A Multimedia Database Project and the Evolution of the Database Course

Mark A. Holliday and James Z. Wang
holliday@email.wcu.edu, jzwang@cs.clemson.edu

Abstract - **The database course needs to continue evolving to reflect changes in the computing environment and changes in current thought about how best to engage the student. We propose one possible evolution that is based on a semester-long multimedia database project involving a web interface, called MeTube. The project supports uploading and downloading videos, images, and audio files as well as supplementary features such as ratings, comments, viewer blocking, and contact lists. Having a project that focuses on multimedia data and is web-accessible using current web technology will be more interesting, relevant, and useful for the students. Our description of the project includes both the requirements and the division of it into assignments. We show an example implementation and report on our experiences with the project. Finally, we explain how the project fits into the context of the course and future directions for the project.**

Index Terms - database management systems, multimedia databases, web interfaces, semester-long projects.

Introduction

In an effort to renew the curricula for the undergraduate degrees in computer science and computer engineering, faculty members from several universities are investigating a common approach. That approach is based on several principles [1, 2]:

- structure each course around a single large-scale problem. A semester-long project allows the student to work on a more realistic and substantial implementation of the course concepts.
- select those problems from the visual problem domain. Our students and society are increasingly visually oriented. These problems will more likely capture the attention and interest of the students.
- select problems that illustrate a connection between scientific and artistic perspectives. Reconnecting these two components of our students' intellectual development enhances their creativity and the quality of their future work.

In recognition of the third of the above principles, we have named our efforts, the τέχνη project. τέχνη (pronounced "technie") is the Greek word for art and shares its root with the Greek word for technology. Much of our group's efforts has addressed applying problems from computer graphics to courses early in the curriculum, in particular, CS1, CS2, and a data structures/algorithms course. These principles are equally applicable to upper-level courses in the computer science major or the computer engineering major. This paper reports on our experiences applying those principles to one such course, the database management systems course.

As remarked by Sciore [3], traditionally some of the topics in such a course address how to use a database system and others address how a database system works. A more recent approach splits the topics into two courses with the first course user-oriented and the second course systems-oriented. It is common for the user-oriented course to require a semester-long project developing a database application. The user-oriented database course already often follows the first of our three τέχνη principles, as a result.

Including a web interface in that project is also not new. Ramakrishnan and Rao [4] describe such a project that was used in 1996. The contributions of this paper are to discuss a new type of web-oriented semester long project for the database course, present our experiences with the project, and to reflect on the issues that arise with respect to the evolution of the database course. The application of last two of the three τέχνη principles motivated the design of this project.

In the next section we review the relevant issues in the evolution of the database course. Section Three describes the project we have used. Section Four covers our experiences with the project. In Section Five we reflect on the implications of our experiences for the future of the database course. We conclude in Section Six.

History

As noted above, soon after the widespread acceptance of the web, some instructors incorporated web interfaces into the database course project [4]. There has been persistent recognition of the importance of incorporating web interfaces into the course [5, 6, 7]. However, the relationship between the web and the database project has faced more complications than one might at first expect. There have been two main problems.

The first is that web application development has been undergoing continuous and major changes. For example, the Ramakrishnan and Rao project uses Perl-based CGI scripts. They emphasized the basics of achieving database connectivity. Many of the subsequent database projects with web interfaces that have been reported in the literature use

Java, JDBC (Java Database Connectivity), Servlets, and Java Server Pages [7, 8]. These comprise an important web technology and are a natural choice because Java is widely used in CS1 and CS2. However, there are a number of other web technologies that are at least as important: PHP [9] and newer web-development frameworks such as Ruby on Rails [10] and the Python-based Django [11] are examples. As a result, including web interfaces in a first database course presents a challenge in that part of the course needs to be updated frequently to reflect changes in the popular environment.

