this system are considered to be part of the
document, and must be manually differentiated as separate from the source. A traditional
desktop application could differentiate between the source, tools, and comments, and better
provide this specific functionality to the user. Of course, the development of a desktop
application would come at the cost of the networking and plug-in potential a web application
brings to the table, not to mention open source initiatives and platform independence and the
fundamental question of this thesis.
SUCCESES

The success associated with this project is the completed project itself, in that the idea is applicable and can be implemented in the browser environment as it stands today. This success is reinforced by the fact that the tool is intuitive and user-friendly. Paul Vetch noted that the “attention to interface details all make for a trust-inspiring and self-explanatory user experience which is important with tools like this, since they are not always obvious.” Very clearly, this tool has a wealth of uses beyond that of simple humanities computing, and all reviewers mentioned the addition of comment threading to further a collaborative atmosphere. The potential uses of such a method of marking up electronic text seem to be broad, with a wide range of possible applications and audiences, ranging from personal use to widespread community adoption. “In addition to the pedagogical and collaboration (peer review) uses, I could also see using this for my own note-taking on electronic resources,” wrote Bethany Nowviskie.

Additionally, Ms. Nowviskie alerted the author to CommentPress—“an open source theme for the WordPress blogging engine that allows readers to comment paragraph by paragraph in the margins of a text” (CommentPress). CommentPress provides a similar service as the application presented here, though obviously ingrained in the blog system around which it was designed. CommentPress, however, does not provide the granularity of data control as the project presented here, but serves as a real-world emphasis on the popularity and possibilities of such an annotation system.
VI. CONCLUSION

This project sought to determine the viability of an annotation system built using Web 2.0 technologies and methodologies and delivered in a shared browser environment. Building a prototype of such a system proved that such an approach was not only viable, but also successful. A large majority of the work on this project took place on the client side of the system via JavaScript and DOM manipulation, and it appears that, with few exceptions, that the state of browser technology is such that the user interface requirements of such a system could be met successfully. The advent of Ajax methodologies took the project beyond the scope of a web page and into the realm of a true rich application, where user actions are met with immediate response without the flash of changed content or the delays associated with complete page refreshes.

INTERPRETATIONS

The results of this project are complete, but may be extended further. As a baseline, the production of the system in question proved unequivocally that such a system could, indeed, be built. In doing so, the logical leap of providing annotation functions via a browser environment and the social computing context that follows was validated, and the application developed during the course of this thesis provides a foundation for extension possibilities. The application discussed in this paper has widespread commercial and academic use, as the idea of electronic annotation is a popular one. Paul Vetch had this to say: “I think it's fair to say that web based annotation as an interaction model is increasingly well understood based on general patterns of usage on the web at the moment, and I think it would be reasonable to
expect ‘general’ users (i.e. casual internet users) to respond well conceptually to this kind of tool and find it to be ‘usable’.”

RECCOMENDATIONS & FUTURE RESEARCH

As mentioned above, this project provides a foundation of annotation tools in a web-based application; the limits of this project reach far beyond the scope of this project. As such, the possibility for future research in this area is large, including: analysis of humanities-based social networking; electronic text algorithms to be applied pragmatically to selected text; application of this technology to other, “richer,” media, including images and video files; and applications of other Unsworth scholarly primitives. Additionally, as a result of feedback gathered from the initial set of users, there exists a growing list of feature requests, included, but not limited to: tags on annotations, JavaScript framework (e.g. JQuery) code base, user profile pages, user or annotation permissions for editing or viewing, image-based annotation, comment highlighting on mouseover, visual relationship between selection and comment, condensed comment view, and threaded annotation-based discussions.

