

Freeware-Based Development Projects As A Learning Framework For Upper-Division Database Course

S. Pham **R. Covington**
spham@csun.edu cov@csun.edu
Department of Computer Science
California State University Northridge
18111 Nordhoff St.
Northridge, CA 91330, USA

Abstract - *The authors describe their experience with a novel approach for teaching a one-semester senior-level database design course in the Computer Science major. This approach innovates by combining coverage of traditional topics – entity-relationship (ER) modeling, normal form analysis, and Structured Query Language (SQL) – with practical hands-on projects that require students to analyze, customize, and deploy web-based applications with database back ends. Project resources are freely available from the World-Wide Web, with little dependence on institutional resources. Final projects are published on the Web for critique by other students. The instructors have used the approach for the past 4+ years. The real-life problems that arise during project completion have proven effective motivators for students to master the course’s theoretical material. The final project becomes a part of the student’s project portfolio. Former students have used such portfolios as an important element of successful jobs interviews.*

Keywords: Database Theory, Database Applications, BBS, Web Portal, Client-Server, Three-Tier Architecture.

1 Background

1.1 Traditional Approach To Course

The typical approach to the senior-level database design course is heavy on theory and light on practice. Topics covered include data modeling with entity-relationship (ER) analysis, analysis of tables for constraint violations via normal forms, and data retrieval through Structured Query Language (SQL). The course may also include a literature search of current database research topics (query optimization, materialized views, etc.), depending on the interests of both the students and instructor. With so many topics placing pressure on the available course time, students may resist the added

burden of implementation-oriented projects. Lack of institutional resources (hardware, software, and technical support) is also a common problem. As a result, students receive little experience with hands-on issues -- database administration, or deployment of live systems -- from typical offerings of this course.

1.2 Enhanced Hands-On Project Approach

The authors feel that giving students hands-on experience with real database deployment is essential to the course learning objectives. In response to the perceived need for a course that gives students this experience while still thoroughly covering the required theory, we have used the approach described here to present a senior-level database design course for the past 9 semesters. A lab manual for the projects has been published as one section of a textbook customized by the publisher for the course [1]. The approach has the following key features:

- It requires students to design and implement a database project which complements the theoretical course content.
- It uses existing applications with a web front end and database back end, for which students design and develop customized add-on “modules”.
- It uses resources and applications that are available for free or at a nominal cost from the Web, thereby avoiding institutional resource constraints. A primary innovation of the approach described here is to take advantage of these freely available resources and suggest how to organize them into a platform or framework for project implementation.
- It culminates in a project that is a concrete deliverable, one that the student can make profitable use of well beyond the end of the course.

There are a wealth of freeware applications on the web that are based on the client-server or 3-tier (web client,

web server, and database server) application architecture. Application genres include forums and web portals, which are described further in the next section. A guideline of particular importance for this course is to choose applications which include a back-end relational database for asset management. Coupled with free or nominal cost web hosting services, such an application suite can be assembled “on the cheap” and fashioned into an effective hands-on learning platform for the course with a modest amount of installation and setup.

Students may be wary of a course that requires mastery of both difficult theoretical concepts and challenging project implementations. The authors’ experience, however, has been positive, and shows that the implementation projects can be chosen so that they capture student interest. Once engaged in the projects, students show a noticeable increase in their interest in the theory.

2 Technical Background on Framework Components

The hands-on portion of the course consists of a series of projects that require students to work individually and in teams to customize a web-based application with a database back end and publish it on the web. The characteristics of an appropriate application include

- Application must include database for asset management.
- Application must include a web-based front end for access to the final product over a web browser.
- Application must publish an API in some suitable scripting language to allow students to add their customizations to the existing application. It is highly desirable for the application source code to be available to support detailed analysis.
- Application must be perceived by the students as current and relevant, to provide motivation to persevere through the project.

