COMPUTER SCIENCE RETENTION STUDY

Data Collection Utility Prototype

Eric L. Brown
Spring 2006
Objectives
The primary objective of the Computer Science Retention Study is to evaluate success/failure patterns of TTU students taking Computer Science (CSC) courses. Of particular interest is the success patterns of CSC majors and the relation of those patterns to overall retention numbers. The execution of such a study is overwhelming at best. The evaluation team must develop testing means, develop meaningful performance metrics, and be prepared to propose solutions for every problem pattern. While all these steps are important, none can be undertaken without acquiring the raw student performance data. It is this data acquisition, a sometimes financially and temporally draining process which is the primary focus of this project.

Proposal
It is my proposal that a set of tools be developed to deal with the automation problems of collection raw student performance data. It must be emphasized that raw, not processed data, is the desired element as processed data can only be created with the processes described in the Objectives section. These tools must be developed with a Unix mindset in that output to standard out/error must be limited to errors and anomalous conditions. The tools must also be modular so that the individual elements can be reused without running the entire tool. All of these elements will contribute to an ease of programming for the web browser user interface (UI).
The Problem

The problem of declining student retention is not unique to any college or university.¹ It is a common problem with varying roots ranging from learning environment conditions to social integration issues to weak logic or general academic backgrounds. Many institutions are looking to computerized tools to assist in dealing with a issue that has a huge “problem of scale” obstacle. Some look to computerized tools as a means of postmortem analysis while others use them as a means of intervention.²

The CSC Department at Tennessee Tech University is evaluating the problem for a variety of reasons. The primary focus of such studies is to determine how best to help incoming freshman with weak logic and math backgrounds to improve their chances of succeeding. A secondary focus of the study is to assist in the development of long-term and repeating regiments that will repair problems in the curriculum and teaching methods as identified by the evaluation metrics. Such a continuing self-study is also an important element in the ultimate evaluation of the department by external accrediting bodies.

The overall study will take place in five distinct phases summarized below:

1. Raw Data Collection - This is the process described in this document.

2. Establishing Performance Metrics - This process will involve members of the faculty developing a set of measures that can be mathematically defined. This step may require a revisit of Phase 1 to get more raw data.

3. Model Data via Established Metrics - With the Phases 1 and 2 complete, the data must be transformed into a more mature working form compatible with desired modeling techniques. Great care must be exercised to assure that the models accurately represent the metrics outlined in Phase 2.


4. Dissemination - With all of the data processing complete, the new population representation must be disseminated to those with the ability to influence the correction of identified problems.

5. Maintenance - The model must be flexible enough to support easy update of the core system, easy extension of the of the system, and simple data acquisition of recent data changes.\(^3\)

As stated earlier, the first step in dealing with such a problem is obtaining the raw data. This can be a trivial task when one is granted some type of programmatic access to the database containing the required data. However, suppose that data is not in such a tangible form. This intangibility could be the result of extreme security concerns, a lack of programmatic connectors to the relational database management system, or a 1970’s era terminal interface again without programmatic access. In the case of the CSC study, the former two items are the primary barriers.

The data collection utility is a collection of concise python scripts designed to perform very specific tasks united via a web front end. No one utility is the “star of the show.” It is the combined utilization of these scripts that make system powerful. Each of the individual scripts are written in a modular fashion as to allow the maximum amount of code reuse. This same modularity makes incorporation of select functions into the python CGI scripts simple.

**System Architecture**

The bulk of the data collection architecture is built upon open source projects and scripting languages. The back end database server is a proprietary Microsoft SQL Server 2005/Windows Server 2003 system. The script and web hosting system is a Mac OS X 10.4.6 system running Apache 1.3.33 and Python 2.4.3. The systems are arranged in a classical 3-tier architecture model (see Figure 1).

The following open source projects/packages were used in the development of the system:

1. **pexpect-2.1beta** ([http://pexpect.sourceforge.net](http://pexpect.sourceforge.net)) - This is a pythonic rewrite of the Expect tool which claims to be 90% compliant with its Tcl namesake. The tool’s primary purpose is the automation of terminal based applications such as ftp, telnet, ssh, and sftp.

---

\(^3\) McHenry, C.E., interview on March 7, 2006, TTU Main Campus.
2. freetds-0.63 ([http://www.freetds.org](http://www.freetds.org)) [in support of pymssql] - FreeTDS is a set of Unix/Linux libraries supporting native connections to Microsoft SQL/Sybase relational database servers.

3. pymssql-0.7.4 ([http://pymssql.sourceforge.net](http://pymssql.sourceforge.net)) - This python modules provides native access to Microsoft SQL and Sybase relational database servers via the FreeTDS libraries. This provides the most efficient access possible from an open source client.

4. PyXML-0.8.4 ([http://pyxml.sourceforge.net](http://pyxml.sourceforge.net)) [in support of Gnosis_Utils] - This is a basic XML processing library for the Python scripting language.

5. Gnosis_Utils-1.2.1 ([http://www.gnosis.cx](http://www.gnosis.cx)) - This set of libraries and python modules provided an enhanced set of XML processors built upon the base PyXML libraries.

6. FCKEditor-2.2 ([http://www.fckeditor.net](http://www.fckeditor.net)) - This set of HTML and JavaScript files create a web-based text editor.

While it may seem a bit odd that a project with such an open source emphasis would make use of a proprietary database server, this was necessary. The third phase of this study will emphasize the graphical modeling of the metrics established in the second phase. Much of this modeling can be performed much easier, for most, in Microsoft Excel. To expedite the modeling process, the decision was made use Microsoft SQL Server 2005 making use of the inherent, native drivers between the modeling tool and the data storage tool. Figure 2 shows a modified Microsoft ER representation of the SQL 2005 database.

The following are the major scripts in the data collection utility:
1. createtables.py - This script drops and creates the raw data table structures on Microsoft SQL Server 2005.

2. dataeditor.py - This script provides an in-browser editor for the text control files.

3. dataeditorpost.py - This is the CGI form handler for the editor.

4. datatest.py - This script provides a sanity check on the number of rows in each tables.

5. form_action.py - This script is the primary CGI script responsible for handling all requests from the control panel.

6. getdata.py - This script is responsible for retrieving the raw text files from the administrative server.

7. index.py - This is the main driver for the utility.

8. mathdata.py - This is the primary screen scraping utility for the RPTMATHINFO report.

9. parse.py - This script parses the text files retrieved from the SIS system and inserts the appropriate data into the SQL Server.

10. samplereport.py - This is a sample report based upon the current data in the database. This section would ideally become a group of reports driven by a form-based menu system.

**Contributions**

While probably more interesting to me than anyone else, my project has a uniqueness in 3 areas:

1. My application is one of the first non-commercial screen-scraping applications used to work with data on the administrative systems.

2. All work performed by my application transmits data over secure, encrypted connections only, from web server request to file transfer. It is currently house on an isolated system.

3. The application is the beginning of a larger framework that will allow for a new level of student data access for the department.
Experiences

This class has added a new dimension to my programming skills. Until this course, the only web-based application that I had developed was a digital library application in C# using web services. I was a part of a two person group with the other individual being the lead programmer and with experience in web development using Microsoft web technologies. Now that I have had exposure to PHP and a refresher course in relational databases, I have been able to transfer those skills to other scripting languages. I also learned to hate JavaScript and its lack of portability across multiple browser platforms.

I used many of the techniques developed in the class in this Python application. I will also use some of the techniques in the testing phase of my thesis.