

White Paper Report

Report ID: 98551

Application Number: HD5000307

Project Director: Michael Newton (msnewton@email.unc.edu)

Institution: Unaffiliated Independent Scholar

Reporting Period: 4/1/2007-5/31/2008

Report Due: 8/31/2008

Date Submitted: 9/15/2010

NEH White Paper

NEH Grant number: HD-50003-07
Finding the Celtic

Project Director: Michael Newton
gaelicmichael@yahoo.com

June 2008

Summary

Finding the Celtic (hereafter referred to as 'FtC') is an experimental online digital humanities collaboratory for Celtic Studies produced by modifying and extending the *Collex* project. *Collex* has been in development in the Applied Research on Patacriticism Lab at the University of Virginia (and applied to Nineteen-Century Studies as the *NINES* online collaboratory) since 2006. *FtC* was funded by a Digital Humanities Initiative grant from the National Endowment for the Humanities from May 1, 2007 to April 31, 2008.

FtC enables users to locate, access, and create visual representations of a sample set of scholarly resources for Celtic Studies that have been identified on the internet. Social software features enable users to collect objects, tag them, view the universe of tags created by other users, and create exhibits from collected objects. *FtC* is hosted by iBiblio at www.celtic.ibiblio.org

The engagement between *FtC* and the ARP Lab has enhanced the *Collex* suite of software components which comprise the online collaboratory. They are available as Open Source Software which can be downloaded from the University of Virginia repository and customized according to the needs of other fields.

This document describes the activities, challenges and accomplishments of the project, both from technological and institutional points of view.

1. Project Achievements

1.1 Adjusted Goals

By the time that the project was under way, the goal set was defined as:

- To configure and adapt the *Collex* digital collaboratory to handle data resources related to Celtic Studies (i.e., defining the appropriate metadata schema and tailoring the code to handle it).
- To add information visualization extensions to *Collex* so that the data resources can be visualized in graphical representations.
- To compile a sample database of digital resources from the multiple disciplines that intersect at Celtic Studies (i.e., literature, history, archaeology, folkloristics, and art history) so as to demonstrate the ability of the system to meet the diverse needs of the humanities.
- To stimulate the creation of online resources for Celtic Studies.

1.2 Collex Design

The name *Collex* derives from the words **C**ollection and **E**xhibits. It is intended to function as a front-end to the Semantic Web for scholars, allowing "users to collect, annotate, and tag online objects and to repurpose them in illustrated, interlinked essays or exhibits."¹ The *Collex* server does not contain primary data resources but rather metadata entries about resources available online; its faceted classification system allows for highly flexible browsing of data and the folksonomy tagging system allows the user community to add value to objects by annotating and labeling them.

FtC was the first attempt to extend *Collex* technology into a new realm (beyond its initial application to Nineteenth-Century Literary Studies). This experience of extended collaboration between the UVA ARP Lab and *FtC* affirmed the effectiveness and desirability of *Collex* technology, but also revealed how deeply code specific to *NINES* had become embedded in *Collex*. This process initially resulted in "quarantining" code specific to *NINES*, improving some of the documentation intended to explain its application outside

¹ The "About" webpage for the *Collex* project, at <http://www.patacriticism.org/collex/about/>

of the ARP Lab, and creating an initial design for a generalized *Collex* engine which can serve a domain-specific client (a design currently in development at UVA).

The FtC metadata scheme design supports a broad range of data related to Celtic Studies, with particular tradeoffs in mind. Sample items in the data base highlight their application to materials relating to a broad range of topics in the humanities.

The two primary visualization types – geographical and chronological – provide obvious and useful means of organizing data results in graphic format. The software extensions to provide this functionality are integrated into the FtC version of *Collex* so that these techniques can be reused by other projects that wish to also provide visualization features.

The rich metadata schema used by FtC enables flexible, multidimensional explorations into the data base and advantageous means of visualizing it. A number of useful questions about patterns in the data can be asked by parameterizing graphical representations, especially on the Graph page.

1.3 Source Code

The code base for FtC is stored in the UVA ARP Subversion repository and is freely available for download at: <https://subversion.lib.virginia.edu/repos/patacriticism/collex/branches/ftc> Technical details about the code (such as versions, dependencies and known bugs) are given in the **README** file in the `/web` directory.

Details about prerequisite Open Source components, downloading the FtC code base, installation and configuration can be found on the UVA ARP Lab Wiki at: http://faustroll.clas.virginia.edu/ARPwiki/index.php/Main_Page

1.4 FtC Website

In constructing the FtC website I also created webpages to explain the project, describe the metadata schema and prescribe system usage, and provide a basic help system for users. This in itself required considerable time and effort. The results can be seen online.

2. Design Process and Results

Some preliminary notes on terminology are necessary to understand the discussions that follow. *Collex* is a *federated data server*, meaning that it contains metadata about primary data resources contained on other servers; in other words, *Collex* can be conceptualized as containing an index to files stored elsewhere on the internet.

The metadata about items are stored in RDF files which are ingested in the *Collex* system; after they have become ingested, these items are often referred to as *documents* (although I generally adhere to the more generic “items”); each document has a number of *facets*; each facet corresponds to a metadata element in the original RDF file.