The authors have explored several application genres that have proven to be a good fit with these requirements. Applications such as BBS (bulletin board systems) or online forums are very popular with students. The more recent offerings of the course have begun to explore web portal applications, which are similar to BBS applications, but broader in scope. Below is a short description of each required element in the project framework:

2.1 Web Hosting

Student projects are required to be accessible from a web page on the Web, with a URL (uniform resource locator) that is shared with the instructor and the other students or teams. This makes it easy for the rest of the class to explore and critique the work from any web

browser. There are many free or nominal-cost providers of web hosting services [2],[3]. The customer opens an account, is given a user id and password to administer the site, a URL for customers to access the site from a web client, allocation of space on a web server, and tools (ftp=file transfer protocol or ssh=secure shell) to upload web pages to the server.

Another option, given that the average desktop PC purchased on a student budget is becoming quite powerful, is for a student to install a freeware web server, such as the Apache web server, on their own PC, and become their own web host service. The ambitious student can build a complete 3-tier application on a single PC. This approach poses more serious technical challenges (additional cost for fixed IP address, security and bandwidth concerns, etc.), and is not the best route for all students.

2.2 Web Portal Application

A number of freeware applications called web portals are available. Web portals provide complete web site deployment and management functions, so that a web site administrator can generate the pages of a site, link them together, and keep them up-to-date while writing little or no HTML (hypertext markup language, the low-level implementation language of web pages). [4],[5]

2.3 Bulletin Board Systems Or Forums

Bulletin Board Systems (BBS or Forums) are popular web-based applications that are fairly mature. The architecture of the applications is well suited to the pedagogical needs of this course. Application assets – forums, threads, messages, user identities, personalized welcome pages, etc. – are all maintained in a backend database [6],[7].

2.4 Database Server

Many of the applications from the categories described above offer downloads with a RDBMS (relational database management system) already bundled. Popular choices are freeware applications such as MySQL (dev.mysql.com) and PostgreSQL (postgresql.org). The user also has the choice of unbundling the database to connect the application with an existing database. Some institutions provide their students with site licenses to commercial database software, but if not, the student can use one of the freeware options. Some commercial database vendors give students limited access to free server software (see for example information on Oracle at www.oracle.com/technology).

2.5 Scripting Languages

Skill with scripting languages (PERL, JavaScript, Python) is required for deployment of any multi-tiered application, and a successful project should help students develop this skill. Scripts are used both to build the front-end GUI and to communicate with the back-end database. Older applications, such as Snitz Forum, are compatible with the Microsoft VisualBasic environment and offer an ASP-based (Active Server Pages) API. Newer applications, such as the PHP-Nuke Web Portal, are compatible with the C/C++/Linux environment and offer a PHP API. Another popular scripting language is JSP (Java Server Pages).

3 Course Structure

Presentation of the course starts with the traditional coverage of database theory -- modeling, constraints and normal forms, SQL and query optimization. In parallel, students begin work on the hands-on projects. Key points of all the project assignments are:

- Projects are a mixture of individual and team effort. For team projects, care is taken to ensure that every student contributes appropriate effort to the team. Students perform sub-projects individually early in the semester. Teams use the sub-projects as the basis for final projects.
- Projects require high-quality documentation. Students should have already developed good technical writing skills before this course. Project documentation must be accurate, well-organized, informative, to the point, and easy to understand. It is evaluated not only by the instructors, but by members of the other teams, i.e., a jury of one's peers, whose judgment is often more critical and unforgiving than that of the instructors. By the end of the course, students will on average produce 700-800 lines of script-based coding, plus 50-80 pages of documentation.
- One reason that documentation is emphasized so strongly is that well-documented projects can be fed back into future offerings of the course. Specifically, particularly interesting projects produced during one semester are archived to be used as "warm-up" assignments for the next semester. This technique generates a constant flow of new ideas into the pool of suggested assignments, keeping them up-to-date. It also ensures continuous improvement of the quality of the course and its projects.

Early in the course, students individually assemble the applications and tools from off the web and familiarize themselves with how to install and use them. This framework of basic tools and code acts as a baseline application that students will enhance during the rest of the

course. Through the use of team-based assignments, students also develop an understanding of software engineering concepts: teamwork, documentation, and peer review. To maintain individual accountability and to provide input to the teams, individual students in the initial stages of the project are given design assignments that they must complete on their own.

