MONK Project

Research Report by Salman Bakht, Pehr Hovey, and Kris McAbee

Summary

MONK, which stands for "Metadata Offer New Knowledge," is a single digital environment of literary texts that endeavors to make "modern forms of text analysis and text mining accessible to humanities scholars."[1] The metadata associated with any given document in the MONK environment ranges from data about individual words, to data about discursive organization, to bibliographic data. MONK offers the ability to read back and forth between these different levels of data and, therefore, to read as closely or as distantly as one wants. The current collection of texts in the MONK prototype consists of about 1200 works, including approximately 500 texts of various genres published between 1533 and 1625, alongside about 700 works of English and American fiction from about 1550 to 1923. Fundamentally, MONK assumes that operating through "coarse but consistent encodings across many texts in a heterogeneous document environment" offers significant scholarly benefit. The single environment that will bundle these operations for this large collection of texts will be housed at http://monkproject.org.

Description

The first phase of the MONK project was largely funded by a two-year (January 2007- January 2009) $1 million grant from the Andrew W. Mellon Foundation.[2] Participating Organizations include Academic Technologies (Northwestern University), the Center for Digital Research in the Humanities (University of Nebraska-Lincoln), the Graduate School of Library and Information Science at the University of Illinois, Human-Computer Interaction Lab (University of Maryland), Humanities Visualization (University of Alberta and McMaster University), Maryland Institute for Technology in the Humanities, and the National Center for Supercomputing Applications, University of Illinois).

The literary texts used in the MONK prototype are not in themselves critical editions, but they have seen some degree of editorial and scholarly intervention in the process of preparing and collecting them into the digital libraries from which MONK draws. Many works are from TCP (Text Creation Partnership) projects and, thus, have similar standards for accuracy, encoding, markup, and fidelity to the original. (See "The Text Creation Partnership," http://www.lib.umich.edu/tcp/.) The largest collections in MONK's corpora include:

1. The Text Creation Partnership (TCP), an archive of ~13,000 texts published between 1470 and 1700 in England)

2. The Chadwyck-Healey Eighteenth-Century Fiction Collection (~100 novels published between 1700 and 1780)

3. The Chadwyck-Healey Nineteenth-Century Fiction Collection (~250 novels written in England between 1780 and 1900

[1] "Notes towards a user manual of Monk," 2009, 5 Mar. 2009 <https://apps.lis.uiuc.edu/wiki/display/MONK/Notes+towards+a+user+manual+of+Monk>.

[2] "Mellon Foundation Awards MONK a Million," 2007, 8 Mar. 2009 <http://www.lis.uiuc.edu/oc/news/displaynews.html?source=e47099ClY2HMvdVKg7pMAw==&year=Tpfh-axYyS2d3HIF5XQQxw==>

4. The Chadwyck Healey Early American Fiction archive (~100 novels written in America between 1780 and 1851

5. The Wright American Fiction Archive (~1,000 American novels published between 1851 and 1875)[3]

MONK is built on the foundation of two preceding Mellon-funded initiatives: the Nora Project (http://www.noraproject.org) and WordHoard (http://wordhoard.northwestern.edu). The Nora Project is a multi-institutional and multi-disciplinary effort to create pattern-recognition and visualization software that facilitates the detection of patterns across digital archives, with a focus on 19th-century British and American literature. Meanwhile, WordHoard's concentration has been on providing a user interface that exposes morphological, lexical, prosodic, and narratological criteria of a deeply tagged corpora containing the entire canon of Early Greek epic in the original and in translation, as well as all of Chaucer, Shakespeare, and Spenser. Although these two projects may look very different to the end user, they share similar text-mining strategies. MONK combines the tactics of these two initiatives "to create an inclusive and comprehensive text-mining and text-analysis tool-kit for scholars in the humanities."[4]

Research Context

The MONK Project is of current interest to the fields of literary studies, humanities computing, as well as social computing at the intersection of these fields. As Claus Huitfeldt argues, "Textual scholars should not relate to markup technology as passive recipients of products from the computing industry, but rather be actively involved in the development and in setting the agenda, as they possess insight which is essential to a successful shaping of digital text technology."[5] The wide-reaching collaboration of MONK exemplifies this active involvement, in as much as it builds on the Nora Project's "guiding assumption" that humanities-oriented text-mining applications "should be prevocational in spirit—rather than vocational or merely utilitarian – and that the intervention and engagement of a human subject expert is not just a necessary concession to the limits of machine learning but instead an integral part of the interpretative loop."[6]