2.1 Metadata Design

The metadata schema design was as much an exercise in anticipating the functionality of the visualization features as identifying the metadata offered by data bases in the present. It was carried out by a committee consisting of Carole Crumbley, Elizabeth Jones, Michael Newton, and Dorothy Verkerk. The schema was meant to be a significant expansion over the domains represented in *NINES* and cover a wider range of materials from the humanities. The types of items deemed to be important to represent in the data base were as follows:

History: Biographical entries, maps, historical events (or periods), and definitions of terms.

Archaeology: Artifacts and sites of archaeological significance.

Art History: Architecture, illuminated manuscripts, and other objects of “art.”

Literature: Manuscripts, transcriptions, editions, and translations.

Folkloristics: Audio recordings (from folklore fieldwork) and representations of human activities (whether in textual, audio or video format).

Each “real-life” entity, whether concrete (such as a building) or conceptual (such as a biographical entry) is represented on the computer by a *digital surrogate*. For example, a manuscript is represented by a digitally scanned photograph of the original and the characteristics of this surrogate will be determined by the granularity of the image (e.g., number of dots per inch), lighting conditions, digital enhancement techniques (e.g., modifying the contrast), and so on. The digital surrogate is not a substitute for the original entity and two digital surrogates of the same entity may differ.

The metadata schema was also driven by the features of the system, especially the graphic visualizations. A geographical display required a geographical coordinate to be of any use; without such information, the extensions would not be usable.

The challenge, then, was to find a minimal metadata schema which would support the essential qualities of all of these different item types without cluttering the schema with elements specific to a particular item type. A table of the elements, specifying the name of the element, whether or not it is required for every entry, whether or not an entry can have multiple values, and its meaning, is given below. Metadata elements that were inherited from the *Collex* schema are underlined. Some discussion of individual items follows.

Element	Req'd	Multi	Meaning
<u>Identifier</u>	Yes	No	Unique identifier for this item
<u>Archive</u>	Yes	No	The name of the archive which contains this item; this enables FtC to exploit data stored about specific archives (such as the archive's home URL).
<u>Title</u>	Yes	No	The human-readable title for this object. It does not need to be unique.
<u>Alternative Title</u>	No	Yes	Alternative names for items; this is useful for providing aliases for items that may be known under different names in different linguistic or scholarly traditions, or have a special designation (e.g., a library index number).
<u>ItemType</u>	Yes	No	What type of item this is. Must be one of the following: Activity, Architecture, Artifact, Audio, Definition, Event, Manuscript, Map, Site, Text
<u>Role</u>	No	Yes	Individuals who have been involved in the creation or transmission of this item in all of its manifestations. Discussed further below.
<u>Date</u>	Yes	Yes	The date or date range associated with this item, the exact meaning of which depends on the type of artifact.
<u>Period</u>	Yes	No	The period associated with this artifact, the exact meaning of which depends on the type of item. Must be one of the following recognized Celtic eras: Hallstatt, La Tene, Romano-Celtic, Early Medieval, High Medieval, Late Medieval, Modern.
<u>Country</u>	Yes	No	The two-letter country code for the country (as defined by the boundaries of modern nation-states) associated with this item. If no single country is appropriate, use code ZZ. (See further discussion below.)
<u>FoundLoc</u>	No	No	The geographical coordinate (in latitude,longitude format) at which the object was found (or associated).

Element	Req'd	Multi	Meaning
<i>Language</i>	No	Yes	The languages that appear on or in the item. There is a set list of allowable language names.
<i>Material</i>	Yes	Yes	The material(s) of which an item is made. If the object exists only in digital form (as in items of type Definition, Event, or Site), the value is Digital. Must be one of the following values: Digital, Paper, Vellum, Leather, Wood, Stone, Gem, Bronze, Copper, Lead, Pewter, Brass, Iron, Silver, Gold, Bone, Textile, Botanical, Glass, Ceramic, Human Remains.
<i>Text</i>	No	No	Contains either (1) a short, plain text value, or (2) a URL to a web-accessible, plain text file containing the text for the item. In both of these cases, the text cannot contain special markup (like XML or other embedded codes). The text is used by the Collex search engine for the purposes of indexing.
<i>SeeAlso</i>	No	No	The URL to the webpage associated with this item. This allows a pointer to be displayed alongside the item that the user can access.
<i>Thumbnail</i>	No	No	The URL to a web-accessible miniature image of the item (100 x 100 pixels in size) to display. If no thumbnail is available, the thumbnail for the archive will be displayed; if there is no thumbnail associated with the archive, a generic image will be shown.
<i>SourceWork</i>	No	Yes	The title of the larger item of which this item is a component. For example, a journal series, an anthology, a manuscript containing many texts, etc.)
<i>Parent</i>	No	No	The Identifier of the "parent" item of which this item is a "child" component. They should both belong to the same archive. (Currently unused but expected to be used to link items.)
<i>Children</i>	No	Yes	The Identifiers of "children" items that belong to this "parent" component. They should both belong to the same archive. (Currently unused but expected to be used to link items.)
<i>Relations</i>	No	Yes	The Identifiers of any items related to this item that are not related in a Parent-Children relationship. For example, different translations of the same original text. (Currently unused but expected to be used to link items.)

