https://doi.org/10.48009/1_iis_2006_51-55

A CAPSTONE PROJECT IN SOFTWARE DEVELOPMENT FOR CIS MAJORS

Dr. Michel Mitri, James Madison University, mitrimx@jmu.edu

ABSTRACT

This paper presents a capstone project given to CIS seniors utilizing programming, database design, requirements analysis, use case, and object modeling, project management, and writing and oral communication skills in an active learning environment. The learning objectives for this project corresponds with most of the learning units described for the upper level courses in the IS 2002 model curriculum and is based on an object-oriented methodology that centers on component-based design. Students also engage in reflective analysis by constructing project workbooks and journals.

Keywords: Capstone, Java, Systems Analysis, Database Design, Project Management, Use Case, ER Diagram, UML, Component Based Design

INTRODUCTION

One of the most difficult skills for CIS students to master is software development, especially in database-oriented, client-server environments. This mastery is essential for successful preparation into the IT field. It is especially important to provide students with realistic systems development experience that facilitates skills in project management, systems analysis, database design and usage, and computer programming, all the while fostering teamwork and developing oral and written communication skills.

In information systems curricula, a viable way to integrate skills into a comprehensive project is via a capstone course, typically offered in the final semester of a student's college career. In the IS 2002 model curriculum, the capstone course is called Project Management and Practice, and involves "… a high-performance team that will engage in and complete the design and implementation of a significant information system [1]." A good capstone project should involve creativity, develop enjoyment of problem solving, and encourage conscientiousness. Students should be taxed in their technical and cognitive skills as they each tackle their individual programming assignments, and they should develop maturity and the ability to work with others as part of their group projects.

OBJECT-ORIENTED COMPONENT-BASED SYSTEMS DEVELOPMENT

Modern-day software applications can be characterized as assemblages of portable software components. This has led to a new approach to software development, often called Component Based Development (CBD), which facilitates software reuse [3].

The capstone project described in this paper relies heavily on component development and assembly. Several major components are developed by the students in individual assignments *prior to* the capstone project. Table 1 lists the components developed as individual programming assignments, and describes the programming techniques and Java constructs used for these components.

DESCRIPTION OF CAPSTONE PROJECT

After the students have completed the individual assignments and gained the advanced programming skills, as well as having constructed some of the important components required for the final projects, they are divided into teams and given a description of their project.

The capstone project takes place as a series of four phases. Each phase involves a cycle that involves planning, analysis, design, and implementation. Thus, the traditional SDLC is applied iteratively throughout the phases of the capstone project, as described in the following sections of this paper [2].

Table 1. Description of components and applications developed by individual students during first half of capstone semester. These will be used in the group project during capstone project.

Components and Applications Developed during first half of Semester	**Purpose**	**Programming Techniques and Java Constructs** (see http://java.sun.com/j2se/1.4.2/docs/api/)
DBSource (component)	Provide a simple interface for an application to connect to and perform queries/updates on a database.	JDBC. Connection, ResultSet, Statement classes and interfaces.
Multithreaded Server (application)	Application that generates threads to service clients and communications with clients via sockets. Receives a SQL statement from the client and sends strings based on query results back to the client. Uses the DbSource class to interface to database.	ServerSocket and Socket classes. Thread class. DataStream class.
ClientConnector (component)	Component used by client applications to communication with the multithread server. Sends SQL statements to the server and places the results into two-dimensional String arrays	Socket and InetAddress classes, multi-dimensional arrays, use of StringTokenizer for breaking up string results.
Bar Chart Panel (GUI component)	Display a bar chart based to visually represent numerical data, along with descriptive labels for bars.	Graphics abstract class, Rectangle class, JPanel class, X-Y coordinate system, Color class, line, shape, and string drawing. Calculating bar heights and widths.

Each team is given a different project. An example project is described below:

Project Planning Services, Inc. is a business consulting firm that provides technical and strategic services to its clients. The company has a need to develop a database and an accompanying application system to capture, store, and manipulate data pertaining to the activities of its consultant employees. In particular, PPS, Inc. needs to know about who the consultants are, the projects that they are working on and the clients they are servicing. The company also wants to track the time that they spend on each task of each project that they work on. In addition, any expenses that consultants incur must also be tracked. Your task is to develop the information system to satisfy these needs.