Because MONK's text-mining focus is on primary texts written in English, and the seed projects WordHoard and Nora both apply to literary texts, the project offers significant interest to various aspects of literary studies. MONK employs a "radical 'divide and conquer' strategy," offering metadata that ranges across bibliographical, lexical, and organizational data.[7] Bibliographic and orthographical variance data potentiates outcomes for history of reading and library studies,

---

[3] "Notes towards a user manual of Monk," 2009, 5 Mar. 2009 <https://apps.lis.uiuc.edu/wiki/display/MONK/Notes+towards+a+user+manual+of+Monk>.

[4] "MONK (metadata offer new knowledge," intute: best of the web, 8 Mar. 2009, <http://www.intute.ac.uk/artsandhumanities/cgi-bin/fullrecord.pl?handle=20080415-09211422>

[5] Carl Huitfeldt, "Scholarly Text Processing and Future Markup Systems," 4 Mar. 2009 <http://computerphilologie.uni-muenchen.de/jg03/huitfeldt.html>. Professor Huitfeldt is the founding Director of The Wittgenstein Archives at the University of Bergen (WAB) <http://wab.askis.uib.no/>.

[6] Catherine Plaisant, Martha Nell Smith, Loretta Auvil, James Rose, Bei Yu, Tanya Clement, "'Undiscovered Public Knowledge': Mining for Patterns of Erotic Language in Emily Dickinson's Correspondence with Susan Huntington (Gilbert) Dickinson," paper presented on "The Nora Project: Text Mining and Literary Interpretation" panel at Digital Humanities 2006.

[7] "Welcome," Metadata Offer New Knowledge (MONK), 4 Mar. 2009 <http://monkproject.org>

while new critical and philological approaches to literature benefit from the lexical and morphological data MONK offers. Likewise, the organizational data about how texts are structured (lines, stanzas, chapters, scenes, etc.) uncovered by MONK's analyses offers inevitable profits for narrative and discursive theory.

The MONK Project serves as a potential model for humanities computing initiatives that similarly bring together philological and social scientific strategies. MONK is built on "the basic assumption that the scholarly use of digital texts must progress beyond treating them as book surrogates."[8] This text-mining objective in resistance to the notion of digital reading as a mere proxy for reading in print or manuscript affects the changing conceptions of digital textuality. The project's extensive scope assembles disparate texts, from the 15th to the 20th centuries, providing an illustration for offering a wide-ranging dataset in a single environment. Moreover, MONK offers a powerful model for collaborative initiatives. Itself a collaboration among many large-scale, interdisciplinary and international organizations, the project also hopes to navigate across texts with open datasets (from WordHoard and Nora) and texts available on subscription basis only (from ProQuest, for example). MONK's aims bode well for promising developments in datastructure. It aspires to create a robust, fast, and flexible datastructure, successful development of which will be of critical interest for future digital humanities projects, including but not limited to digital literary archives.

Finally, MONK endeavors to integrate user contributions (WordHoard already offers the sharing of wordsets among users) such that the project resonates with intersections between scholarly ventures and social computing. Further discussion of the social computing elements of MONK, as well as ideas for other areas of expansion, are explicated below in the "Evaluation of Opportunities/Limtations for the Transliteracies Topic" section.

Technical Analysis – Software

(Note: This report was written in March 2009, prior to the public release of the MONK software and the completion of detailed documentation of all of the subcomponents. Much of the information below was derived from the MONK Project Wiki and discussions with the MONK Project team. This information is subject to change prior to the public release of MONK.)

MONK Technologies

While MONK is most directly characterized by the data mining algorithms and statistical techniques used in text metadata analysis, there are also many different general-purpose technologies used to develop the subcomponents that are loosely grouped together to create the overall system.

Java, JavaScript, ExtJS & AJAX

Java is a general purpose programming language developed by Sun Microsystems and first released in 1995. Java programs run in a Java Virtual Machine (JVM) which abstracts away differences in hardware and operating systems architecture to enable one application to run on many platforms. Java syntax was patterned after C and C++ but Java has more of an emphasis on Object-Oriented Programming (OOP) and has less lower-level system functionality as a

[8] "Project Description," Metadata Offer New Knowledge (MONK), 4 Mar. 2009 <http://monkproject.org/?page_id=13>.

consequence of running in a virtual machine. MONK uses Java either directly or through the use of Java-based application frameworks for most of its subsystems.