Additionally, the selection and DOM manipulation portion of this project can technically stand alone, and as such may make an interesting foundation for stand-alone tools aimed more at users who will use the feature set by themselves. For example, this functionality could be built into a bookmarklet, or a browser bookmark that executes JavaScript code instead of opening a URL, to allow the user to highlight and comment on any web page he or she so desires, storing the information in a cookie on the user’s machine, rather than in a remote database. A bookmarklet implementation is merely one idea among many, and a bright developer could apply this concept many different ways going forward.
BIBLIOGRAPHY


APPENDICES

APPENDIX A: TECHNICAL ALGORITHMIC EXPLANATIONS

The basis of an annotation is a W3C Range, which contains a reference to the starting HTML text node, and the offset to the starting caret position from the start of the text node; it also contains the same references to the ending node and offset. In general, this is a simple process, both to create a Range from a user selection and re-create a range based on a stored XPath and offsets. However, the adage of changing the environment one observes is an applicable one in this case. Once the DOM structure is changed, these changes must be taken into account.
A-1: Tree Walking

The system needs to do a conversion between Range nodes/offsets, which need to be based on the text nodes as they appear in the HTML, and XPATHs and offsets stored in the database. From a text node, the system walks up the tree using .parentNode until it finds a permanent node; in doing this the system can also check and make sure the user isn’t doing anything illegal, such as trying to annotate an annotation. Then, the system must walk the tree again, finding every permanent text node between the start of the permanent node and the selected text node, counting contents to create a permanent offset to be stored in the database.

Moving from permanent to document nodes—the reverse of the above—is a similar process. The system simply walks through every permanent text node contained in the permanent (stored) node, keeping track of content lengths, which results in a text node and a “real” offset given that text node.
A-2: Deleting an Annotation

On deleting an annotation, the system gathers the ID of the annotation from a position relative to the delete button that was clicked. The system previously named all markup associated with each annotation based on its ID, so it can then create a reference to everything that must deleted or updated. An annotation has more than one markup element to control style and presentation: for example, the beginning caret, the ending caret, and all underlines are controlled with HTML elements. Once references to these nodes are acquired, the system deletes them from the DOM. But, it’s not that simple. For the spans surrounding text to provide underlines, etc., the system must remove the tags from the markup, rather than the DOM node (which includes its contents). Simply, the DOM needs to go from this:

![DOM Node Example](image)

To this:

![DOM Node Example](image)
# APPENDIX B: DATABASE TABLE DESCRIPTIONS

<table>
<thead>
<tr>
<th><strong>thesis_sources</strong>: this table holds all the sources currently in the system</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>id</strong></td>
<td><strong>int</strong></td>
</tr>
<tr>
<td><strong>contents</strong></td>
<td><strong>blob</strong></td>
</tr>
<tr>
<td><strong>title</strong></td>
<td><strong>text</strong></td>
</tr>
<tr>
<td><strong>author</strong></td>
<td><strong>text</strong></td>
</tr>
<tr>
<td><strong>year</strong></td>
<td><strong>int</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>thesis_authors</strong>: this table holds information on all users in the system</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>id</strong></td>
<td><strong>int</strong></td>
</tr>
<tr>
<td><strong>name</strong></td>
<td><strong>text</strong></td>
</tr>
<tr>
<td><strong>email</strong></td>
<td><strong>text</strong></td>
</tr>
<tr>
<td><strong>password</strong></td>
<td><strong>text</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>thesis_annotations</strong>: this table holds all the annotations/comments currently in the system</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>id</strong></td>
<td><strong>int</strong></td>
</tr>
<tr>
<td><strong>source_id</strong></td>
<td><strong>int</strong></td>
</tr>
<tr>
<td><strong>author_id</strong></td>
<td><strong>int</strong></td>
</tr>
<tr>
<td><strong>timestamp</strong></td>
<td><strong>int</strong></td>
</tr>
<tr>
<td><strong>comment</strong></td>
<td><strong>text</strong></td>
</tr>
<tr>
<td><strong>start_element</strong></td>
<td><strong>text</strong></td>
</tr>
<tr>
<td><strong>end_element</strong></td>
<td><strong>text</strong></td>
</tr>
<tr>
<td><strong>start_offset</strong></td>
<td><strong>int</strong></td>
</tr>
<tr>
<td><strong>end_offset</strong></td>
<td><strong>int</strong></td>
</tr>
</tbody>
</table>
APPENDIX D: SOURCE CODE AND TEST APPLICATION