Table 1: FtC Metadata Schema

There are some metadata elements which ideally should have allowed multiple values, but because of design considerations, could not. It was necessary to limit items to having a single country code, for example, in order for searches and displays to work in a reasonable manner (see Section 4.4.1 below for more).

The definition and usage of some values have been complicated because of the wide range of times and cultural and scholarly traditions the data set attempts to include.

2.1.1 Names

I determined that names (used in the Roles element, and for the Title in the case of biographical entries) should be given in the order first name to last name in the language of the person in question. While the standard usage in modern English is surname followed by Christian name and middle name, surnames are a modern convention that do not hold in earlier periods or even in Celtic languages to the present (to a

large degree). Given that epithets are also common in Celtic names, the use of these are encouraged to facilitate disambiguation. Some examples include:

Donnchadh Bàn nan Òran Mac an t-Saoir
Dafydd ap Gwilym
Alasdair mac Mhaighstir Alasdair Domhnallach

2.1.2 Countries

The values of the CountryCode element are also defined in a “non-standard” way. It was deemed important to recognize those Celtic cultural zones in existence in the High Medieval period that are nations according to traditional usage of the term but have been subsumed within the United Kingdom and France in the modern era. This led to the creation of several new CountryCode values (some of which replace previously defined countries outside of the Celtic world): BR (Brittany), CN (Cornwall), EN (England), IE (the island of Ireland, not just the Republic of Ireland), MN (the Isle of Man), SC (Scotland) and WA (Wales). The definition of these discrete countries was deemed important given that they have produced a considerable amount of material with distinctive characteristics.

2.1.3 Dates

Although the Date element in *FtC* works in a similar way to the corresponding element in *NINES*, it was necessary to extend the format given the wide range of dates (c. 2,000 BCE to the present, as opposed to the “long” nineteenth century used by *NINES*). Years before the Common Era are indicated by a negative sign (“-”) and ranges expressed with “x”. For example, the date range 100 BCE to AD 100 is represented by “-100x100”.

2.1.4 Object Relationships

NINES-Collex had already defined the Parent and Children elements to allow “containment” relationships between items to be specified (e.g., articles within journals). I suggested that this concept of containment be extended semantically in the *FtC* metadata schema to allow for specifying location and artifact composition. A Site item can be a “parent” to all of the archaeological artifacts that are found there. The hierarchical relationships between an item and the subsidiary items of which it is composed – such as composite artifact made of many smaller artifacts – are another example. This could also be used to express “lineage” relationships, such as a parent Manuscript from which editions of Text items are created.

Although the system does not currently make use of these relationships, it is possible that future versions of *Collex* may allow searching, browsing, and visualization based on such hierarchies, and so defining them at an early stage was important.

2.1.5 Roles

NINES-Collex defined the Roles element to specify individuals who were involved in the creation of an item. I augmented the usage of this element in two ways: I created the Role:PER specifier to refer to performers or tradition-bearers from whom audio or video recordings were made (and whose role is different from that of an author or translator). I also extended the usage of the Role:PUB (publisher) specifier to include patrons of the arts in the pre-Modern period (whose function was somewhat analogous to that of the modern publisher).

2.1.6 Ideas discussed but discarded

It may be of future interest and utility to know about those elements discussed for inclusion but ultimately discarded.

We discussed a proposed Motif element, given that certain culturally-specific motifs recur in different media (text, visual art, etc), but the list was potentially long and difficult to define, and a similar result can be provided by users exploiting *Collex*'s folksonomy tagging feature.

We discussed a proposed SourceType element, which would specify whether the item was a primary, secondary, tertiary (bibliographic) or derived (created by computation) resource, but this was deemed implicit and unnecessary.

We discussed a proposed SubjectDomain element, which would specify whether the item related to Politics, Law, Military, Religion, etc., but determining the value(s) appropriate was cumbersome and subjective, each item was likely to have multiple values, and a similar result can be provided by users exploiting *Collex*'s folksonomy tagging.

We discussed a proposed SocialScale element, which would specify the scale at which an item related in social terms (Personal, Family, Village, Tribe, Superstate), but this too was likely to be difficult to determine.

We discussed a proposed RhetoricalMode element which would specify whether an item was expressed in the form of Fiction, Criticism, Didactic, Expository, etc., but this category is specific to Text items and would be unused for other item types.

Another strike against all of the above proposed elements is that they do not relate to the elements of standard metadata schema (e.g., Dublin Core) and hence would require extra human intervention.

2.2 Feature Design

2.2.1 Browsing

The FtC user interface for multifaceted browsing was borrowed directly from *NINES-Collex* with several modifications:

- **FtC-specific facets in constraints panel:** Because of the embedding of metadata schema-specific code in *Collex*, changing the user interface to handle the FtC metadata schema was non-trivial. Besides accounting for the difference in facets (and the differing repository scheme represented by the "Resource" facet), translating between the 2-letter CountryCode values and country names was deemed a necessary effort for the user experience (i.e., so that the value "IE" was displayed as "Ireland").
- **Constraint Display:** By (European) convention, readers/users scan visually from left to right; accordingly, the display of cause and effect should move from left to right. The constraint set which determines the resulting data set was thus reorganized from its original order on *NINES*; in FtC, the range of possible constraints is displayed on the left-hand side, while the currently selected constraint set is on the right-hand side.
- **Side Bar:** The *NINES*-specific facets in the (left-hand) side bar were replaced by those of FtC. The efforts necessary for this conversion were similar to those for the constraint panel.
- **Navigation Bar:** New buttons were added to the pre-existing navigation bar to allow the user to move between the *Collex* browser and the new visualization pages.

