Innovate: Journal of Online Education

Volume 3
Issue 1 *October/November 2006*

Article 6

11-1-2006

Looking Toward the Future: A Case Study of Open Source Software in the Humanities

Harvey Quamen

Looking Toward the Future: A Case Study of Open Source Software in the Humanities

Looking Toward the Future:
A Case Study of Open Source Software in the Humanities

by Harvey Quamen

The Masters of Arts in Humanities Computing (Huco) program was established at the University of Alberta in 2001. Since then, we have been training young humanities scholars in the intricacies of multimedia, markup languages, project management, research methods, and even game theory (Gouglas et al. 2006). Many of our students—like those in other branches of the humanities—have bright futures in education, law, public relations, marketing, and other disciplines. Increasingly, our students find an outlet for their skills in non-profit organizations, many of which have little to no budget for computer technology or support. One of my goals within the Humanities Computing program has been to introduce open source software (OSS) and its politics of community and sharing into my courses. Open source's wide array of software provides computing solutions for numerous contexts.

When illustrating the advantages of OSS to my students, the exemplar that I cite is a project on which I myself have been working for a few years now—a manuscript-tracking database for *English Studies in Canada* (ESC), a journal on whose staff I serve as an associate editor. The ESC Database, as it has come to be called, is an illustrative case study in that it reveals both the successes and challenges of using OSS in academic contexts that are severely limited by staff and by budget. The project itself is still in progress, but far enough along that a scorecard of successes and failures might be instructive to others in similar situations.

Background: *English Studies in Canada*

English Studies in Canada (ESC) is a medium-sized, peer-reviewed, quarterly academic journal devoted to "any topic which falls into the disciplinary purview of 'English studies' broadly understood" (ESC Digital 2004, "Call for Papers") and is the official journal of the Association of Canadian College and University Teachers of English (ACCUTE).

I joined the editorial team when ESC moved to the University of Alberta in 2003. At that time the team was understaffed and overworked, and the journal itself under-read and overdue. As our first priority, we raised ESC's profile by redesigning the hardcopy ESC. We began to typeset the journal in-house using Adobe InDesign in order to gain control and to save money. We gave ESC a Web presence, and we recruited new subscribers and lobbied for more article submissions—all facets of ESC's public persona.

However, other problems lingered in ESC's private persona. Even though the editorial staff all worked in the same building, communication among us was slow and sometimes inaccurate. Each new manuscript submission generated an overstuffed, legal-sized file folder of documents, but typically only one person at a time accessed the file—and sometimes important files lay trapped in locked offices when editors were off campus or out of town. Hardcopies, communications with authors and with potential peer reviewers, reminders to reviewers to speed up their reviews, inquiries from authors, and the eventual results of the adjudication process itself were not always accessible to the rest of the team, which led to increasingly egregious mistakes: we lost e-mail addresses, some correspondence never made it into the files, and we sometimes forgot that peer reviewers had neglected to send in their reports. I once typeset a supposedly finished article only to be notified by a puzzled author that acceptance of her piece was contingent on making certain revisions that she had not yet even begun.

Our biggest problem was that editors had difficulty finding peer reviewers. Our team scoured our parent

organization's membership directory but found the descriptions about member's various areas of expertise to be unhelpful; sometimes the entries were just vague, sometimes they were ambiguous, and sometimes they were downright wrong. The editors begged for a better way to find peer reviewers.

An online database seemed the obvious answer to all of these problems. With such a tool, everyone could get real-time answers to questions and could do so no matter whether one was working on campus, at home, or in the midst of a research trip. Most of us chose not to work in the tiny, cramped ESC office—we used it mostly as a central location for files and other resources—and so a single desktop computer running a database application like Microsoft Access did not seem to suit our style. Like many other academics, we found ourselves working from our campus offices, from home, from the library, from coffee shops, and elsewhere. In considering how to put our database online for easy Web access, we explored the potential of open source tools; we finally selected MySQL and PHP as the best solution for our staff needs.

MySQL and PHP: Open Source Tools for Variable Web Content

One of the problems facing traditional Web sites coded in HTML is that each Web page becomes static and difficult to change. Software like Macromedia's Dreamweaver (recently acquired by Adobe) or Microsoft's Frontpage can help make Web design a bit more like word processing, and they have helped make Web design less difficult for people who do not wish to deal with specialized code. However, these tools still require ongoing maintenance operations that entail substantial investments in time and personnel. Moreover, the use of standard Web designing software is not a viable option for more complex, dynamic Web sites. The sophisticated search engines on sites such as Google or Amazon, for example, deliver new, unique content for each search one performs; in contrast, if one relies on tools such as Frontpage or Dreamweaver, it is simply not possible to code every possible combination in advance of the off chance that someone might search for those precise results later. Stronger tools are necessary in order to build HTML Web pages out of variable content.