JavaScript is client-side scripting language that can be embedded in a web page to enable a wide range of capabilities beyond basic HTML. The code runs on the client side which means it needs to be downloaded each time the website is visited. Contrast this with server-side languages where relevant code is run by the web server before the page is returned to the user. JavaScript was designed to resemble Java but is not directly related to the Java programming language and instead derives its design from the Scheme programming language.

Asynchronous JavaScript and XML (AJAX) encapsulates a set of programming techniques and technologies that work together to create interactive web pages. Asynchronous means that the page can load and store data in the background without needing to explicitly reload the entire page. This technique enables web pages to act more like a desktop application rather than a sequence of static pages.

ExtJS (http://extjs.com/) is a JavaScript library that provides User Interface (UI) widgets and data handling functions for use in building interactive web pages. Most UI Widgets can use AJAX to retrieve data from a server as well as submitting new data to a server. MONK Workbench uses ExtJS as a framework for developing new clients and components to interact with the datastore.

REST and XML-RPC

Representative State Transfer (REST) is a method for building web applications that can maintain a logical state in order to more closely behave like traditional desktop applications. REST is not a specific platform but instead defines a "collection of network architecture principles which outline how resources are defined and addressed."[9] The overall idea with designing a system to be RESTful is to ensure that its resources are uniformly and predictably accessible across different domains without the accessing application needing to know specific infrastructure details (such as intermediate firewalls or proxy servers).

Remote Procedure Call (RPC) is a method for remotely executing code located in another application. The idea is for a client application to access the functionality of another (server) application without having to specifically code the remote application. Instead, an RPC method takes the same method parameters that would have been used if the code were present locally within the application and returns the results of the function call.

The separation of client and server allows for complex code to be run on a more powerful remote computer with results being returned to a less-powered "thin client." RPC also allows for applications written in two different programming languages to exchange information if there is a compatible RPC implementation for each platform.

[9] "Representational State Transfer," Wikipedia, 10 March 2009 <http://en.wikipedia.org/wiki/REST>.

XML-RPC is an RPC implementation that encodes the function call parameters using XML and uses HTTP as the transport protocol.[10] For this reason it is widely used in developing web applications that span multiple servers or implementation domains.

REST and RPC are considered to be two different solutions to the same class of problems and as such are typically not used in the same application unless both interfaces are required. Both can be classified as part of a Web Service which is "a software system designed to support interoperable machine-to-machine interaction over a network."[11] MONK has middleware based on the Spring framework that provides both REST and XML-RPC interfaces to the underlying datastore and analytics functionality.

Text Markup Languages

Text must be presented to MONK in a useful format before it can be stored in the database. Markup languages are used to encapsulate all relevant information about a text in a standardized way so it can be easily processed.

Extensible Markup Language (XML) specifies a format for storing text data and metadata together in a way that adheres to a defined schema to enable efficient information interchange. XML is commonly used in web applications across many domains because it is very flexible. It is described and maintained by the World Wide Web Consortium (W3C). The tagging structure is similar to Hypertext Markup Language (HTML) except the permissible values for the tags are not pre-defined. Since the tagset is extensible, XML can store almost any kind of data so long as it can be categorized in some fashion to be described using a finite domain of tags. XML is typically validated against a project-specific schema that describes how a well-formed document should look. This validation step is critical in ensuring proper data interchange between different applications.

The Text Encoding Initiative (TEI) is a consortium of many institutions dedicated to exploring ways to efficiently and safely encode physical texts digitally without losing information present in the original manuscripts. It is also focused on encoding methods to ensure easy digital text interchange between projects and institutions.

TEI publishes many guidelines that are used by digital humanities institutions when they are dealing with creating and processing digital texts. As of TEI version P5, XML is the exclusive markup language used for TEI as the newest guidelines rely on features in XML and other technologies such as XSL for processing TEI documents. [12] TEI guidelines essentially describe valid tags and the circumstances in which they are used. Since TEI aims to encompass any text in any language the full tagset is very large – nearly 500 in the current version[13] – but typically only a small subset is needed for a specific application. These subsets are referred to as the schema or Document Type Definition (DTD) and are used when validating a TEI-encoded text.