The decision was made not to release the source code for this application as an addendum to this thesis. However, a test version of the system described in this thesis will be maintained for as long as possible at the following URL, which includes user-viewable JavaScript files that make up the bulk of the features described in this paper:

http://basseq.com/thesis
Pluggable Ajax Frameworks Using Product Family Development: Hypertext-Based Text Analysis Tools For Electronic Scholarship

Submitted by

John Whitte

Computer Science

STS 401
Section 13 (12:15 p.m.)
November 8, 2007

On my honor as a University student, on this assignment I have neither given nor received unauthorized aid as defined by the Honor Guidelines for Papers in Science, Technology, and Society Courses.

Signed: ________________________________

Approved 

Date ________

Technical Advisor: Thomas B. Horton

Approved

Date ________

Science Technology & Society Advisor: Patricia C. Click
# TABLE OF CONTENTS

Informative Abstract.................................................................................................................................................. iii

I. Rationale & Objectives........................................................................................................................................... 1
   Social Context..................................................................................................................................................... 1
   Technical Context............................................................................................................................................... 2
   Objectives ......................................................................................................................................................... 3

II. Review of Technical Literature ...................................................................................................................... 4
   Analysis of Electronic Text............................................................................................................................... 4
   Agile Web Frameworks with Ajax .................................................................................................................... 5
   Product Family & Plug-In Engineering ............................................................................................................ 6

III. Statement of Project Activities ..................................................................................................................... 8
   Activities ......................................................................................................................................................... 8
   Schedule ......................................................................................................................................................... 10
   Personnel ....................................................................................................................................................... 10
   Resources ....................................................................................................................................................... 11

IV. Expected Outcomes......................................................................................................................................... 12

V. Preliminary Analysis of Ethical Implications ............................................................................................... 13
   Bibliography................................................................................................................................................... 14
   Required Appendices ....................................................................................................................................... 16
   Budget & Equipment List ............................................................................................................................... 16
   Biographical Sketch of Student .................................................................................................................... 16
INFORMATIVE ABSTRACT

This paper outlines a project to develop a framework for humanities-based text analysis tools using web 2.0 approaches including Ajax and agile web frameworks. From a software engineering viewpoint, individual text tools will be regarded as a product family, or a series of software products that share a set of requirements and specifications. This project should advance the field of humanities computing, specifically related to literary analysis, as well as provide an insight into the possibilities of web-based computing. A variety of activities have been identified whose completion is vital to the ongoing and final success of this project. To judge the accomplishment of this project, a set of goals, objectives and expected outcomes have been outlined, and a pool of resources and personnel have been identified to aid the author in the completion and success of this project. Finally, this paper addresses preliminary science, technology, and society concerns associated with the use—or more appropriately, misuse—of this project’s deliverable. This paper should provide a solid foundation in the pursuit of research, development, and analysis of the proposed software suite and the completion of the thesis, slated for Spring 2008.
I. RATIONALE & OBJECTIVES

More than seven centuries of written word have left humanity with a wealth of knowledge and information. The advent of the computer—and, later, the Internet—helped humanities researchers analyze text faster and more effectively; the use of text tools to discern patterns, similarities, and usage provide those studying such literature a wealth of statistics. These data are used to provide insights into our language and culture, comparisons between genres and authors of literature, and an understanding of our literary progression from the works of Aeschylus to the words being penned or typed at this instant. The objective of this project is to research and create a working example of a collection of contextual operations to be performed on a subset of a repository of information. The motivation for this project lies in the author’s appreciation of classic literature as well as a realization of need for textual research techniques utilizing the power and accuracy of a computer. Additionally, an application platform such as the one suggested here would be applicable in research areas ranging from literary investigations to commercial data mining; the amount of material available on the web necessitates tools to accurately draw logical conclusions from raw data.