2.2.2 Settings

It was important to make use of color as a visual dimension to represent information on the graphical displays. It is necessary, however, to translate between the enumerated values of facets (such as CountryCode) and color values (for example, Ireland corresponds to the color Green). I thought it important to allow the user to specify that translation, especially because, in the case of some facets, there are not enough color values to cover all facet values. It would be too tedious for the user to have to provide that translation for every visualization, so I chose to store it in the record created for the user when s/he

creates an account. I believe that it would be too cumbersome for the translation to occupy the same screen space as the visualization itself, so I created a webpage dedicated to allowing the user to specify the translation of these values. Default settings are provided to users who are not logged into an account.

2.2.3 Atlas

A geographical display, placing the items resulting from the current constraint selection onto a geographical map, was an important visualization type in this project. The current implementation uses GoogleMaps to display an icon at the item location and, when the user hovers the cursor above the item, the item title.

2.2.4 Timeline

A chronological display, mapping the items resulting from the current constraint selection on a time chart supporting date ranges (a single year is not sufficient), also was an essential visualization type in this project. The current implementation uses MIT's SIMILE-Timeline widget; besides indicating time by the horizontal position of the marker/bar, further information can be conveyed by the color of the marker/bar and title text. Further textual information can be shown in a pop-up window activated by the user by selecting the timeline bar. These visual features can be mapped to either the Country, Period or ItemType facet of the items displayed.² This is a powerful way to convey several facet values simultaneously on the Timeline display (although it may be necessary to experiment with value mappings on the Settings page for the results to be aesthetically pleasing or quickly comprehensible).

2.2.5 Graph

It was further envisioned that users would be able to create abstract graphs displaying the numeric facet values of items. Since there are no item facets that contain numeric data, however, graphs such as X-Y plots could not be generated. I decided that the most useful representation, given the current metadata schema, would be a series of bar graphs displaying the distribution of facet values across two facet "dimensions."

The user interface allows the user to specify which two facet dimensions are analyzed. The *Group By* dimension (which can be set to ItemType, Country or Period) first divides all resulting items into a groups for each value of this facet (each discrete value having its own bar). The *Count* dimension (which can be set to ItemType, Country, Period, Material or Language) creates a separate segment corresponding to each discrete value within the group (colored according the Settings page).

This visualization allows for a powerful means of analyzing the distribution of values across the data base by creating a graphical representation which is easy to comprehend visually.

2.2.6 Help and User Guide

It was necessary for me to create an entirely new online help system and user guide (including an explanation of the metadata schema and guidelines for its use for contributors).

3. Implementation Details

3.1 System Architecture

Collex consists of a set of interoperating Open Source software components. The main functions of the system are ingestion (reading RDF files and converting them into an internal representation) and user web services.

² Since only one visual item appears per data base item, visual characteristics have to correspond to facets with a single value (i.e., facets that don't require a value or can have multiple values cannot be represented).

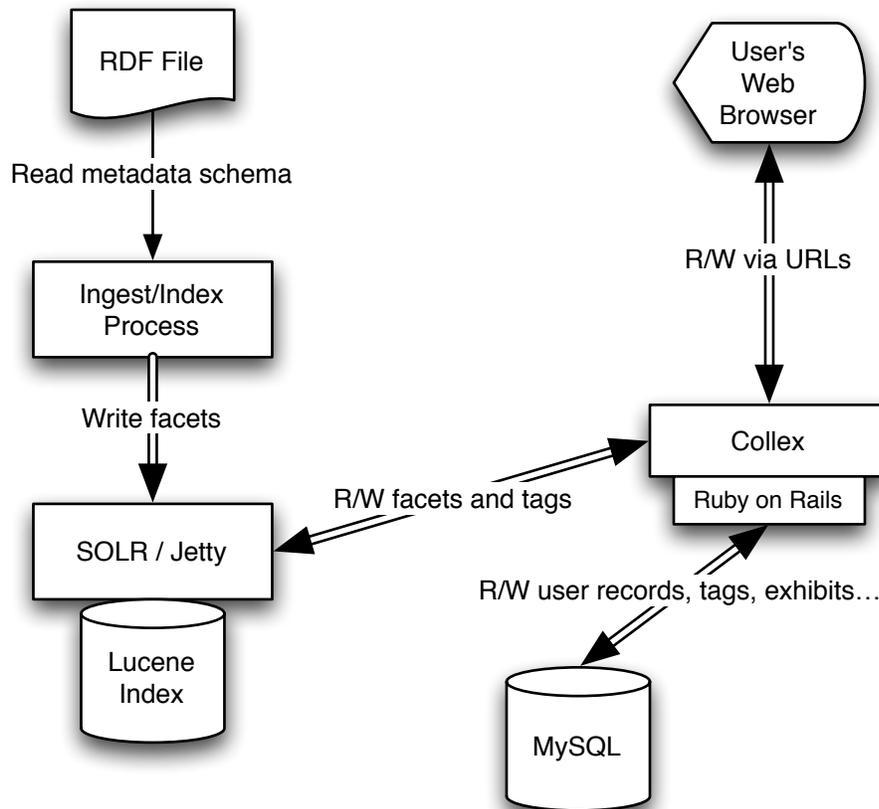


Figure 1: Simplified diagram of system components

3.1.1 Ingestion/Indexing

The indexing process is handled by a component dedicated to that task. This component opens RDF files located in the specified directory, parses the individual elements of the metadata schema, checks for errors, and writes valid data into the Lucene index (via Jetty) as the facets of documents.