The second problem is the lack of space in the first database course. Covering the material on web interfaces results in the removal of other material. Two solutions have been used to address both of these problems: moving web interfaces into a second database course [8] or into a separate web application development course [12]. The disadvantage of the both of these solutions is that often there is not sufficient space in the curriculum for a second database course or a separate web application development course. Often the first database course is an elective as reflected in that only a few hours of database topics are part of the computer core in the 2001 Curricula recommendations [13].

In our opinion, it is important that there be a semester-long project in the first database course, that the project has a web interface, and that it adopt a currently, widely-used web technology. As the discussion above indicates, due to the resulting time constraints, we need to choose a web technology that can be learned quickly and we need to reduce the coverage of some other database topics.

The nature of the database application chosen for the course project is key and ties directly to the τέχνη principles. Typically, the instructor chooses a business application for the project in the database course; this is not necessarily the best choice. Students are likely to be more interested in problems selected from the visual problem domain. Supporting a visual problem seems natural because the project should have a web interface. Furthermore, a visual database project lends itself to showing a connection to the artistic mindset.

This reasoning led us to adopt a multimedia database project loosely motivated by the YouTube website; hence the name MeTube. One important difference is that MeTube stores audio files and images as well as video files. The original MeTube project was developed by one of the authors at Clemson University. In the next section we describe the revised project requirements used when the project was assigned during the Spring 2008 semester in the database course at Western Carolina University along with the experiences from that offering of the project.

A Multimedia Database Project

We organized the project as a series of parts spread over the entire semester as shown in Table I. Each part included one or more deliverables and built on the previous part. The deliverables were working code whenever possible to encourage students to start using MySQL and PHP early in the semester. For example, the relational schema is delivered as a script of SQL statements for creating the tables.

Table I: Project Parts and Deadlines.

Part	Deadline	Deliverables
E-R Diagram	End of 5th week	Graphic of the diagram
Relational Schema and Database Tables	End of 7th week	A working file of SQL statements to create and delete the tables
Functional Requirements and Corresponding SQL Queries	End of 9th week	A requirements specification document. A working file of SQL statements to create and populate the tables. A working file of the SQL queries.
System Implementation	start of 14th week	Working website including all php scripts. A test plan document.
Presentation	14th week	Powerpoint and live demonstration of website
Final Report	Day of the Final Exam	Previous documents plus a user manual and a copy of the powerpoint slides

We did not accept late submissions for any of the deadlines, to encourage students to not fall behind. We allowed teams of one, two, or three students.

We chose to make the requirements relatively specific for this project [14]. We needed to decide which of the project's features would be required and which would earn extra credit. We chose to require only the minimal functionality for this type of website and to award extra credit for all other features. A major reason for both of these decisions was our perception (which was subsequently confirmed) that even a website with minimal functionality and explicit specifications would still prove to be a challenging project given the time constraints and the need to learn the web development environment which includes PHP.

We based the required functionality for the project on two types of data: media files and registered users. These are shown in Table II.

Table II: The two main types of data and their features: media files and registered users.

Type of Data	Description
Media File	
File Properties	Media type (video, audio, and image) Upload time Popularity
Meta Data	Title, description, whether ratings are allowed, whether comments are allowed, whether this media file is globally shared
Multiple Instance Characteristics	Keywords, categories, comments, ratings

Registered User	
Account Properties	User name, email address, password
Lists of Other Users	A contact list of other registered users. A list of other registered users that are blocked from downloading files uploaded by this registered user.

Unregistered users are not specified as a data item, but they are supported because all users are able to:

- download, that is, view, a media file;
- search the MeTube system based on media file properties, meta data, and keywords;
- browse the media files by category, upload time, and popularity.

A number of possible extra credit features were identified which included users being able to:

- organize media files they viewed into play lists;
- invite friends to view/download media files through messages;
- organize the media files they uploaded into channels;
- subscribe to any channel created by another user;
- view and manage the channels the user created as well as channels the user subscribed to.

Figure 1 shows the Entity-Relationship diagram one of the teams developed to model this data using the Dia diagramming program [15]. It includes entity sets for the two main types of data (user and media file), but this team identified a number of other entity sets and a sizeable number of relationship sets.