[10] "XML-RPC," Wikipedia, 10 March 2009 <http://en.wikipedia.org/wiki/XML-RPC>.
[11] "Web Services Glossary," 10 March 2009 <http://www.w3.org/TR/ws-gloss/>.
[12] "TEI: Learn the TEI," 10 March 2009 <http://www.tei-c.org/Support/Learn/>.
[13] "TEI: Introducing the Guidelines," 10 March 2009 <http://www.tei-c.org/Support/Learn/intro.xml>.

Tags in a TEI tagset fall into one of two categories depending on the type information they enclose.[14] Metadata tags capture information about the text including author, manuscript description, or copyright date. Structural tags describe the actual layout and appearance of the text. These have similar functionality to many of the tags in HTML that are used to specify paragraph breaks, heading styles, etc.

TEI provides a special format called One-Document-Does-It-All (ODD) for describing the specific TEI schema in use for a project that is portable across multiple validation systems. The ODD format is similar in appearance to XML and is used to create a single file that can be converted into various domain-specific formats. TEI provides a web interface named Roma (http://tei.oucs.ox.ac.uk/Roma/) for easily generating validation schemas in Document Type Definition (DTD), RELAXNG or W3C Schema format.

The MONK project uses TEI-Analytics (TEI-A) as the source format for texts to be ingested into the system. TEI-A is a subset of the full TEI P5 specification with current schema information available from the University of Nebraska, Lincoln (http://segonku.unl.edu/teianalytics/). According to a MONK paper abstract "from one perspective, TEI-Analytics is a minor modification of the P5 TEI-Lite schema, with additional elements from the Linguistic Segment Categories to support morphosyntactic annotation and lemmatization."[15] Source text may come to MONK in a different XML format (or other version of TEI) and so it must be converted to TEI-Analytics before further processing.

Extensible Style Language (XSL) is a relative of XML that is used to specify transformations of XML documents. It is described by the W3C and consists of three sub-parts:

- XSL Transformations (XSLT) - specifies how to transform an XML document into another representation
- XSL Formatting Objects(XSL-FO) - specifies how to format an XML document for display
- XML Path Language (XPath) - a language for addressing the parts of an XML document.

MONK uses XSLT style sheets in a package named Abbot to convert source texts in XML into compatible TEI Analytics format before they can be ingested into the datastore.

XSLT is used by running a processor program that takes an input XML document and an input XSLT style sheet (which is also in XML format) and outputs a resultant XML file. The XSLT style sheet contains a list of transformation rules that dictate what the output should be for various elements that may be encountered in the input document.

Web Application Frameworks

MONK applications are built on several different web application frameworks including Spring and Apache Struts. Spring (http://springsource.org) is an open source application framework for the Java platform. Like most application frameworks, Spring consists of many different elements

[14] "A Gentle Introduction to XML," 10 March 2009 <http://www.tei-c.org/release/doc/tei-p5-doc/en/html/SG.html>.
[15] "TEI-Analytics and the MONK Project," 10 March 2009 <http://www.cch.kcl.ac.uk/cocoon/tei2008/programme/abstracts/abstract-169.html>

bundled together. These elements (or modules) provide different services that a complex data-driven program may need. By using a framework such as Spring an application can integrate diverse functionality and perform complex tasks using proven methods without having to code such functionality from scratch. Many features of Spring can be used in any typical Java program but there are extensions that enable it to be used when developing web applications on the Java Enterprise platform.

MONK incorporates various middleware programs built on the Spring framework that provide communication interfaces to the back-end datastore and analytics functionality that can be used by client applications.

Struts (http://struts.apache.org/) is an open source framework for developing web applications on the Java Enterprise platform sponsored by the Apache Foundation. Struts uses and extends the Java Servlet API to provide the framework to develop applications using the Model View Controller (MVC) paradigm.

MVC is a way of segmenting the application into three domains -- Model (the database and logic needed to interact with the database), View (the presentation layer that the user interacts with) and Controller (the logic that transfers information between the Model and the View). The goal with MVC is to make an application easier to write and maintain. Segmenting the application into three separate domains encourages security by discouraging data access code from accidentally being intermingled with public interface code.

Java Servlets are Java Objects based on a web server that receive data requests from a client and construct dynamic responses that are sent back to the user. Servlets are the Java counterpart to other server-side platforms used to dynamically generate web content such as PHP and ASP.NET. MONK uses Struts as the underlying framework of the TeksTale classification system application.