SOCIAL CONTEXT

Ray Siemens—Canada Research Chair in Humanities Computing at the University of Victoria and a member of the Text Analysis Portal for Research (TAPoR) project—has noted that electronic text, while useful, does “not yet offer text-analysis features commensurate with expectations even at the beginning of the last decade” (Siemens 2). Though research and analysis, the project aims to apply text tools in a completely new
environment, and hopefully address such expectations. The possibilities for computer-aided research in regard to electronic text are large, with application ranging from traditional humanitarian research to text mining and data analysis.

Additionally, software engineering as a profession is quickly moving toward the delivery of applications across a network, often with a shared code base. These applications, entirely Internet-based, allow users to access a variety of data, interact with other product users, and store client-specific settings in such a way that moving between workstations requires no user-specific profile or data acquisition. From a deployment view, such applications are easier to support—as patches and updates are applied across a customer base instantly—and run almost regardless of normal hardware and operating system (OS) metrics, current browser quirks notwithstanding. Advances have been made to bring previously desktop-specific features—such as interactivity and rich media—to web applications. Through the software investigations in this project, the author will examine applications of traditional software engineering techniques to more dynamic combinations of client-server architecture.

TECHNICAL CONTEXT

The author’s exploration of this subject area will include the creation of a web-based collection of text tools. These tools will provide a set of operations to be performed on a repository of information based on author, work, chapter, paragraph, line, and other metrics. This collection of tools makes up a product family, defined by Carnegie Mellon’s Software Engineering Institute as “a set of software-intensive systems that share a common, managed set of features” (SEI). These tools will share the fundamental ability to examine a textual input combined with several other parameters to provide an analysis of the document in
question; specifically, the project will incorporate an Application Program Interface (API) and a pluggable development environment to facilitate the creation of additional tools in the product family with little overhead cost. This task will be accomplished with the aid of a web framework, as software suites such as Ruby on Rails are designed for the rapid development of web applications with an emphasis on the simple addition of features or subroutines. This assortment of tools will be presented to the user with a logical and interactive user interface utilizing semantic HTML and Asynchronous JavaScript and XML (Ajax) requests.

Research into the fundamentals of this project will provide a solid foundation for future work, with particular emphasis on agile web framework development, text processing and data mining techniques, and product family development practices. In the area of humanities research methods, papers by John Unsworth and others will be provide what Mr. Unsworth refers to as “scholarly primitives,” or a set of functions and approaches that “form the basis for higher-level scholarly projects, arguments, statements, [and] interpretations” (Unsworth 1). The source of such material includes the University library system, project such as TAPoR and The University of Virginia’s Institute for Advances Technology in the Humanities (IATH). This project will combine client-side interactivity via Ajax with agile software engineering techniques, a combination not widely explored in today’s academic tools or thinking.

OBJECTIVES

The following list of objectives defines the goals of this project, as well as additions to the fundamental knowledge of agile web development frameworks, product family engineering, and humanities text analysis tools needed to produce the desired outcome.
1. Design and develop a set of tools sharing common requirements as well as similar inputs and outputs combined with a parent controller application.

2. Analyze the success of implementing such a tool set as a web framework utilizing Ajax and web 2.0 techniques.

3. Fill the gap of text-analysis features as defined by John Unsworth, if possible.

4. Uncover and discuss the ethical concerns of such a project and its related applications and their impact on the technical community as well as the world at large.

II. REVIEW OF TECHNICAL LITERATURE

The humanities community has been excited about the advent of electronic text—and the analysis possibilities inherent with the backing of a computer—since the widespread adoption of the Internet. As this proposal has already demonstrated, a pluggable web framework built with Ajax technology will provide a solid foundation for text tool development and deployment via a web browser. This section of the proposal briefly review academic thought on the development of electronic text tools. Additionally, it will provide an overview of product family engineering as applied to application plug-in development, as well as Ajax and web framework basics.