That an edge is boldface indicates total participation of that entity set in that relationship set. For example, every media file participates in the Cat relationship set since every media file must belong to at least one category.

If there is an arrow on the edge from an entity set (for example, media file) to the relationship set (for example, uploads), then only one entity in the other entity set can be related to this entity (for example, only one user can upload this media file). In other words, an arrow indicates a key constraint.

That the keyword entity set is in boldface indicates that it is a weak entity set. This is an error in this team's diagram since it implies that the attributed named keyword does not uniquely identify a particular entity of the Keywords entity set.

The contacts relationship set in the top left of the diagram is noteworthy as an example of a unary relationship set that involves only the users entity set. One user is the owner of the contact list and the other users are on the contact list.

As a web environment we used the Apache web server, PHP for server-side scripting, and the MySQL relational database management system. This is a widely used environment and has a relatively small learning curve. We provided the students with accounts on our department Linux server that has the Apache web server and MySQL server that they used for their projects.

The students had not used PHP before, but it is a small language. We spent two fifty-minute class periods teaching the aspects of PHP that the students needed for the project. These aspects included example code fragments showing how to connect to a MySQL server and how to send a SQL statement to that server.

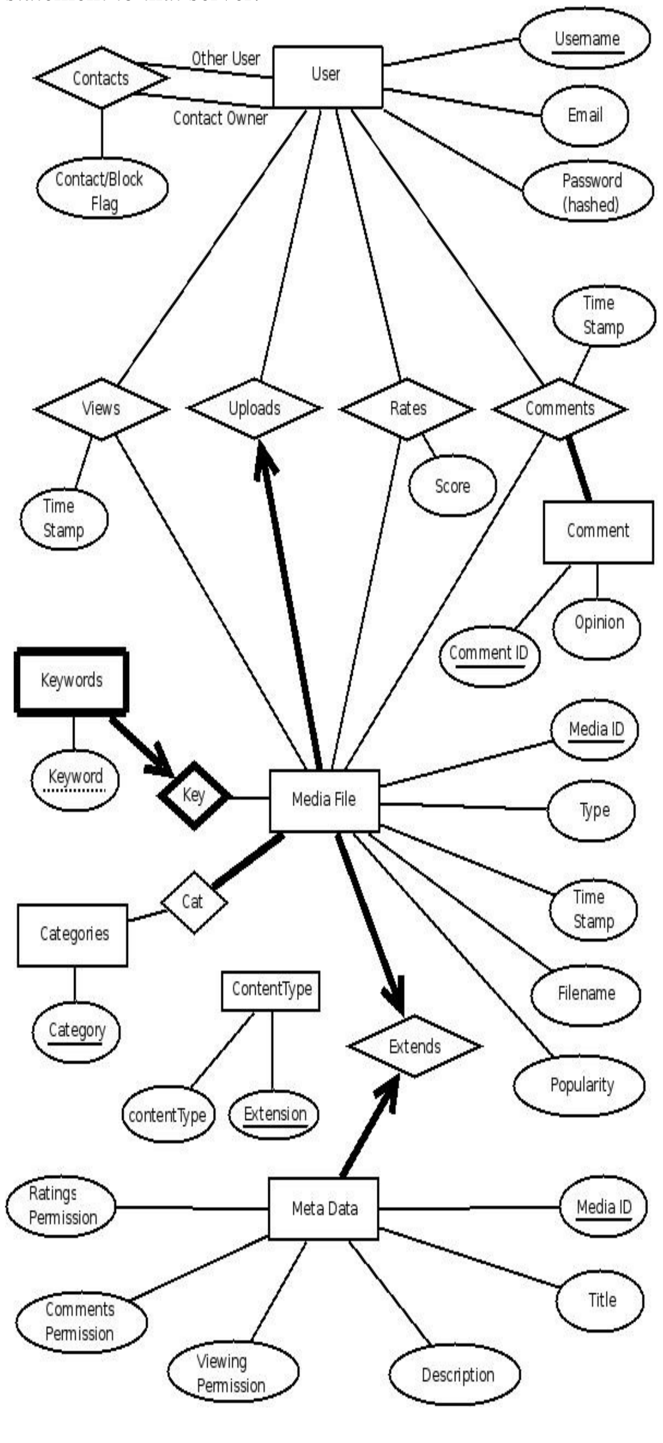
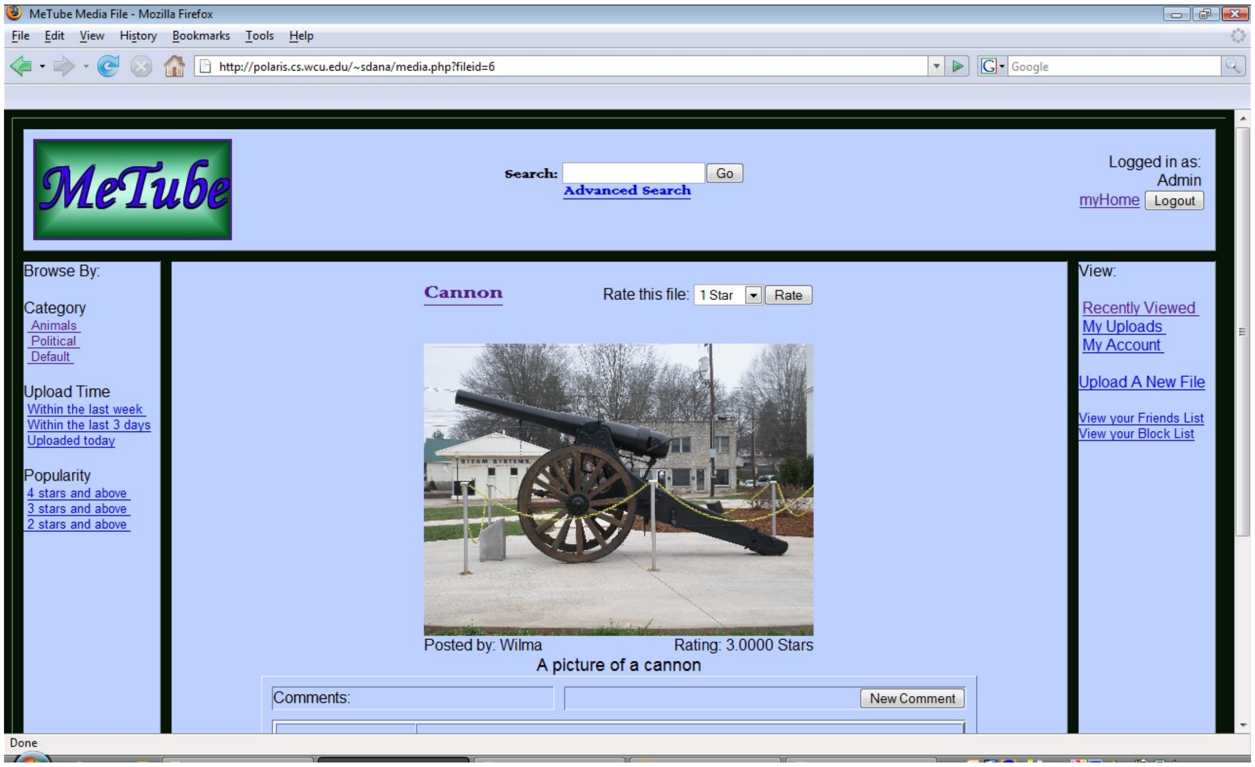


FIGURE 1: THE ENTITY-RELATIONSHIP (ER) DIAGRAM OF ONE TEAM.



http://polaris.cs.wcu.edu/~sdana/media.php?fileid=6

Logged in as: Admin

Cannon

Posted by: Wilma Rating: 3.0000 Stars

A picture of a cannon

FIGURE 2: THE INITIAL PAGE AFTER LOGGING IN.