In order to handle *FiC*-specific RDF files, the indexer code (written in Java) had to be rewritten significantly to handle a different set of metadata elements, data validation conditions, and date formats. The behavior of the Lucene index and the format of the data stored within it is controlled by an XML specification file which also had to be modified to specify the data formats used by *FiC*.

3.1.2 Collex

Collex is the abstract name for the suite of services and libraries that handles URL requests. It relies upon the Lucene index to store the facets and user tags, and upon MySQL to manage the other data storage needs (a cache for facets and user tags was later added to MySQL as well). *Collex* is built upon the foundations of Ruby on Rails for the framework of web services (mapping URLs to code components, AJAX-functionality, etc.).

As *Collex* evolved in the UVA ARP Lab to address the needs of *NINES*, there was little incentive to separate out what was specific to *NINES* from the general functionality of the system: explicit names of facets were hard-coded throughout *Collex* for data storage and retrieval, webpage generation, and so on. This collaboration represented the first extended opportunity for the ARP Lab to revisit issues dealing with the abstraction of *Collex* services independent of the set of facets specific to a particular application. Our first strategy was to identify those code files in which facet names appear, to prevent their propagation as much as possible, and to parameterize facet values wherever possible (see <http://>

faustroll.clas.virginia.edu/ARPwiki/index.php/CollexGeneralizationIssues). It is anticipated that future work to generalize the *Collex* engine at the ARP Lab may benefit from this process in the future as other communities express interest in exploiting the system's many useful features.

3.2 Visualization Features

3.2.1 Atlas

The Atlas is implemented by a relatively simple Ruby on Rails controller and view that render an HTML webpage with an embedded GoogleMaps viewer. The data points for the items on the map are also embedded in JavaScript on the webpage that initializes the GoogleMaps viewer.

In comparison to the complexities of installing and configuring a dedicated geoserver, GoogleMaps are a simple and light-weight means of providing online maps that does not burden the *Collex* server. All that anyone who wishes to reuse the FtC code needs to do is to obtain their own Google key and insert it into the atlas view (in `web/app/views/layouts/atlas.rhtml`).

3.2.2 Timeline

The Timeline is implemented by a relatively simple controller and view that calls the Timeline widget provided as Open Source software by the MIT SIMILE project (<http://simile.mit.edu/>). Because the Timeline widget is not implemented on an independent, external web server, the data for the display is provided by means of a *Collex* web service which outputs data in XML format.

3.2.3 Graph

The Graph is implemented by use of the Gruff graphing library (see <http://gruff.rubyforge.org/>) which in turn requires that ImageMagick and RMagick be installed on the *Collex* server. I chose Gruff because it is written in Ruby, has a simple API (Application Program Interface), and produces elegant visual displays of several types.

Unfortunately, the Gruff code had errors which I had to debug and remedy myself, perhaps due to characteristics of the data not common in other applications. This necessitates incorporating the modified Gruff code into the FtC code base in the Subversion code repository.

4. Challenges, Difficulties, and Successes

The following sections discuss some of the challenges and difficulties experienced during the course of the project, as well as some of the accomplishments of the project. This discussion is meant to explain why certain decisions were made, particularly in the hopes of informing future work of a similar nature.

Certainly one of the greatest hurdles to overcome was the time necessary to become conversant with a number of new tools and technologies: Subversion (the code repository system), Lucene (especially the configuration files), the Ruby programming language, the Ruby on Rails framework, and the Aptana integrated development environment were all new to me and progress required at least a modicum of acquaintance, and sometimes much more, for project development.

There were a number of other complicating factors, however, that were much more difficult to control.

4.1 Institutional Difficulties

4.1.1 Communications

I often found it awkward to be working remotely with code still under development for which documentation was scarce. Programmers at the UVA ARP Lab work in close contact with one another and did not have other pressing reasons for keeping external web resources updated. Although the ARP Lab has a Wiki for the purposes of documentation (see <http://faustroll.clas.virginia.edu/ARPwiki/>

[index.php/Main_Page](#)), it did not always reflect the current state of the code or lab practices. The visits I made in person were therefore of immense value to me.

4.1.2 Data collaboration

I spent a great deal of time initiating contact with online data holders in the hopes of getting contributions of metadata (in the form of RDF files). I am currently in discussions with the Digital Library of Core Materials on Ireland project based at the University of California, Berkley (http://www.jisc.ac.uk/whatwedo/programmes/programme_digitisation/ireland.aspx) about receiving metadata contributions from their work. A few other institutions are interested in principle and may be able to make contributions in the future. Many of my efforts, however, proved ultimately unsuccessful for a number of reasons: a lack of response to communications, a lack of personnel or resources to carry out the necessary processes, or a lack of the appropriate technology (e.g., persistent and unique URLs, which are of vital importance for *Collex*, are not used by some digital repositories).