ANALYSIS OF ELECTRONIC TEXT

The problem at hand revolves around the analysis of digital text. With more than a million dissertations already translated to digital versions and the increase in the amount of research both going online and using digital tools, shortcomings in the use of text analysis
tools have become apparent. As discussed in Section I, the state of electronic text is far behind the desired features of electronic text analysis tools. Daniel Cohen believes that the research environment provided by the Internet is easy to use in that it “takes advantage of semantic bibliographic information on the Web” (Cohen 3). However, he also remarks on the difficulty of abstracting the traditionally manual process of text analysis to a computer, commenting that judgments on texts, even informal, are “often distillations of years of education, experience, and thought” and “remain, for the most part, in the realm of what may be called professional folk wisdom, passed along from one person… to the next” (Cohen 1).

Ray Siemens observes that despite such shortcomings, the computer has, since the early 1990’s, provided “the natural companion to such electronic texts: useful, integrated text analysis software” (Siemens 91). Siemens then goes on to define dynamic text as the combination of electronic text and incorporated textual analysis software, and discusses the idea that such tools need to be integrated seamlessly, and should be offered via an interface that is intuitive and straightforward. As mentioned above, John Unsworth has listed a series of “scholarly primitives” for projects, arguments, statements, and interpretations. These include: discovering, annotating, comparing, referring, sampling, illustrating and representing; they provide the basics of many existing text analysis tools.

AGILE WEB FRAMEWORKS WITH AJAX

A fundamental aspect of this project will be web page development with Ajax. According to Jessmyn West, “Ajax… turns the Web on its head, fundamentally altering the idea of what a Web page is supposed to be by changing user interactions with online content” (West 1). Ajax, coined by Jesse James Garrett in 2005, defines a set of client-server
applications without the need for page refreshes, which adds a level of interactivity to web applications not possible with traditional hypertext. As Garrett stated in his article that began the Ajax revolution, “an Ajax application eliminates the start-stop-start-stop nature of interaction on the Web by introducing an intermediary—an Ajax engine—between the user and the server” (Garrett). This process actually improves page responsiveness rather than diminishing it, despite the increased communications overhead. Figure 1, below, illustrates the difference between traditional web architectures versus the Ajax model.

Figure 1- Comparison of web application models.

Ajax is more of a set of tools for agile web development rather than a specific piece of software of set of instructions, and as such is constantly evolving in terms of techniques, libraries, and applications. Current “Ajax applications take advantage of dynamic HTML, which consists of HTML, cascading stylesheets, and JavaScript glued together with the document object model” (Paulson 14). The core technical component of an Ajax application is the JavaScript object XMLHttpRequest, which makes HTTP requests and receives responses quickly and in the background, without visual interruptions. West notes that “Ajax operates like a Portuguese man-of-war. It’s a colony of technologies that work together
towards one goal: delivering content and processing data on the Web without the reload, reload, reload tango that we’re used to with Web-based applications” (West 1).

PRODUCT FAMILY & PLUG-IN ENGINEERING

From a software engineering perspective, a large challenge associated with the development of this project lies with the creation of the individual tools and the common requirements associated with them. This collection of tools may be thought of as a product family, even though such a representation is usually reserved for a larger series of software applications, usually at the enterprise level. For the sake of analysis and discussion, this project will address the requirements and specifications of the individual plug-in tools as such a family. This approach makes sense, as the basis of product family engineering revolves around strategic reuse of business and technical strategy. Software product line deliverables pertain to market strategy and an application domain, sharing software architecture in order to manage requirements and complexity. The bottom line of product family engineering is that “strategic software reuse through a well-managed product line approach achieves business goals for efficiency, time to market, productivity, and quality” (Clements 82).