To illustrate the typical MeTube website we have included one figure which is a screenshot of one of the pages of the website for one of the teams. Figure 2 shows the initial page of the MeTube website that appears after a user logs in.

- The top pane shows a search box in the center and to the right indicates the user who logged in and the means to logout.
- The left pane shows links supporting browsing by category (animals, political, and default), by upload time (all media files uploaded today, in the last three days, or in the last week), or by popularity.
- The middle pane contains the media file itself in the center with the media file title and a rating mechanism above it. Immediately below the media file the name of the owner of the file appears along with the file description, and the average rating. Only partially shown and further below are a series of text boxes, one for adding a comment and three for showing recent comments.
- The right pane shows links to pages showing the media files this user has recently viewed or has uploaded. It also links to pages for updating the user's friends list and for updating the list of users to be blocked. Finally, it has links to the account maintenance page and the page to upload a file.

A MeTube website contains many other pages. The initial page of the website is for a user who has not yet logged in. It is similar to Figure 1 in appearance but with less functionality. In particular, the right pane is empty and the right part of the top pane shows fields for logging in and for registering as a user.

A browser page for any user (logged in or not) lists all the media files by a certain trait (category, upload time, or popularity). For each file its title, average rating, description, list of keywords, list of categories, and an icon indicating its meda type are all displayed.

The page for uploading a media file is only accessible by a registered user after logging in. It has the same top pane, left pane, and right pane as in the page shown in Figure 2. The only difference is that the center pane contains fields for specifying all the information needed before uploading a file. For example, the file sharing level (everyone, only registered users, only friends) specifies who can view the file.

EXPERIENCES

Eight students completed the course (nine started); they chose to form four teams of two each. Although some of the students had done some web development before, none had used PHP significantly or MySQL at all. All the teams completed all of the required functionality for MeTube and prepared well-designed versions of all the other deliverables

(ER diagram, relational schema, requirements specification, SQL create table and query statements, test plan, user manual, and powerpoint slides). Several of the teams completed a small number of the extra credit features. We seemed to have set the size of the project at the right level.

The semester-long project clearly satisfied the three τέχνη principles identified earlier. By implementing a multimedia database with a web interface we engaged the students interest through a visual problem domain. Because audio files were stored along with images and videos, additional senses were involved. The media files themselves made a connection to art. The user makes an aesthetic choice each time the user selects a file to upload or view. Furthermore, the web interface implies that a significant part of the project involves designing an appearance to the web page that is appealing.

The success of the project is due to the overall design of the course. As mentioned earlier, the content of the typical computer science database course has changed over time with a split between user-oriented and system-oriented courses. The challenge of finding time in the course for covering web interfaces as well as other newer topics remains. Our course design is of interest because we needed to include time for covering web interfaces in order for the students to complete the project. We also felt strongly that the first database course should include some coverage of system issues, that is, how a database system works. To satisfy these constraints in a semester with fourteen weeks of classes required careful tradeoffs.

Choosing PHP (and Apache and MySQL) for the database environment allowed us to spend only a few fifty minute class periods on the web interface. Our textbook [16] contains a thorough coverage of both user-oriented and system-oriented topics. We first addressed the most important user-oriented topics (ER model, the relational model, logical database design and translation of ER to relational, SQL, the web interface, and relational algebra). We then addressed some of the key systems topics (file organization and indexing, B+-trees, hash tables, external merge sort, and query evaluation including implementation of the relational operators). Addressing the user-oriented topics first helped the students start on the project as early as possible. We addressed relational algebra after all the other user-oriented topics so that SQL and the web interface were not delayed.

Based on the students' comments and questions and their performance on the tests, quizzes, and the project, our approach to the course design and project were successful. Anecdotally, the students appeared excited about the course and the project; they found it both interesting and clearly saw how it would be useful to learn. In fact, almost all the students immediately followed the course with summer coops and other jobs in database development using their new skills.