MySQL and PHP are such tools. MySQL, which bills itself as "the world's most popular open source database," can store, organize, and retrieve vast amounts of data stored on a Web server. PHP, a Web-scripting language invented by Rasmus Lerdorf in 1995, acts as the ambidextrous intermediary within such a database environment. On the one hand, PHP can query the database, retrieving records as necessary; on the other hand, PHP can also wrap the data in HTML in order to create Web pages on the fly. By allowing designers to streamline the process of information management while standardizing the process of Web site design, these technologies offer a vital resource for any organization. For academic journals in particular, which must regularly sort, distribute, and update large quantities of information, this open source technology has much to offer. I know first-hand that many other academic journals are in the process of implementing this solution for their own needs; however, for organizations that have not yet taken this step, I believe that an introduction to some of the features of the ESC Database may be of special interest.

The ESC Database: Development and Design

In the summer of 2004, I started using these software tools to write a small Web-based database that would help us track our manuscripts. Our business logic (i.e., the editorial process itself) still was not clear even among ourselves, and so I sometimes relied upon my best guess for handling various situations that had not yet occurred in real life. Yet as occasionally happens, writing the database application served as a way to formalize our own production processes; each major decision about the design of the database essentially entailed a decision about how to structure, distribute, and manage the editorial cycle in such a way that there was an option for every eventuality. For this reason too, the project became larger than I expected and required a greater initial investment of time and energy than I had anticipated. I did not imagine that the project would last any longer than the immediate summer, but I spent the following summer (and nearly every interim holiday) incorporating a fuller range of functions to manage our editorial process. By the time the database finally came online in February 2006, the tiny project had grown to 34 MySQL tables and over 6000 lines of PHP.

While the ESC Database still remains open for future development, the features and functions in the initial version may illustrate the value of MySQL and PHP for other academic journals. First, the database included comprehensive listings of all manuscripts, authors, and reviewers as well the entire membership directory of our parent organization (ACCUTE) and a listing of over 400 department and university addresses. More importantly, the MySQL framework allowed me to organize the Web site to provide convenient searching, navigation, and linking across all different fields of information contained in the database. For example, I was able to incorporate a variety of search techniques based on the major categories of "objects" contained in the database (Exhibit 1). The database design, in turn, allowed any user to establish a range of relationships among different objects by embedding links from one object to the next, which established a cross-referencing network for the search engine (Exhibit 2). This cross-referencing network was further expanded through the keyword functions supported by the MySQL system; the keywords were particularly crucial in helping the ESC Database fulfill the needs of editorial staff. Users could search for reviewers via keywords that designated particular areas of specialty, search for manuscripts via keywords that designated a given period or focus, or generate lists of potential reviewers for a given manuscript based on shared keywords (Exhibit 3). Moreover, I was able to incorporate a Keyword Thesaurus that permitted both an "exact" mode and a "fuzzy" mode of searching, which gave greater flexibility to users as they searched for manuscripts (Exhibit 4). Needless to say, such capabilities would have been far beyond the scope of any HTML-based Web authoring tool.

The MySQL framework also allowed me to include additional features and functions to support the needs of editorial staff. For example, I designed the database to allow for the creating and archiving of specialized reports to assist editors in their tasks and decisions (Exhibit 5). I also incorporated a variety of administrative pages to provide guidance regarding topics such as HTTP Basic Authentication as well as a function that would allow me to post announcements to users that they would see the next time they logged in (Exhibit 6). In this fashion the database not only provided easy navigation through a complex network of information but also supported timely, convenient communication among members of the editorial staff.

Assessment and Future Plans

Having a no-cost, centrally located means of communication has been essential to our work at ESC. While our operating budget is drawn from subventions from ACCUTE as well as from the Social Sciences and Humanities Research Council of Canada (SSHRC), those contributions cover less than 50% of our annual expenses. We therefore see cutting costs and increasing efficiency as a matter of survival rather than just a means of improving our financial health—and to this end, the ESC database has provided a resource that serves our needs within our budgetary constraints. Meanwhile, the database has also helped provide a technological foundation for our distinctive working environment. Since we all volunteer our time, we work on ESC projects at different times and even at different places. Through the ESC database, editors now can see when new manuscripts enter the editorial cycle, even if they are not communicating face-to-face or sitting in a meeting, and they can all get a detailed account of ESC's workload at any given moment.