4.2 Design and Conceptual Difficulties

4.2.1 Code stability

Unravelling the *Collex* code once and modifying it to handle *FtC* specifications would have been challenging enough; as it happened, the *Collex* code and design has been constantly in flux and went through several significant transformations during the time in which I was working on *FtC*, including a renovation of the Lucene technology, the creation of the Exhibit Builder, the creation of a caching system, and an update of the *Collex* code for Ruby on Rails 1.2.6 (updated from 1.2.3). Of course, given the speed at which computer technology moves, no code base can expect to be stable for any great length of time without becoming obsolete.

4.2.2 Multiple code bases

NINES-Collex and *FtC-Collex* were split into two independent branches of the Subversion code repository; whenever significant updates to *NINES-Collex* were made, I copied them into a directory on my development computer and then selectively merged relevant portions into the *FtC-Collex* code base, making modifications for *FtC*-specific cases when necessary. The process of updating and merging code was laborious and time-consuming but necessary, given the differences between the applications.

4.2.3 Documentation

The pace of my work on the software, especially in the early stages, was also slowed down by insufficient documentation regarding aspects of *Collex* or the software related to it. Documentation was sometimes created after I requested it.

Programmers may feel that times does not allow them to document their code in detail, or that good name choices make their code “self-documenting,” or that the use of their code (whether by test units or the application itself) is documentation enough; a newcomer to the internals of the software (as well as the language itself) may feel otherwise. The shortage of documentation also left me to resort to pleas for help over email and increased the strain on their resources available to me.

4.2.4 Proprietary metadata design

The *FtC* metadata schema was designed with visualization extensions to *Collex* in mind, as well as particular ways of classifying and analyzing the data which are specific to Celtic Studies and not part of the standard Dublin Core Metadata Initiative. This sets a “high-bar” for entry into the system, requiring human intervention, a sophisticated automated procedure, or the elimination of potentially useful facets. The LongLat element, for example, is required in order to place an item on the Atlas, but is missing for many

collection items; the Period code, which defines early historical eras according to developments specific to Celtic art, is not a standard used outside of Celtic Studies (although it can be easily calculated from a date).

4.3 Technological Difficulties

4.3.1 Multiple environments

I did my development work on a Mac OS X computer but deployed the system on a iBiblio host which runs Red Hat Linux. The various components of *Collex* are able to run in these different computer environments, but they did not always do so consistently.

The most vexing of the problems I encountered related to the support of UTF-8 characters. Supporting non-standard ASCII characters is crucial because of the needs of languages other than English, particular Celtic languages with vowels with length marks. Texts needed to be able to be created on one computer and transferred to another with consistent results. Unfortunately, despite considerable time spent by several people, there were problems with the use of UTF-8 encoded files that could not be resolved. One result was that the facet value “La Tène” (standard spelling) had to be changed to “La Tene” to avoid the use of UTF-8 character encodings.

4.3.2 Version dependency

During the course of development several problems occurred because code was written to operate on or with a version of the operating system, programming language, programming environment, or code library other than what was loaded. Such dependencies ought to be stated explicitly and clearly in the documentation where all users can check for prerequisite conditions.

For example, the version of the Mongrel server loaded on iBiblio was unreliable in its rendering of HTML pages on Linux, a problem that did not happen for the same code on Mac OS X. This was probably due to dependency problems (the problem has gone away after upgrading the operating system and the relevant server components).

In the fourth quarter I upgraded the operating system on my development computer to Mac OS X 5 (Leopard), which caused a number of items to crash, unexpectedly costing me over a week of development. As I did this upgrade before other members of the ARP Lab and some of my problems were specific to *FtC* (i.e., *RMagick* would not compile), I had to spend time solving these problems on my own.

Large software projects are typically compromised of a set of components created independently. These linkages create a complex set of interdependencies that can be easily broken if unstated conditions about one component in this chain of dependencies do not hold true (e.g., a library relies upon a particular version of another library). When bugs occur, they can appear far from the source of the problem and be hard to trace.

4.3.3 Open Source software reliability

The Open Source software movement has been a great boon to many communities in many ways, but the quality of software code is not of uniform standard. Users have to be able to provide their own technical support for Open Source software, which can entail configuration or modification at many levels of detail and expertise.

For example, the Gruff graphing package was ideal in many ways, but it had logic bugs that appeared in my application. It was necessary for me to debug and modify the code. It was advantageous to be able to exploit the Gruff code base rather than writing my own graphing library from scratch, but I did not expect to have to concern myself with its inner workings.

4.3.4 Debugging

Debugging code in a multi-tiered (or server-client) architecture is convoluted and thorny. There are multiple layers of interaction between various components outside the *Collex* engine, such as when GoogleMaps are

involved. The rationale for embedding the data points in JavaScript on the Atlas webpage (explicitly creating GMarkers for each item) illustrates some of these points. The *Collex* server can determine the identity of the user by reading the session id (stored in a cookie accessible to the user's web browser); this links user identity to session variables, which in turn allows the server to retrieve the current constraint set (saved in the session variables) and resulting items.

