

Information Management Initiatives, Teaching, Research and Collaborations

C. Frank Starmer



Institutional Problem Solving with Web-accessible Databases, Tool-based Software Architecture and something new:

Crafting Software Inductively

We are evolving a paradigm for enterprise data management based on providing MUSC students, faculty and staff with a software infrastructure composed of web-based tools that enable them to solve their own problems as effectively away from their desk as when they are at their desk. The main idea is to build an internet-centric problem solving workspace from tools (both open source and proprietary). Our approach is based on constructing this workspace on top of an IT software infrastructure that promotes access to information resources (through our web portal, <http://my.musc.edu>), convenient transport of data from repositories to your desktop and rapid prototyping of browser-accessible applications. Our approach is client-centric, problem-driven, tool-based and reflects our commitment to creating an infrastructure that provides seamless transport of information between any two points. Instead of enterprise-wide systems, we are addressing institutional data management needs with compositions based on web-based tools glued together with a scripting language. Our software development strategy is incremental and based on growing solutions starting with a simple instance, identifying the tools required to enable the processes associated with the instance, looking for common denominators among similar problems and moving on. In other words, we are identifying instances of problems from which we can generalize enterprise-like solutions, but from the bottom up instead of from the top down. The concept of generalization is critical, and provides a metric from which we can measure the utility of a solution. Problem solutions that do not readily generalize are lower priority than solutions with a large leveraging potential. Our strategy is predicated on using industry standard data packaging, transport and tracking protocols in order to facilitate coupling (via an interface) of solutions. The overall paradigm is based on XML(data packages) + HTTP(data transport) + SOAP (activation of remote services). Our tools are designed to address seven critical areas:

- Capture key strokes once
- A high potential for tool-scripted solution generalization
- Package key strokes and displayed data in a portable format (tab delimited, pdf, XML object, etc)
- Transport and track data packages with standard protocols: ftp, http
- Browser-based access to institutional databases - see Web-based access to MUSC resources and mySiteMaker a nifty web interface to a database.
- Applications designed to be "wrappers" around institutional resources. The primary objective here is to loosely wrap browser-based applications around data repositories. Using common interface engines to data repositories minimizes disruption of applications with system upgrades or system replacements. HOWTO make a wrapper, and basic components of a web applications [NEW](#)
- Applications designed around the email model of a workflow manager. Workflow managers provide a mechanism for capturing data and manually or automatically processing forms-based data. See our Broadcast Message Workflow Manager and associated workflow monitor/Workflow Monitor. The workflow monitor facilitates monitoring the flow of documents around the university, the document service time and the search for a lost document.

In addition to the tool-based architecture, we are evolving a software design strategy that focuses on building a software infrastructure that facilitates information transport, an essential enabling resource for MUSC students, faculty and staff. We refer to our strategy as "inductive software engineering". By inductive, we mean that we start by solving one client's problem, looking for common ideas and repeating themes (generalization potential, a precursor of reused software -- in contrast to designing reusable software which may never be reused). The repeating themes become targets for tools and a component of our software infrastructure.

Among our tools are ones designed to facilitate access to and transport from institutional databases, ones designed to capture data on a form, perhaps prepopulated with existing data from one or more enterprise resources: see the IT Lab toolbox . In addition - here is a useful suite of tools to build a web interface to a MySQL database: mySuite and here is an extremely popular tool for building a browser-based interface to MySQL. So who crafts these tools? To deal with implementation, I started the IT Lab and populated it with 7 of the greatest and friendliest folks in the galaxy - see myFolks .

The strategy

We select a problem with an end user, and explore how to solve the problem with existing tools. If we find some tool is missing, we develop a new tool, that fits with extending the range of problems we can solve; thus the inductive component. Emphasis is placed on developing tools that address more than the immediate problem under investigation without destroying the one-function:one-tool concept. One goal is to rapidly prototype a script in order for the user to determine if the solution is pointed in the correct direction. In this manner, there is successive refinement of what the user perceived he/she needs while providing a foundation for tool use and development. Our goal is that when first meeting with a client, by the end of a session, we will have implemented a web-accessible prototype of a solution to their problem.

The results

Our data-entry tools are based on PDF and HTML forms and unix-like filters for electronic signatures, credit card validation, merging data from a database with a PDF template, creating tab-delimited records from a submitted PDF form, placing and extracting data in an HTML and PDF form etc. We use MySQL as a backend database manager and a number of PHP3 scripts to interface the user with the database (PHPMySQLAdmin etc) . We use JDBC as the primary database interface with some use of ODBC (when all else fails). We have recently found a neat trick - to change the mime-type of a text file so that it can be automatically imported, for example, into an excel spread-sheet. Visit myGrants for an example (click on Return Results As and select Excel). This strategy for moving data is simple, powerful, and requires no special installations or maintenance. Another triumph for http protocols. We have built a portal (see myMUSC) to explore distributed content management of our portal. We have also built a web-page registration system to facilitate central coordination of web-site integrity while promoting distributed web-site management. For an example of a tool that is able to manipulate URLs, Try you hand at copy/paste of a table from a web page

- Authentication and encryption solutions for public and wireless access areas [NEW](#)
- A Survey Tool (requires MNA Authentication)
- Linking finance systems with XML objects and ftp transport. [NEW](#)
- The latest in IT Lab-ware: How to make a web wrapper around a program
- The IT Lab Web Camera: Watch our 9am Mondays (EST/EDT) lab meetings
- The MUSC IT-Lab: Problem Solving with Tool-based Information Management
- A simple workflow manager to facilitate moves to implementing paperless processes
- An example of workflow in action with a small demo [NEW](#)
- How it works with another demo [NEW](#)
- Our premier list of active tools [NEW](#)
- The MUSC Portal: Configure your personal web page
- The MUSC Web Registration System (requires MNA authentication)

Hits since March 20, 2002

000338

C. Frank Starmer (starmerf@musc.edu)

Copyright C. Frank Starmer 1998-2001