3.1 Project Organization

Students complete three sub-projects over a 15-week semester. The first sub-project is completed by each student individually. During the first project, the student becomes familiar with the tools and applications, assembling them from sources on the web and installing them. It ends with a short simple customization project with many coding details provided by the instructors in the form of "hints" during lecture.

The second sub-project is an individual design and implementation exercise at a more complex level. Topics are drawn from previous semester projects.

For the third project, students form teams of 4 or 5 students each. Ideally, the teams should be organized around students with the right mix of skills. We want to avoid creating one team of 4 or 5 expert coders and one team of 4 or 5 take-charge managers. Individuals bring their work completed during their first project to their teams. Among themselves, the teams evaluate individual strengths, and devise a compromise project that they commit to implement. The final phase of this project also gives the teams an opportunity to "close the loop". That is, each team acts as the implementer and evaluator of another team's work performed during the second project. To successfully implement a project designed and already implemented by another team, the new team must rely only on the other team's official documentation, which includes a user's manual and detailed implementation instructions.

The semester finishes with each team making a formal presentation of their projects. The presentation must include a live demonstration of the project running on the web host and accessed via a standard web browser. Beyond the end of the course, students use their projects as references during the job interview process, and the instructors pick particularly good projects for archiving and incorporating into the next semester's version of the course, where the project will be presented to a new group of student teams as one of their options of projects to implement.

In more detail, the project phases are (see Figure 1):

- Administrative Preliminaries: individual students obtain web host and database accounts and install the

applications on the web host. Applications include downloaded BBS software, or BBS plus customized add-on modules, possibly the result of an implementation effort from a previous semester.

- Application Enhancement: students learn how to use scripting languages to modify and develop their own add-on blocks or modules (Figure 1, Part A)
- Design Exercise: instructor assigns another customization task to individual students that will require appropriate analysis and modification of the backend database to solve.
- Solution Implementation: students form teams to reconcile designs produced by individual team members and produce a compromise design with elements from all the original designs that they will implement as a team. The final design reflects a contribution from each student’s personal area of expertise.
- Documentation: teams generate documentation, including a user’s manual containing installation instructions, plus implementation and analysis steps. This documentation is evaluated from the perspective of next semester’s students: could the documentation from this project be handed to students in the following semester and could they follow the instructions and successfully install this application? (Figure 1, part B)
- Testing of Other Team’s Projects: teams then take the customization projects generated by other teams, and using their documentation as a guide, integrate it into their own application, evaluating the other team for quality of documentation and ease of use of their product.
- Final Presentation: teams give a formal presentation of up to one hour on their project. The presentation focuses mostly on a live demonstration of their deployed application to the class via a web connection to their site. (Figure 1, Part C)

3.2 Feedback Between Project Phases

There is much opportunity for feedback between the phases of the project, as shown in Figure 1.

The larger circles correspond to the three projects. The smaller circles inside the larger ones indicate the skills or knowledge areas emphasized by that project. The boxes represent the database and documentation developed by the students. The lines with arrow indicate direct flow of work products. The dashed lines represent more general and indirect influences between tasks, deliverables, and projects.

3.3 Example Student Project Topics

To give a flavor of the kind of projects that recent student teams have implemented, here are three example project topics.

Dual Language BBS or Forum. This is a customized module added to an existing BBS to allow a forum member to post the same message in two languages (English and Spanish, or English

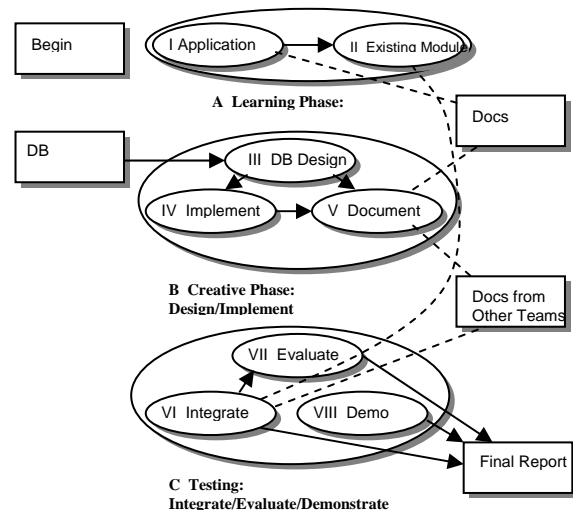


Figure 1: Project Phases and Feedback Paths

and French, or English and Japanese). When another member comes online to read messages, they can choose which language version to read. Note that the students do not implement any automated translation capability. They merely give the BBS the ability to tag messages with language information before archiving them.