We also used a survey of student perceptions of the classroom environment. Walker and Fraser [17] developed this survey originally for distance education but it is applicable to a face-to-face classroom environment such as our course offering. They have shown that there is strong correlation between traditional student outcomes and perception of classroom environment. The survey has 34 questions divided into six areas: instructor support, student interaction and collaboration, personal relevance, authentic learning, active learning, and student autonomy. Each rating is on a five category scale (always, often sometimes, seldom, and never) which we report on a 1-5 scale with 5 being the highest. For each student we averaged the ratings within each area. We then report the average of those averages over all the students. The survey was taken by all of the students with one week of classes left in the semester. Table III shows the results.

TABLE III: A SUMMARY OF THE DELES SURVEY RESULTS.

Area	Average Score
Instructor Support	4.53
Student Interaction and Collaboration	4.25
Personal Relevance	3.81
Authentic Learning	4.2
Active Learning	4.375
Student Autonomy	4.075

With only eight students it is not possible for us to perform a statistical analysis of the results. However, the results appear positive. In five of the six areas the average rating was between the rating of Always (value 5) and Often (value 4). The one surprising exception was the area of Personal Relevance where the average rating was between Often (value 4) and Sometimes (3). In anecdotal discussions after the survey was completed, a number of students criticized the statements in the Personal Relevance area such as "In this class I learn things about the world outside of the university." and "I apply my out-of-class experience."

FUTURE DIRECTIONS

Based on the experiences we have described involving the MeTube project and the database course in the Spring 2008 offering, we made some changes for the Spring 2009 offering. In terms of course content, we returned relational algebra to its traditional place before the introduction of SQL. In our experience having already learned how to construct queries using relational algebra expressions is helpful when the student learns the SQL Select statement. This does delay the introduction of SQL but not enough to cause problems in completing the project for the conscientious student.

The second change is in selecting the software used for the MeTube Project. This change is not due to dissatisfaction with the original software, but simply to experiment with an alternative. Instead of the MySQL database system, we are using Apache Derby [18], an open source relational database system implemented in Java that supports almost all the SQL-2003 Mandatory features. It is also small (only two megabytes for the base engine and the embedded JDBC driver). It can be embedded in any Java-based solution or used in client/server mode as a network server. We are using the JavaDB distribution [19] of Apache Derby.

We replaced PHP with Java and the Java DataBase Connectivity (JDBC) API for the database application environment. If the user interface of the Java program is a web page, then the Java program needs to be a series of Java Servlets or Java Server Pages. However, what is important is that the user interface be graphical and be easily reached from the web. Consequently, we are using the Java Swing API [20]. In our opinion, Java Swing creates a more attractive user interface and is a better programming environment, then HTML and Javascript.

The resulting Java program can be connected to the web using Java as an applet. However, we have chosen an alternative which is to use the Java Web Start framework [21]. This allows a Java application to be started directly from a web browser.

Conclusions

The database course, that is, the undergraduate, junior/senior course for the computer science major, needs to evolve as the computing environment changes and as we better understand the most effective ways to enhance student learning. The pervasiveness of the web is an important aspect of the computing environment and the τέχνη principles form an effective method for enhancing student learning. Both of these forces lead to the adoption of a semester-long project that involves developing a multimedia database with a web interface. In this paper we described such a project and reported on our experiences. We have also described how that project can fit into a course that covers both user-oriented topics (including the web interface) and systems-oriented topics.

Acknowledgment

Thanks to Julia Mack and Sam Dana for allowing us to include a screenshot from their MeTube implementation. This work was supported by the National Science Foundation under grant CPATH EAE 0722313.

References

[1] Davis, T., Geist, R., Matzko, S. and Westall, J., "τέχνη: Trial Phase for the New Curriculum", *Proc. of the Thirty-Eighth SIGCSE Technical Symposium of Computer Science Education*, Covington, KY, USA, March 2007, pp. 415-419.