OpenLaszlo (http://www.openlaszlo.org/) is an open source application development and delivery platform used for rich Internet applications. Applications made using OpenLaszlo can be deployed as traditional Java servlets which are compiled and returned to the browser dynamically or as Adobe Flash applications. OpenLaszlo is cross-platform since it uses Java and Flash and is delivered via a standard web browser. OpenLaszlo applications are typically intended to provide an experience similar to traditional desktop applications with the benefit of "ubiquitous" access afforded by web-based delivery mechanism.

OpenLaszlo is sponsored by Laszlo Systems (http://www.laszlosystems.com/), a private for-profit company which originally developed the platform and continues to market their own Laszlo applications. MONK uses OpenLaszlo as the delivery platform for the FeatureLens Frequent Pattern Analysis tool.

Data Exploration Frameworks

The Software Environment for the Advancement of Scholarly Research (SEASR) is a programming environment designed to enable digital humanities research. The goal is for a platform that enables researchers to create powerful applications that can use a variety of source

data formats as well as enable interoperability with other research projects developed on the platform.

According to the SEASR website (http://seasr.org/), SEASR addresses four key needs:

- the ready transformation of (semi-)unstructured data (including natural language texts) to the structured data world through building extensible software bridges
- improved basic knowledge discovery through supporting enhanced analytics
- time- and distance-independent scholarly and technology exchange through constructing virtual research environments
- fully open-sourced development for maximizing community involvement, such as by sharing user applications through community repositories

MONK uses SEASR-based applications for performing statistical analysis on natural language texts.

Data2Knowledge (D2K) is "a generalized visual-programming framework for data-mining developed and still being improved at the National Center for Supercomputing Applications, in the Automated Learning Group."[16] The goal with D2K is to enable researchers to rapidly create applications to perform typical data-mining tasks that are useful for "prediction, discovery, and deviation detection, with data and information visualization tools." [17] D2K components are written in Java which enables cross-platform compatibility.

The Nora project, a predecessor to MONK, uses D2K applications for "Data Exploration" but the developers say they are looking to wrap "Data Preparation" into D2K as well. MONK has adapted many of these applications for its own analytics use.

Datastore
MySQL is a relational database system that uses Structured Query Language to retrieve and store information. It is mostly open-source but is maintained by a private for-profit company named MySQL AB. The system provides a database server that hosts multiple separate databases and can be accessed by many users simultaneously. In addition to an optimized Enterprise edition there is a free Community version.

MySQL has been shown to scale well for applications storing a very large amount of data as well as hosting a large amount of concurrent connections (clients). For this reason the database system is popular for web applications and is available for many different operating systems. MONK uses MySQL to store all the text in the system after it has been tagged and ingested. Middleware software accesses the datastore and provides connectivity to client applications.

[16] "Project Description," Metadata Offer New Knowledge (MONK), 10 Mar. 2009 <http://monkproject.org/?page_id=13>.
[17] "Project Description," Metadata Offer New Knowledge (MONK), 10 Mar. 2009 <http://monkproject.org/?page_id=13>.

MONK Subsystems

MONK consists of many subsystems that interact with each other and operate in many different domains. Some are command line applications that run on a server or other high-powered machine that processes data one time when it is being analyzed for inclusion in the system. Other systems operate as client-side web applications that are used by researches to access the knowledge contained within MONK. Between these two domains there are many subsystems that act as bridges between different platforms.

It is important to note that unlike a more homogenous application, each subsystem has differing target user demographics and most end users will not directly use many of the subsystems but will instead interact with the data produced by them. Most of the information for this section was provided in personal communication with Amit Kumar, a programmer working on MONK at the University of Illinois at Urbana-Champaign.

Abbot

Abbot encompasses the software needed to convert source texts into a standardized representation. Natural language text for the MONK project comes from many different digital collections that use a variety of storage formats and often have slight differences in how certain textual elements were encoded in the initial digitization stage. Abbot takes these texts and converts them as best it can into TEI-Analytics format, a subset of the Text Encoding Initiative's P5 specification.

Data loss is an issue when converting between different formats as other text formats may track elements that are unsupported in the final TEI-A format. The MONK project Wiki describes these issues in detail (https://apps.lis.uiuc.edu/wiki/display/MONK/Abbot+and+TEI-Analytics+texts).