BBS with Adaptive Spelling Error Detection and Correction: This module allows the forum user to create a “frequently misspelled words list” that the forum can use in future posts to offer automatic corrections or possibly just suggestions. It’s not a conventional spell checker in the sense of MS Word, but rather a filter tailored for the spelling habits of the individual forum user. The user could also use it to implement a series of keyboard shortcuts. The user enters a few characters, which the software mod recognizes as the trigger for a longer phrase, and then automatically completes the typing and enters it into the current message.

E-Book or Electronic Book or Smart Book Application: This project allows one or more authors to compose and publish a work online. Contents of the work can be continually updated in real time. The existing forum

capabilities can be used to allow other forum members who read the e-book online to post comments about specific passages both for themselves and the author. The author can provide a dynamic glossary of technical terms and hyperlink them to the text, so that on a mouseover event for one of the glossary terms in the main text, the glossary term is displayed as a tool tip, or temporary window with a definition that hovers over the term itself. The application defines roles for authors, readers, publisher (e-book repository administrator), and editor.

3.4 Example Database Customization Task

An example task assigned to students to learn how to customize the backend database that supports the BBS software (as described in Figure 1, part A) is to create a personal text edit area on the BBS GUI when the student logs in. The student must

- Study the database schema to learn the tables that hold data identifying registered BBS users
- Add an attribute to the appropriate table to hold the “personal area” text
- Study the ASP scripts that build the GUI to place the personal area text onto the GUI when the student logs in.

A partial schema for the FORUM_MEMBERS table is as follows (taken from the Snitz BBS):

TABLE: FORUM_MEMBERS

Field Name	Data Type
MEMBER_ID	AutoNumber
M_NAME	Text
M_USERNAME	Text
M_PASSWORD	Text
...	...
M_HOBBIES	Memo
...	...

The student customizes this table by adding an additional attribute to hold one personal text area per forum member:

M_PERSONAL	Memo
-------------------	-------------

The student then modifies the ASP code responsible for generating the GUI to include the new personal area.

```
<%
strSql = "SELECT M_PERSONAL FROM
FORUM_MEMBERS "
strSql = strSql & "WHERE M_NAME= ..."
set rs2 = my_Conn.Execute(strSql);
nPersonal = rs2("M_PERSONAL")
Response.Write(nPersonal)
%>
```

This example is an initial assignment given merely to help students acquaint themselves with the applications and tools, for example: 3-tier systems, SQL basics, ODBC

(Open Data Base Connectivity) queries and result sets, scripts to modify tables, and ASP (Active Server Pages) basics. Other customizations assigned later in the course require more complex modifications, including adding new tables to the existing database schema, composing queries that join the new tables to existing tables based on existing primary keys. For example, the previous example could be easily extended to allow each forum member to control multiple personal text areas (which requires a new table). Depending on the nature of the modification, students may be required to submit normal form analysis of the modified schema as evidence of a correct design.

3.5 Excerpt from Student’s Normal Form Analysis and Implementation Instructions

The following is a small excerpt from work submitted by a student team which implemented a customized add-on module for a BBS/forum application. The new module allows a forum user to conduct an online survey of other forum members. The example is very brief, but is meant to illustrate the level of detail required to implement the projects.

