Mid-Project Report to the Center for the Public Domain: Open US Code

As we pass the six-month mark, we are happy to report that progress on the Open US Code project is substantially on track, with significant progress to relate.

Technical direction and progress

Implementation has been informed by the work done on the Enact system in use in Tasmania (documented at http://elj.warwick.ac.uk/jilt/00-1/arnold.html). Early on, we recognized that the Structured Information Manager (SIM) software, developed by the Royal Melbourne Institute of Technology and used in Enact, would provide an excellent repository database and upper-tier delivery system. The decision to use SIM has in many respects set the direction for the rest of the project.

An overview of the process will be helpful in understanding where we are at this point. The problem of constructing a point-in-time version of the US Code can be broadly broken into two activities: constructing software that will transform the text of the Code as delivered by the House of Representatives into a series of well-formed XML documents suitable for importation into a database repository (SIM), and building a series of Web applications that deliver point-in-time versions assembled from that database on the fly. At this point we are nearing the end of the first of those activities while beginning work on the second. Along the way we have encountered further technical possibilities that when implemented will increase open access to the Code.

In some ways the first part of the process, the conversion of the House of Representatives format (hereafter 'HR format') into well-formed XML, is the more daunting of the two steps. It is daunting principally because of the uneven nature of the input data; the HR format gives the appearance of uniformity and regularity but in fact suffers from problems of structural and editorial uniformity. We decided, then, that the best technical approach would be to first create software that would move the HR-formatted data into XML using a loose, intermediate DTD that would regularize the data while remaining flexible enough to handle anomalies. At this first stage, we would primarily be focused on representing the structure of the document accurately but generically; tagging would be somewhat abstract. In a subsequent step or steps we would transform the "loose" XML into the final format to be fed to the repository, using a DTD with a much more concrete, human-readable tagset.

During November and December of 2000, a team of eight graduate students in computer science acting under the supervision of LII Co-director Thomas Bruce created a software system to accomplish this step, which they dubbed LDMS (for "Legal Data Markup System"). LDMS is written in Perl; the project is exhaustively described at http://leda.law.cornell.edu/~LDMS/ . Documentation at that site includes requirements documents and slides illustrating problems with various formats as well as the structure

and design of LDMS. Particularly relevant to our discussion here is the requirements document (in PDF at http://leda.law.cornell.edu/~LDMS/docs/SoftwareRequirementsSpecification-2000-11-28.pdf).

Work on the second-stage transformation (from LDMS output to XML suitable for importation into the repository database) is now underway with Bruce and Sylvia Kwakye (from the original LDMS team) as collaborating authors. We are using XSLT as the vehicle of choice, and anticipate completion in about one month. There are significant windfalls from this approach. In addition to creating XML to be imported into the point-in-time system, we should shortly be able to replace our existing HTML offering with a much better-formatted and accurate version, make "print-friendly" consolidations of arbitrary size available in PDF, and create alternate versions suitable for delivery to a variety of personal devices to the extent that it makes sense to do so (some sections of the Code, such as the Federal Rules of Evidence or Civil Procedure, or the Uniform Code of Military Justice, are much more suitable for this form of delivery than others).

Next steps

Within the past week, we have hired a programmer with significant SGML and XML experience to work on the next step, the incorporation of the second-stage XML into SIM and the construction of the SIM-based applications that will deliver the final, point-in-time product. Obviously it is much too early to say very much about that work as yet, but we are quite happy with the qualifications of this team member and anticipate a happy outcome.

We will soon being acquiring the hardware to be used for delivery and building the SIM server that will run on it; we anticipate that this will be done smoothly and without difficulty over the next month or two. At the same time we are moving to acquire the historic data we need to provide the retrospective parts of the system.

The existing LDMS web site provides thorough documentation of that aspect of the project, and we would like to create an overall project website incorporating the LDMS information into the larger project context, reporting and demonstrating our further progress and including ongoing project documentation such as this report.

Availability of historical data

Transformation of the historical data needed to create a retrospective system remains to some extent an unexplored area. We know that we have access to at least two years' worth of US Code data in the current HR format, and we have tentative agreement with individuals in the House to provide us with data going further back. We are not certain about the format in which this historic data will come. We may encounter difficulties in

dealing with retrospective data if it cannot be made available to us in the current HR format. Surely these problems can be overcome by writing further transformation software, but that might be the cause of some delay.

Funds

At this point we have expended only those funds represented by Bruce's effort on the project. In the next phase, we will be paying for a salaried programmer and for project hardware; we do not anticipate any problems with data acquisition that would require expenditures.

Timetable for Final Report

Our original proposal contained no projected completion date for this work. Our budget memo estimated it at a year and a half of programming effort, which still looks right. While we experienced some delays in bringing our full-time programmer on board, owing primarily to the difficulties of hiring such a person at academic salary in a tight labor market, we are progressing well. Some of the delay was offset by the availability of graduate students to work on the LDMS project. Nonetheless, we do not foresee being finished at the date set by the grant letter for our final report.

Our assumption is that you would like us to submit our final report on schedule with full particulars on what we have accomplished by November 2001. If you would rather have us delay the final report until we have a functional system, that would, in all likelihood, put the report off until early 2002. If that is your preference, please let us know.