The specific steps involved in the Abbot process are described on the project Wiki (https://apps.lis.uiuc.edu/wiki/display/MONK/Abbott+process). After converting the files to TEI-A format they are passed to MorphAdorner for further processing. According to the MONK project Wiki they have converted over 1,200 works into TEI-A format using Abbot as of April 12, 2008.

MorphAdorner

MorphAdorner is a Java command-line program which performs "morphological adornment of words in a text."[18] Morphology concerns the identification, analysis and description of structure of words. Text adornment is the process of annotating source text in order to record metadata that can be used by other applications to explore the data.

Natural language source text in TEI format provides the input to the program and TEI-encoded text is returned as output. The adornment is done by adding additional XML tags to the output while also copying any input tags.[19] This XML-based approach has the benefit of maintaining format compatibility with TEI rather than other adornment methods that typically include adornment tags inline with the source text.

[18] "MorphAdorner," 10 March 2009 <http://morphadorner.northwestern.edu/morphadorner/>.
[19] "MorphAdorner XML Output," 10 March 2009 <https://apps.lis.uiuc.edu/wiki/display/MONK/MorphAdorner+XML+Output>.

MorphAdorner provides "methods for adorning text with standard spellings, parts of speech and lemmata"[20] as well as including tools for tokenizing text, recognizing sentence boundaries, and extracting names and places. Many of these tools are separately testable through demonstrations on the MorphAdorner website (http://morphadorner.northwestern.edu/morphadorner/).

The overall dataflow consists of:

- Input
- Sentence Splitting
- Tokenization
- Spelling Standardization
- Part of Speech Tagging
- Lemmatization
- Output [21]

MorphAdorner was developed at Northwestern University by Philip R. Burns and Martin Mueller and exists as a separate project from MONK.

One of the most important functions of MorphAdorner is Part of Speech (POS) tagging which is the process of adorning words in a text with each word's corresponding part of speech. This information is critically important to enable higher-level analysis of the relationships between natural language texts. POS tagging uses a tagset that defines which tags are valid and in what conditions to apply each tag. By default MorphAdorner uses a part of speech tag set designed by Martin Mueller named NUPOS. See the Algorithms section for more information on POS tagging.

Acolyte
Acolyte is a process for specifying additional metadata to be included with a text when it is ingested into the system. This metadata is often provided by collection curators to supplement what is available from the text. This stage is important because text comes to MONK in various formats and may not have all relevant information stored within the source file.

PRIOR
PRIOR encompasses the software routines that ingest text in TEI-A format into the datastore. This is the process where text that has been annotated is mapped from a TEI-A XML format to a set of database tables in the MySQL database. This mapping is crucial to ensure that important metadata has not been lost and that it can be properly queried by MONK applications. PRIOR combines the annotated text with additional metadata supplied by Acolyte in order to create a unified datastore representation. John Norstad at Northwestern University is the primary datastore designer.

[20] "MorphAdorner," 10 March 2009 <http://morphadorner.northwestern.edu/morphadorner/>.
[21] " MorphAdorner: Tech Talk," 10 March 2009 <http://morphadorner.northwestern.edu/morphadorner/techtalk/>.

Statistical Analytic Facilities
Statistical text analytics facilities are built on the SEASR platform and used by end-user clients for Data Exploration. Access to these analytical facilities is provided by the MONK Middleware. See the Algorithms Section for more information on the statistical methods used in MONK.

Monk Middleware
Middleware is the "glue" that bridges the back-end MONK infrastructure with client applications running on many different platforms. MONK middleware is built on the Spring framework and provides RESTful and XML-RPC interfaces to the underlying datastore (MySQL database) and SEASR-based analytics procedures.

Through the use of middleware a wide variety of MONK end-user applications can access the common datastore and analytics facilities of MONK without being constrained to specific implementation platforms or programming languages. This enables MONK to retain its usefulness as new and improved application platforms are developed.

MONK Client Applications
Though the project is not finished there are a few client applications already developed that can be used by researchers to explore the metadata. These client applications are typically web-based and access MONK services via the middleware. Many of these applications are currently password protected and not available for public trial while the project is being finished.

Monk Workbench is an EXTJS based framework for developing components that consume the services. It allows client applications to use a set of modular components that access MONK services.[22] The emphasis on modularity enables components to be used in multiple client applications to make creating new applications quicker and easier.