The preferred solution for serving geographical data would have been to create a URL on the webpage to pass to GoogleMaps which would in turn request a "geodata" service from *Collex*. That design was not possible, however, given that the item set (resulting from the current constraint set) is specific to the user session and user session variables are not visible to the GoogleMaps server (nor would it be safe to pass the session id over the internet). This scenario is depicted in the illustration below.

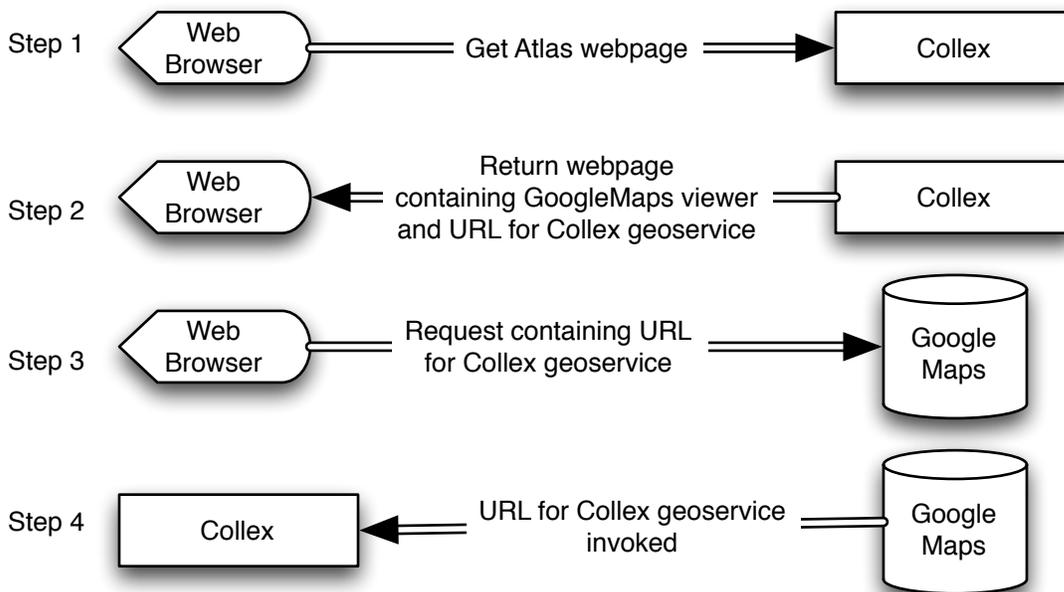


Figure 2: Web requests in a flawed geoservice scenario

At the point at which the *Collex* geoservice would try to serve the request issued at Step 4 above (and create XML-formatted KML data), it would not be able to connect the request with any user session and corresponding item set. While this may seem to be a trivial point, it underscores the complexity of making independent components operate over the internet in a secure fashion. Determining the source of such problems when so many features seem to be provided "invisibly" by Ruby on Rails actually took me considerable time.

4.4 Design Idiosyncracies

Metadata design inevitably involves choosing the appropriate set of tradeoffs. These tradeoffs need to be understood by those creating metadata for *FtC-Collex*, using the system in a sophisticated way, or considering their own design.

4.4.1 Range overspill

The Period and CountryCode elements are required in every metadata entry; each can have only one value (for reasons explained in section 3). While this does not present a problem for most items there are several types of items for which this is problematic, such as items whose date range spans multiple periods (such as definitions of terms or concepts with great longevity, archaeological sites in use for a very wide span of time), or items that belong to multiple countries or none specifically (such as definitions of Common Celtic

terms which were current in any countries in which Celtic languages were spoken). I have suggested several strategies to address these shortcomings, none of which is entirely satisfactory:

- No country code: the code ZZ can be used to indicate that the item is not associated with any specific country.
- Multiple representative: define one item for each relevant period or country.
- The essential representative: define the item according to the characteristics most representative or “essential” to our understanding of it, or according to its formative period or location.

4.4.2 Archetypes and Variations

Textual items present challenges: there are inherent complexities in representing items from a literary tradition³ with a long history of transmission, recension, variation, edition, and translation. While scholars may speak of a particular tale such as *Táin Bó Cuailgne*, for example, it gets recorded in many variants over a long period of time: it may be written in a number of manuscripts in different recensions; editors may create diplomatic editions of these texts; translators may render these editions into languages other than the original; and so on. The idea of the *Táin Bó Cuailgne* – a narrative with a particular plot and set of characters, motifs, settings, and so on – is distinct from the many different forms in which it can be manifested, each of which can be distinct from the others.

I have suggested one solution for dealing with this. An item can be defined and represented in “archetypal” form by an entry whose *ItemType* is *Definition*, and whose other elements (such as *Date*, *Period*, *Country*, and *Language*) represent the hypothetical “ur-text” of the abstracted form of the item. Each edition of the item can be represented by an entry whose *ItemType* is *Text* and whose *Date*, *Period* and *Role* elements reflect the work of each performer, editor, or translator. This scheme allows for the existence of multiple editions and translations – clearly an unceasing activity of literary scholarship – independent of the original narrative abstraction. It also recognizes that a modern edition of an older text is an entity in its own right and a product of a particular individual, time, place, and methodology.

5. Future Research and Development

5.1 Collex Generalization

The UVA ARP Lab programmers are currently working on issues of generalization for *Collex*. This collaboration has been highly beneficial in highlighting the embedded *NINES* code in *Collex* and the need to remove domain-specific information. This development work is still underway.