The project outlined in this proposal will tie together these three research areas, combining the analysis of electronic text, agile web frameworks with Ajax, and product family and plug-in engineering together in a new and innovative way. The application to be delivered as part of the final thesis will explore new and different avenues in respect to all three fields.
III. STATEMENT OF PROJECT ACTIVITIES

This section will discuss the activities to be undertaken in the research and their relationship to the overarching objectives and goals. This section will also outline the sequence and timing of these events, establish collaborators, and describe the resources this project will utilize.

ACTIVITIES

A series of activities must be undertaken to complete this project. Some activities are sequential, while others may be performed in parallel with other tasks. This section will discuss these tasks as well as contingency plans and the time allocated to each task.

Pre-Implementation

1. Conduct background research. Research will be undertaken to determine the state of the community at large and to identify the aspects of electronic text analysis algorithms highly regarded or desired by humanities researchers. In all likelihood, this will be an iterative process as additional metrics are established, but will take an up-front time of two weeks.

2. Gather requirements. Part of the background research will be determining the aspects of the application’s requirements. The basics of this document will be provided by technical advisor Tom Horton, and will be completed by the author during the last week of the research phase.
Implementation

1. Design. From the requirements document, a specification document will be created establishing the specifics of the project. Part of this phase will include prototyping of basic functionality related to the pluggable interface, et cetera. The time frame for this activity is two weeks.

2. Code. Most of the time spent on this project will be the implementation of the application outlined by this proposal. As with typical software engineering processes, this activity will be iterative as additional risks are identified, mitigated, and incorporated into the design of the application. This will take one month.

3. Collect data. Towards the end of the coding activity, we will need access to real electronic text sources. Other collaborators will provide these sources, but given communication and turnaround time, a week has been allocated to this task.

Analysis & Wrap-Up

1. Test. User acceptance testing to test the accuracy of the implementation as well as the usability of the implementation itself will be undertaken. This task will be completed with the aid of Tom Horton and will take three weeks.

2. Analyze. The success of this project and the application developed during the implementation phase is determined by the usability and accuracy of the system. To do this, we will compare the system to the requirements and identify accomplishments and shortcoming. This task will be undertaken, again, with Tom Horton and will take two weeks as part of the testing phase.

3. Write thesis. As a result of the steps above, a thesis paper will be written outlining the steps and results of each phase, as well as an analysis of overarching concerns
related to this project and an overview of existing academic research. This task will be undertaken towards the end of the project, and will be finished by the third or fourth week of March 2008.

SCHEDULE

As discussed in the Activities section, some activities may be performed in parallel, while others are dependent on the completion of other activities. Figure 2, below, presents these activities in a visual representation to show the path to project competition as well as task dependencies and schedule.

![Gantt chart of project activities.](image)

**PERSONNEL**

The network of collaborators for this project is small, but those involved are heavily engaged. The author will provide most of the work on this project, ranging from research to programming and the final analysis. Technical advisor Tom Horton will provide the requirements document for the end product, as well as provide support in the gathering of
electronic text and other resources. Professor Horton received his Ph.D. from the University of Edinburgh, Scotland, in 1986 and has been an Associate Professor of Computer Science at the School of Engineering and Applied Science at the University of Virginia since 2001. His research interests include text processing and humanities computing, as well as domain engineering for developing reusable components, tools, and environments. He is a lead investigator on the Nora Project, a multi-institution research project designed to explore data mining and visualization in digital literary libraries.

Science, Technology and Science advisor Patricia Click will help address the social and ethical context of this project, as well as aid in the organization and communication of ideas in the written paper. Professor Click received her Ph.D. in American History from the University of Virginia, and is currently an associate professor in the Science, Technology, and Society department in the School of Engineering and Applied Science at the University of Virginia. Her research, according to the STS webpage, “aims to add value to the field of traditional history as much as it aims to add to the field of [Science, Technology and Society].”