[2] Davis, T., Geist, R., Matzko, S. and Westall, J., "τέχνη: A First Step", *Proc. of the Thirty-Fifth SIGCSE Technical Symposium of Computer Science Education*, Norfolk, VA, USA, February 2004, pp.125-129.

[3] Sciore, E. "SimpleDB: DB Internals", *Proc. of the Thirty-Eighth SIGCSE Technical Symposium of Computer Science Education*, Covington, KY, USA, March 2007, pp. 561-565.

[4] Ramakrishnan, S., and Madhu Rao, B., "Classroom Projects on Database Connectivity and the Web", *Proc. of the Twenty-Eighth SIGCSE Technical Symposium of Computer Science Education*, San Jose, CA, USA, February 1997, pp.116-120.

[5] Springsteel, F., Robbert, M.A., and Ricardo, C.M, "The Next Decade of the Database Course: Three Decades Speak to the Next", *Proc. of the Thirty-First SIGCSE Techn ical Symposium of Computer Science Education*, Austin, TX, USA, March 2000, pp.41-45.

[6] Robbert, M.A., and Ricardo, C.M., "Trends in the Evolution of the Database Curriculum", *Proc. of the Eighth Annual Conference on Innovation and Technology in Computer Science Education (ITiCSE 2003)*, Thessaloniki, Greece, July 2003, pp.139-143.

[7] Dietrich, S.W., Urban, S.D., and Haag, S., Developing Advanced Courses for Undergraduates: A Case Study in Databases, *IEEE Trans. on Education*, pp. 138-144, vol. 51, no. 1, February 2008.

[8] Dietrich, S., Urban, S. and Kyriakides, I. "JDBC Demonstration Courseware Using Servlets and Java Server Pages", *Proc. of the Thirty-Third SIGCSE Technical Symposium of Computer Science Education*, St. Louis, MO, USA, February 2002, pp. 266-270.

[9] *PHP Manual*, http://www.php.net.

[10] Thomas, D. and Hansson, D.H., *Agile Web Development with Rails, Second Edition*, The Pragmatic Programmers, 2008.

[11] Django Web Framework, http://www.djangoproject.com.

[12] Yue, K-B., and Ding, W., "Design and Evolution of an Undergraduate Course on Web Application Development", *Proc. of the Ninth Annual Conference on Innovation and Technology in Computer Science Education (ITiCSE 2004)*, Leeds, United Kingdom, June 2004, pp.22-26.

[13] ACM/IEEE-CS, *Final Report of the Joint ACM/IEEE-CS Task Force on Computing Curricula 2001 for Computer Science*, 2001, http://acm.org/education/curric_vols/cc2001.pdf.

[14] Holliday, M., Course Project: CS 453 Handout, Spring 2008, Western Carolina University.

[15] Dia Diagramming Program, http://live.gnome.org/Dia.

[16] Ramakrishnan, R., and Gehrke, J., *Database Management Systems, Third Edition*, McGraw-Hill, 2003.

[17] Walker, S.L., and Fraser, B.J., "Development and Validation of an Instrument for Assessing Distance Education Learning Environments in Higher Education: The Distance Education Learning Environments Survey (DELES)", *Learning Environments Research*, 2005, no. 8, pp. 289-308, Springer.

[18] Apache Derby, http://db.apache.org/derby/.

[19] JavaDB Distribution, http://developers.sun.com/javadb/.

[20] Java Swing API, http://java.sun.com/j2se/1.5.0/docs/guide/swing/index.html

[21] Java Web Start framework, http://en.wikipedia.org/wiki/Java_Web_Start

Author Information

Mark A. Holliday, Professor, Department of Mathematics and Computer Science, Western Carolina University, Cullowhee, NC 28723, holliday@email.wcu.edu

James Z. Wang, Associate Professor, School of Computing

Clemson University, Clemson, SC 29634, jzwang@cs.clemson.edu