TeksTale is a classification application based on the Apache Struts framework. It is partially developed by the National Center for Supercomputer Applications at UIUC.

MONK Flamenco (http://flamenco.berkeley.edu/) is a port of the Berkeley Flamenco project. Flamenco is a "search interface framework" that is designed to help users efficiently move through large information spaces. It focuses on metadata as a means of helping the user figure out where to go next and how to find the specific subset of information that they need. The framework is open source so MONK project has ported this library to interface with the MONK datastore and bring charting capabilities to MONK.

FeatureLens is application for frequent pattern analysis. It consists of a ruby web server that interacts with the database and an OpenLaszlo client. It is designed to highlight reoccurring patterns across a wide set of documents. Detailed information on the software architecture has been posted to the MONK Wiki (https://apps.lis.uiuc.edu/wiki/display/MONK/Code+Documentation). There is also a live demo available to the public as of March 2009.

[22] "MONK Workbench," 10 March 2009 <https://apps.lis.uiuc.edu/wiki/display/MONK/Monk+Workbench>.

Technical Analysis – Algorithms

This section describes the text-analysis algorithms used in the software developed by the MONK Project.

Goals

MONK aims to aid humanities scholars in discovering patterns in the texts they study by compiling metadata from these texts. This metadata ranges from the level of word occurrence to top-level bibliographic data. Consequently, the algorithms used by MONK attempt to develop this metadata, either by analyzing the text itself, or by performing analysis on lower-level metadata.

Although many of the algorithms used in MONK are also used in other text-analysis systems, the specific objectives within the humanities and, hence, the methods for judging the effectiveness of these algorithms differ from those in other fields. A report by the Nora Project defines the best algorithm in the context of literary study as the one that provides, "the most interesting results . . . the set of results that help you to think more deeply about the ideas your investigating."[23] In application such as spam filtering, where the goal is to accurately define which texts belong to a certain category (spam), success can easily be judged by a human reader. In contrast, a humanities scholar would likely explore classification that cannot be clearly judged by a human reader in order to develop a further understanding of the features that categorize a set of texts.

Tagging

Described most simply, MONK uses a catalogue of the frequency of words within each text as the lowest level of metadata. (This is often called the "bag of words" model.) However, for the purposes of literary analysis, it is not sufficient to classify word tokens simply by searching for combinations of characters. Firstly, MONK differentiates words based on grammatical information. For example, use of the word "love" as a noun must be differentiated from its use as a verb. The association of this information with each word is called part-of-speech (POS) tagging.[24] Additionally, as MONK is developed for use with a large variety of texts, there may be a great deal of orthographical variance between texts. MONK applies a layer of "virtual orthographic standardization" (VOS) that standardizes word tokens prior to tagging.[25]

Collocations and N-grams

This "bag of words" model, which does not consider the ordering of words, is sufficient in many cases. However, the MONK Project and its predecessors also have methods for analyzing groups of words occurring in a text. Firstly, a text may be arbitrarily segmented into multi-word tokens called "N-grams." Nora allows for analysis of text segmented into bigrams (two word tokens) or trigrams (three word tokens) in addition to individual words.[26] MONK tags each word with its position within a text so, in practice, MONK can access a word's immediate neighbors by adding or subtracting 1 from the position index. MONK and WordHoard also consider collocation, collections of words that occur together frequently. Phrases (which are called "multiword units" by WordHoard) a particular type of collocation. Simply, an N-gram that occurs statistically often in a text is likely a phrase.

---

[23] "NORA Final Report."
[24] MONK Project, "Project Description (Long)," 2007, 23 Oct. 2008 <http://monkproject.org/?page_id=13>.
[25] MONK Project, "Project Description (Long)," 2007, 23 Oct. 2008 <http://monkproject.org/?page_id=13>.
[26] "NORA Final Report."