While these are significant benefits, the project is still very much a work in progress. The ESB Database is still in a frenzy of data entry; graduate student assistants are combing through our files, aggregating data, and inserting it into the system. After our data entry phase is finished, however, I plan to implement a few changes. First, I underestimated our need for a document management system, and rather than changing the database itself, we are currently considering the possibility of supplementing it with Knowledge Tree—an open source application that will help us keep track of document revisions. Second, the entire database needs a rewrite; this may seem a drastic measure, but it is a necessary one. Even with my best intentions, I have made both coding mistakes and poorly considered decisions in version 1.0 of the ESC Database. For example, creating two separate objects for author and manuscript still feels somewhat inefficient, and the editors have found this aspect of the database a bit counterintuitive. Third, the programming code is voluminous enough that moving to an object-oriented programming style would help simplify the program, reduce errors, and allow new features to be added more easily. Were I to begin again today, in fact, I would opt for object-oriented programming immediately. Open Source advocate Eric Raymond (2001) reminds his

readers of Fred Brooks' mantra—"Plan to throw one away; you will, anyhow"—and explains: ". . . you often don't really understand the problem until after the first time you implement a solution . . . So if you want to get it right, be ready to start over at least once" (25).

Such changes to the database would not only improve it for our own purposes; they would also make it a potentially valuable application for open source release. The key factor in this regard is that the database needs to accommodate different editorial procedures and workflows. As its name implies, the ESC Database is tailored to the ESC editorial process. It uses our terminology, embodies our assumptions about how the editorial process works, and takes no account of any procedures that ESC does not implement. For example, ESC has abandoned a traditional editorial decision—"Revise and Resubmit"—in favor of a new category that we prefer: "Accept with Specified Revisions." Any journal using "Revise and Resubmit" in its editorial process has no way to reinsert that category without first reprogramming the entire database, and this feature alone might prevent other journals from using it. Consequently, an open source version of this software will need to formulate and document strategies for its own redesign in different contexts, perhaps by using a variant or subset of RuleML or Business Rule Markup Language (BRML) or perhaps by inventing a new, small, fleet-footed XML-based schema.

In making such adjustments to the ESC Database and eventually offering it as open source software, I believe that it can serve as a prototype for other academic journals to adopt and develop for their own needs. While the staffing policies and editorial processes of academic journals are not all the same, it remains the case that the overwhelming majority of such journals rely substantially on the work of unpaid graduate assistants, interns, and volunteers rather than a cohort of full-time, salaried employees. At the same time, the nature of the work—tracking manuscripts, assigning reviewers, corresponding with reviewers and authors, providing feedback to authors, and preparing manuscripts for eventual publication—is very demanding, and under such circumstances it becomes all the more crucial to ensure convenient access to information while supporting efficient, structured collaboration among all staff members. For journals that have a large base of subscribers and substantial financial support from their hosting institutions, proprietary software applications may already offer the best solution for addressing these challenges. However, for journals that remain constrained by fewer resources, particularly journals in the humanities, open source applications can provide a vital option for achieving the same benefits as they seek to fulfill their own mission.

Conclusion

Since at least the 19th century, the humanities have defined themselves in opposition to the sciences (Mary Shelley's *Frankenstein* is a requisite text in that regard), and academics in this field have become unaccustomed to turning to technology for help. However, the premise of the Humanities Computing program where I teach is that the time has come to tear down that wall and look to the other side to find viable solutions to problems we currently face. Traditional humanists often look for solutions in the past, but it is time to start looking hard at our futures. Open source can both provide our technological solutions and foster our ideological values of collegiality and community, but the onus is currently on humanities scholars to become more computer literate and to educate ourselves about the benefits of open source. That is not only an important lesson I hope to instill in my students, but a lesson that I have already learned in my work at ESC.

References

ESC Digital. 2004. Home page. http://www.arts.ualberta.ca/%7Eesc/ (accessed September 30, 2006).

Gouglas, S., S. Sinclair, O. Ellefson, and S. Sharplin. 2006. Neverwinter Nights in Alberta: Conceptions of narrativity through fantasy role-playing games in a graduate classroom. *Innovate* 2 (3). http://www.innovateonline.info/index.php?view=article&id=172 (accessed September 30, 2006).

Raymond, E. S. 2001. *The cathedral and the bazaar: musings on Linux and Open Source by an accidental revolutionary.* Cambridge, MA: O'Reilly. Also available online at

http://www.catb.org/%7Eesr/writings/cathedral-bazaar/ (accessed September 30, 2006). [Editor's note: The online version of this source revises and adds to the print source.]

COPYRIGHT AND CITATION INFORMATION FOR THIS ARTICLE