The general architecture for the systems developed in the capstone projects is shown in Figure 1.

Each group creates a database and three separate application programs geared to fulfilling the needs described in their cases. The application programs communicate with the multithreaded server and use the client connector and bar chart components (described in Table 1). The italicized components in the figure are created by students as individual programming assignments prior to the capstone. The non-italicized portions are created by teams during the capstone project. Details about the development process are described in the remainder of this paper.

PROJECT SCHEDULE

The group project is divided into four overlapping phases: (1) database development (consisting of ER-modeling, relational database design, and data population); (2) systems analysis tasks (including requirements gathering, use case construction, and user interface design); (3) software construction (involving Java programming of applications as well as development of UML class diagrams); and (4) documentation and presentation (which involves demonstrating the project results in a public presentation and creating a users manual).

Phase 1: Database Development

In the first phase, student teams are presented with a verbal description of the business entities and their relationships. Based on this description, the team must develop a data model followed by a relational database design. The typical data model involves 7-8 entities in a variety of 1:N and M:N relationships Based on the ER model, a relational database is constructed, such as the one shown in Figure 2.

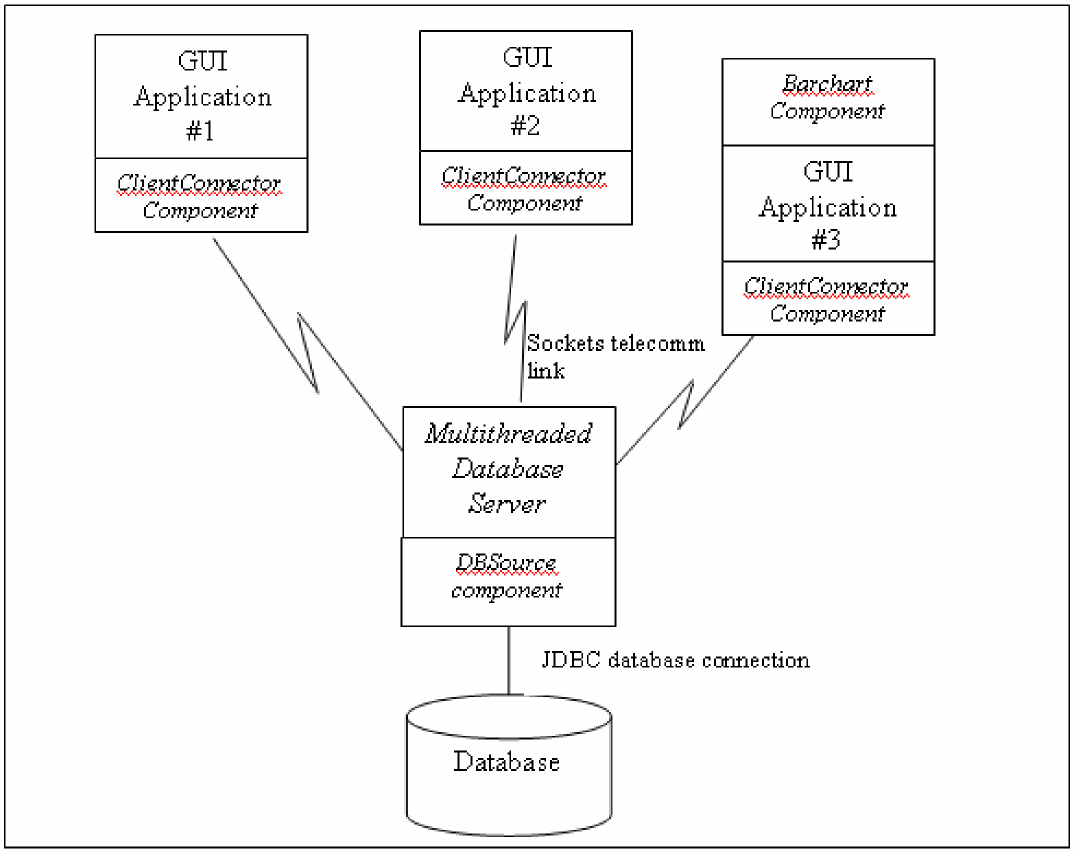


Figure 1. System architecture of capstone project application. Italicized components are developed as individual assignment during the first half of the semester.