The MONK project describes two cases where it would be beneficial to regard sequences of words as single tokens. The first is short sequences of words that are used together often enough that they "might as well be words" such as "out of" or "in so far," and the second is multiword names such as "William Shakespeare."[27]

Statistical Routines

MONK offers a number of statistical routines that make use of the low-level word token catalogue. Initially, the MONK Project is focusing on developing routines that compare two sets of texts using Bayes classifiers and Dunning's log likelihood statistic.[28]

MONK uses Bayes classifiers to classify texts as belonging to one of multiple categories of texts[29]. This classification is based on a statistical comparison of the texts being analyzed to a corpus of texts that have already been analyzed and classified. An article on the use of nora describes a typical process of using automatic classifiers in the digital humanities context:

> 1) the system provides the user (in this case, a scholar) with a sample of documents from the collection
>
> 2) the scholar chooses among the sample documents those which are of interest for a particular study. In the two Nora project examples, a sample of poems from a collection of Emily Dickinson was rated in terms of erotic content, and a sample of novel chapters was rated according to their instantiation of the concept "sentimentalism."
>
> 3) the system performs a set of "feature extraction" actions in order to determine shared characteristics of the selected documents
>
> 4) the scholar examines the shared characteristics and iteratively adjusts the result as necessary
>
> 5) the system applies the resolved characteristics to the larger collection in order to automatically identify similar documents
>
> 6) the scholar studies both the shared characteristics and the result set, often by using a visualization tool (in Nora, the InfoVis toolkit).30

As described earlier, the goals in this process differ greatly from the use of text-analysis for spam detection, which also uses Bayes classifiers. In the digital humanities process, the scholar is often more interested in the features that lead to the automatic classification, rather than the

---

[27] "Notes towards a user manual of Monk," 2009, 5 Mar. 2009 <https://apps.lis.uiuc.edu/wiki/display/MONK/Notes+towards+a+user+manual+of+Monk>.
[28] "Notes towards a user manual of Monk," 2009, 5 Mar. 2009 <https://apps.lis.uiuc.edu/wiki/display/MONK/Notes+towards+a+user+manual+of+Monk>.
[29] "Notes towards a user manual of Monk," 2009, 5 Mar. 2009 <https://apps.lis.uiuc.edu/wiki/display/MONK/Notes+towards+a+user+manual+of+Monk>.
[30] Stan Reucker, Ximera Rossello, Greg Lord, Milena Radizkowska, "The Clear Browser: Visually Positioning an Interface for Data Mining by Humanities Scholars," *Digital Humanities 2006*, Jul. 2006: 258.

classification itself. Additionally, this process allows the scholar(s) to iteratively modifying the classification process which enables open interpretation of the texts.

Ted Dunning's log likelihood ratio is a statistical routine developed in 1993 specifically for text-analysis. Dunning's paper, "Accurate Methods for the Statistics of Surprise and Coincidence" describes how many statistical methods, such as Pearson's χ^2 test and z-score tests, are insufficient for text-analysis because the corpora are often not large enough and typically contain a large amount of rarely occurring word tokens, while these tests assume a large sample set with a near normal distribution.[31] The routine using a log likelihood ratio that Dunning suggests is effective for analyzing text.

As used in MONK, the corpus being analyzed is compared to a reference corpus, producing a list of words (or other tokens) that are unusually common or rare. The working user manual for MONK describes an example of the routine's use:

> The list produced by it is often interesting precisely because the words on it are so ordinary. If you compare Julius Caesar with Shakespeare's other tragedies, 'she' is by far the biggest outlier, and it is underused. Overused words are 'man', 'countryman', 'today', 'mighty', 'do', 'countryman', 'street', 'run', and 'honorable'. One can make quite a bit of this list, and generally the list of over- and under-used words in a given text circumscribes major thematic areas with considerable precision.[32]

Evaluation of Opportunities/Limitations for the Transliteracies Topic (and the Bluesky Group)

[TO BE ADDED]

Resources for Further Study

"ExtJS." <http://extjs.com/>.

"The Flamenco Search Interface Project." <http://flamenco.berkeley.edu/>.

"MONK Project." <http://monkproject.org/>.

"MorphAdorner." <http://morphadorner.northwestern.edu/morphadorner/>.

"The Nora Project." <http://www.noraproject.org/>.

"OpenLaszlo." <http://www.openlaszlo.org/.>

"Roma." <http://tei.oucs.ox.ac.uk/Roma/.>

"Spring Source." <http://springsource.org>.

[31] Ted Dunning, "Accurate Methods for the Statistics of Surprise and Coincidence," *Computational Linguistics*, Volume 19, number 1, 1993: 61-74.
[32] MONK Project, "Project Description (Long)," 2007, 23 Oct. 2008 <http://monkproject.org/?page_id=13>.

“Struts.” <http://struts.apache.org/.>

“WordHoard.” <http://wordhoard.northwestern.edu>.