It should be pointed out that the visualization extensions I have developed are specific to *FtC*: that is, they check explicitly for particular facets and facet values. It would be desirable for a generalizable *Collex* engine to incorporate the visualization extensions and allow them to be generalized in a consistent and similar manner.

5.2 Internal referentiality

An unexpected complexity emerged from the representation of textual items: while each entity in the *Collex* data base must be represented by a single set of metadata values, texts contain references to entities which are not captured by the metadata for the item itself but which the user may wish to locate or ascertain.⁴

³ These issues exist whether the medium of the literary tradition is oral or written. Oral narratives, for example, are identified by entries in the Aarne-Thompson Tale Type Index but exist as a number of different eco-types in different contexts and many different variants when recorded in specific performances.

⁴ *Collex* does support full-text search, which can identify any textual phrases contained within the item, but this does not exploit any of the browsing or visualization features that function on basis of the metadata itself.

For example, a modern article of literary criticism written in the year 1990 in Wales may have as its subject an Irish medieval text; there is nothing in the metadata for the article that links it to the item(s) it references or discusses.

While *Collex* was not meant to “look inside” of an item (beyond textual search), and while users can tag the item with labels that indicate such references, future research may wish to analyze the content of textual items (which has repercussions for search, representation and user interaction). The Perseus project has been developing tools that parse texts and look for named entities (such as people and places) which are related to this same issue (see for example <http://www.dlib.org/dlib/july00/crane/07crane.html>). I understand that the Digital Library of Core Materials on Ireland project is also planning to address this issue.

6. Recommendations

There are formidable technological challenges which still confront any digital humanist wishing to make use of the latest computer-based tools. Despite certain areas of progress, the complex networks and chains of dependencies between software components and the contexts in which they operate can create debugging nightmares for all but the most seasoned system administrators and deter digital humanists from their goals.

The NEH can play an influential role in promoting and maintaining high standards in the software development which they fund; promoting high standards can increase the payback to many communities and users of software and its products. Documentation of code is an under-valued but highly important activity that can have a marked impact on those communities that wish to make later use of software resources. The software development process often neglects to allocate sufficient time and resources to the parallel creation and maintenance of documentation.

Scholarly appraisal of the human costs of software development has recognized for decades the importance of clear design and explicit documentation, but I would like to highlight several aspects of this project (shared by others of a similar nature) which made programming and debugging without full documentation particularly vexing:

- *Non-typed Objects*: In a typed object-oriented programming languages (such as Java) the types of parameters passed to methods/functions is clearly defined. Ruby takes pride in being non-typed; while this allows for run-time dynamism and flexibility, it makes deciphering and debugging code without explanatory text much more elusive.
- *Multiple-technology interaction*: The current generation of web-based user interfaces relies upon several discrete but interacting technologies which rely upon one another. For example, the web-page is rendered with HTML and CSS but uses Javascript for interactivity, which also interacts with Ruby on Rails code on the server.
- *Multi-tiered message-passing*: Web-based services essentially work by passing messages between different systems: the user (using a web-browser) requests an item from the web-server, which may make a request from another service, or return an HTML page to the user with embedded service requests, and so on. The repercussion on design and debugging is that software is asynchronous and the flow of control is no longer a simple linear sequence.
- *Non-linear code invocation*: Software that operates as a web-based service is invoked by specific patterns in URLs; the software may itself generate URLs to indirectly invoke other code far removed from it. This adds to the complexity of predicting and analyzing the flow of control.

I would like to suggest that documentation can help to address the complexities listed above by including the following information, at appropriate levels of granularity.

6.1 Object / Component / Unit

At the highest level of granularity, documentation could include:

- Date created and last updated, current version id number
- What language and environment (including version information) this code is written to run in
- What objects/components/units this code is dependent upon (including version information)
- How this object/component/unit gets invoked and by whom

For example, a Ruby object might begin with the following documentation block:

```
## Atlas Model 1.3
## 2008/02/03: Created by Michael Newton
## 2008/02/13: Added use of SortObject
##
## Written in Ruby 1.8 for Ruby on Rails 1.2.6
## Dependent on SortObject (version 1.4), gruff (version 0.1.2)
## Invoked by URLs generated on Collex search page (by link_to)
```

6.2 Method / Function / Procedure

At the level of individual methods / functions / procedures, documentation at the top of each could include:

- Purpose of method/function/procedure
- Input parameters: what they are and what they should include
- Return value(s): type and contents
- Side effects: any changes to variables other than input/output
- How this method/function/procedure gets invoked and by whom

For example, a method for a Ruby object might begin with the following documentation block:

```
## GetXML: Called to create XML output used by GoogleMaps
## Input:    <opts> hash of optional tags; keys are facet names
## Returns:  stream in XML format, or ""
## SideFX:   <num_objs> in Atlas class is updated
```

An action in a Ruby on Rails controller might begin with the following documentation block:

```
## GetXML: Called to create XML output used by GoogleMaps
## Input:    <params[id]> is the id of the object to render in XML
##           <params[max_size]> integer [1..20] which sets limit on size
## Returns:  stream in XML format
## SideFX:   <session[:num_objs]> is updated
## InvokedBy:URL generated by Atlas#page which embeds object id
```