Table: Question

- Attributes: { Question_ID, Question_Desc, Question_Order, Question_TypeID, Question_Status, Survey_ID }
- Functional Dependencies
- FD = { Question_ID ->Question_Desc, Question_typeID; Question_ID, Survey_Id ->Question_order, Question_status; }

Comments

Not BCNF (Boyce-Codd Normal Form), not 3NF (3rd Normal Form). Table should be split into two:

- R1={Question_ID, Question_desc, question_typeId}
- R2={Question_ID, survey_id, question_order, Question_status}

Both resulting tables are in BCNF.

Implementation of Module: Instructions to Other Teams Wishing to Implement This Module

To implement the survey module, there are two instructions, depending on type of database server:

If Running SQL Server

- Copy our customized files “submitsurvey.asp” and “survey.asp” into the homepage directory (directory where “default.asp” is located)
- Create the tables required by our relation schema.
- Add stored procedures (see our “Stored Procedures Documentation” for definitions).

- Set up an ODBC connection.
- Insert the exact text below in “default.asp”, after line 410 or after “WriteFooter”
%><!--#INCLUDE FILE="survey.asp"--><%
- You must have a connection string for the ODBC connection created in Step 4. It is used in two places: line 26 and line 91 in “survey.asp”. This is the connection to your SQL Server.

If Running Other (Non SQL Server) Server

Use the following query:

```
Sql = "SELECT * FROM Snitz_UserSurvey
WHERE User_ID = "
"& request("SurveyMemberID") &"
AND Survey_ID = "& request("SurveyID")"
```

4 Evaluation And Assessment

Based on student feedback, the projects as described satisfy the primary goal of hands-on reinforcement of the abstract theory of database design. But they have also proven to be a good fit with an emerging trend in upper-division CS courses: the need for a “capstone” project (i.e., a project that requires a senior-level student to draw on skills developed across multiple courses to produce a significant piece of work that demonstrates mastery of the complete discipline). The projects have also shown value as a component of another trend for upper-division CS students: the need for students to maintain a portfolio of significant work accomplished across their student career. Assessment activities increasingly require degree-granting institutions seeking accreditation to verifiably demonstrate (not just claim) that students are achieving the institution’s advertised learning objectives, and the class portfolios of completed projects described here are a possible contribution to that goal. Class portfolios also benefit the student during interviews for jobs or internships. The portfolio is a concrete demonstration of level of accomplishment in the discipline. It includes a URL that links to a live website where the project is hosted. Some students prefer to include an executable version of the project on an attached CD/DVD and make it part of the portfolio submission.

5 Conclusion

Motivating students to master theoretical material such as database design is a challenge. Many students in computer science are working in the industry part or full time, and have a no-nonsense attitude when evaluating the practical relevance of topics from a traditional curriculum. Finding project assignments that clearly demonstrate theoretical issues and solve a practical problem is a good approach for winning over these skeptics, but department

budgets and resources may place limits on scope of project.

By taking advantage of web hosting and freeware web-based applications that can be customized with add-on modules, no department resources are required. Students learn much about the software architecture of the applications that they study, how to develop new modules and integrate them with the existing ones, and how to propose features that go beyond those offered by the current applications. Applications are drawn from practical areas that students are naturally motivated to learn about. Students must use theory presented in class to solve their database design problems.

By working in teams, students are motivated by the need to contribute and carry their weight, to get recognition from their peers for good ideas, to develop leadership skills, to get exposure to solving problems in groups, and even to promote friendly rivalry between teams. The team approach also helps make size of the problem manageable for a one-semester course. The publishing of the finished project on the web motivates students to put forward their best work for public display. Beyond that, many students feel that these projects have turned them into contributing members of the community of web users and developers. For many of our students, success on the course projects has also led to big increases in their self-confidence and self-esteem.

6 References

- [1] Lewis, Philip M., Bernstein, A., Kifer, M., Riccardi, G., Pham, S., Covington, R., *Database Design: Custom Edition for the Department of Computer Science, California State University, Northridge*. Pearson Custom Publishing, 2003.
- [2] <http://www.brinkster.com>
- [3] <http://www.freeservers.com>
- [4] <http://www.postnuke.com>
- [5] <http://www.php-nuke.com>
- [6] <http://forum.snitz.com>
- [7] <http://www.phpbb.com>