**RESOURCES**

The resources involved with this project are fairly minimal in that no specialized equipment or lab time will be needed to complete the objectives. The project does require a web server with a server-side language installed. The author maintains both a local and remote web server running Apache 2 and PHP, which will be utilized for this project. Additionally, a database of electronic text, the format of which has not been determined, must be obtained. Worthy Martin of IATH has offered support in this area, and these leads will be pursued once requirements and specifications have been set.
IV. EXPECTED OUTCOMES

Upon submission of the thesis, a working application consisting of three or four tools and a fully functional interface will have been developed. Future work may be undertaken to explore additional accomplishments or missed requirements, if such a need arises. The project will be complete upon delivery and analysis of the software package discussed in the above sections, and success will be determined merely by the completion of these items. The risk of failure in this project is minimal, as the software merely tests what is possible and desirable in humanities computing. Even if analysis shows that this approach is not a viable one in regards to the issue of electronic text analysis, the project will be a success in that it has concluded definitely the feasibility, or lack thereof, of such an application.

The results of this project will not be quantitative, but rather qualitative, in that we will be determining the ease and applicability of using web 2.0 concepts in regards to humanities computing. Fundamentally, the aim of this project is to determine if this process works, and the ease of adding and exploiting the capabilities of both the computer and the Internet to advance electronic text analysis. The precise heuristics to determine what constitutes the viability of this project will be laid down during the requirements and specification phase of this project. Regardless of the outcome of this project, at the very least it will contribute knowledge to the field of humanities computing, and at best a solid software foundation for the development of future electronic text tools and an overarching application API.
V. PRELIMINARY ANALYSIS OF ETHICAL IMPLICATIONS

The ethical concerns of a project whose primary focus is software revolve more around the use—or more appropriately, misuse—of that application. In the case of software to aid the field of humanities research, there exists an inherent warning about the overuse of computational resources; specifically, the concern that software will take away from the field rather than add to it. If a computer application transforms a process that is currently a manual process requiring scholarly expertise into a pro forma activity, then its addition to the field may be more a detriment to researchers than the aid it was intended to be. One must also take into account the electronic sources being utilized in the project, and recognize the potential for copyright or other abuse by users, should such prohibitions be applicable.

Additionally, as this piece of software performs basic data mining, privacy is also a concern, should such a tool system be used on a data source containing personal information. However, this project breaks no barriers in the field of data mining, and as such the tools developed by this project will probably be considered rudimentary by the standards of that industry.
BIBLIOGRAPHY


REQUIRED APPENDICES

BUDGET & EQUIPMENT LIST

All development will take place on student-owned hardware, including but not limited to desktop and server environments. A small portion of this project may occur on University-owned resources to which the student maintains access, such as ITC computer labs or CS department servers. There is no budget for this project, as the student, University, or CS department will provide all resources.

BIOGRAPHICAL SKETCH OF STUDENT

John Whittet is a fourth-year Computer Science major with a minor in Engineering Business and an expected graduation date of May 2008. His interests lie in the future of the Internet to develop and deploy desktop-quality applications with the addition of networking, delivery, and information provided by the Internet itself. He boasts more than five years of web design and development experience in academic, non-profit, and commercial areas. Additionally, he also has both academic and commercial experience with traditional software development processes. Applicable academic classes include most notably CS 340 (Advanced Software Development), as well as the rest of the established BS Computer Science curriculum at The University. Professionally, John has worked for and with Fortune 500, non-profit, and private companies in the areas of defense, consumer products, and internet technology: all on projects related to the development of inter- or intranet applications.
Hypertext-Based Text Analysis Tools
For Electronic Text: Pluggable Web Frameworks
and Product Family Development

Submitted by
John Whittet
Computer Science

STS 401
Section 13 (12:15 p.m.)
September 18, 2007
Science, Technology, and Society Advisor: Patricia C. Click
Technical Advisor: Thomas B. Horton

On my honor as a University student, on this assignment I have neither given nor received unauthorized aid as defined by the Honor Guidelines for Papers in Science, Technology, and Society Courses.

Signed: _______________________________
More than seven centuries of written word have left humanity with a wealth of knowledge and information. The advent of the computer — and, later, the Internet — have helped humanities researchers analyze text faster and more effectively; the use of text tools to discern patterns, similarities, and usage provide those studying such text a wealth of statistics. This data is used to provide insights into our language and culture, comparisons between genres and authors of literature, and an understanding of our literary progression from the works of Aeschylus to words being penned or typed at this instant. My objective is to research and create a working example of a collection of contextual operations to be performed on a subset of a repository of information. My motivation lies in both an appreciation of classic literature and a realization of need for textual research techniques utilizing the power and accuracy of a computer. Additionally, an application platform such as the one suggested here would be applicable in research areas ranging from literary investigations to commercial data mining; the amount of material available on the web necessitates tools to accurately draw logical conclusions from raw data.

My exploration into this subject area will include the creation of a web-based collection of text tools. These tools will provide a set of operations to be performed on a repository of information based on author, work, chapter, paragraph, line, et cetera. This collection of tools makes up a product family, defined by Carnegie Mellon’s Software Engineering Institute as “a set of software-intensive systems that share a common, managed set of features” (SEI). These tools will share the fundamental ability to examine a textual input combined with several other parameters to provide an analysis of the document in question; specifically, I will create an API and a pluggable development environment to facilitate the simple creation of additional tools in the product family. I hope to do this utilizing a web framework, as software suites such as Ruby on Rails are designed for the rapid
development of web applications with an emphasis on the simple addition of features or subroutines. This assortment of tools will be presented to a user with a logical and interactive user interface utilizing semantic HTML and AJAX. Any development will take place on personal or University-provided servers.

Research into the fundamentals of this project will be undertaken, with an emphasis on agile web framework development, text processing and data mining techniques, and product family development practices. My own knowledge falls shortest in the area of humanities research methods; as such I will refer to papers by John Unsworth and others regarding what Mr. Unsworth refers to as “scholarly primitives,” or a set of functions and approaches that “form the basis for higher-level scholarly projects, arguments, statements, [and] interpretations” (Unsworth 1). The source of such material includes the University library system, projects such as the Text Analysis Portal for Research (TAPoR) and The University of Virginia’s Institute for Advanced Technology in the Humanities (IATH).

My own background in this area overlaps with, but does not encompass such a project. I have been active as a web software developer for close to six years, primarily for the commercial sphere, but also for the defense community with work on a content mining and search tool. On the technical side of the project, I am perfectly suited a developer with experience in web frameworks, HTML, and digital information. However, I do also have an interest in two potential uses for such a collection of tools, specifically commercial data analysis and literary examination. I find the creation and aggregation of information on the Internet fascinating, and a humanitarian approach to analyzing this “new” content would provide interesting comparative material to traditional works of literature — classic and contemporary — by authors ranging from Hawthorne to Wallace. This project provides the basis of a professional relationship with both advisors. Professor of Science, Technology, and
Society Patricia Click’s interest in history, specifically the “intersection of technology, culture, and society in nineteenth-century America” will provide a basis for the humanitarian side of this project. On the technical side, Thomas Horton, Associate Professor of Computer Science, provides a focus in software engineering and humanities computing and has expressed a genuine interest in learning more about this subject area himself.

Looking ahead to The Proposal and The Thesis, I foresee a wealth of research material for the literature review. I have discussed the keywords upon which I’ll focus my efforts and I am confident they will result in excellent background information for this research project — in all likelihood resulting new avenues of approach and ideas for this assignment. I look forward to exploring the social context of this project, specifically in regards to humanities researchers and text analysis. I also expect to explore other branches of research, e.g. web application frameworks and product family development, and their applications to this project. I foresee some ethical questions being raised regarding data mining and the privacy implications thereof, but while there are applications of data mining with this project, the emphasis is academic and commercial analysis and research. I see these assignments as an integral part of my thesis work, and look forward to exploring all issues, technical and social, in future papers.
BIBLIOGRAPHY