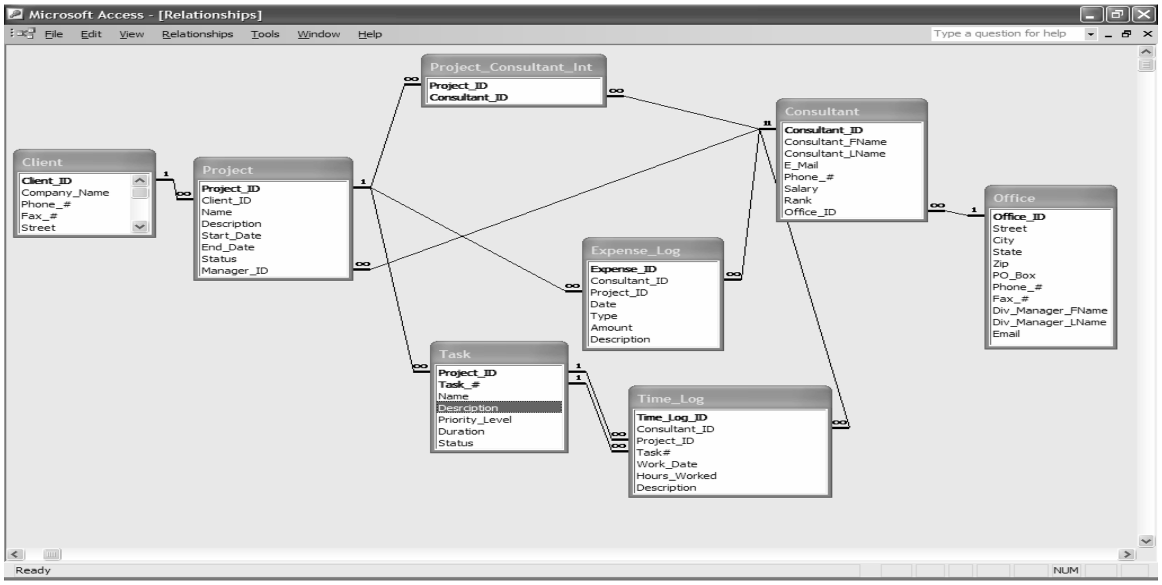


Figure 2. Database design constructed based on ER model.

Project teams are also given 4-5 text files containing data which must be inserted into the database. This data is non-normalized, often duplicated and/or involving mixed entities. Students must write a Java program to read the text files and programmatically generate the SQL statements necessary to populate the database. Their efforts writing this utility program give students realistic experience. Most systems development projects involving construction of new databases require some sort of data extraction and collection tasks from disparate data sources in order to produce the initial data in the new database. In addition, their programming efforts make use of a previously built component, specifically the DBSource component (see description in Table 1).

The final deliverable from Phase 1 includes (1) an ER diagram of the data model, (2) the fully implemented and populated database, and (3) the source code for the Java utility program. At the end of this phase, these items will be included in the project workbook for review, grading, and comment by the instructor.

Phase 2: Requirements Analysis

The second phase involves interviewing the user (a role played by the instructor of the course) in order to ascertain the functional requirements of the system. Unlike Phase 1, in which a fairly complete written description is provided to the students, Phase 2 involves no written document for students to work with. All information about the desires of the user must be obtained via interviews, so students make use of oral communication skills and interviewing techniques they learned in the systems analysis prerequisite course. This interview process is done in a series of meetings between the instructor and the team members over the period of one week. As a result of this process, the team must produce use cases for the applications that will be developed. Each project involves three separate application programs (see Figure 1), geared to three different types of user. Consequently, the use case diagram will display three actors, and will include 3-4 use cases for each application of the system. Each use case must be described in detail in a written use case document.

The final deliverables for Phase 2 include (1) the use case diagram, (2) written descriptions for each use case, and (3) images of the user interface designs. At the end of this phase, the project workbook is updated to include these deliverables, and is reviewed and graded by the instructor.

Phase 3: Software Construction

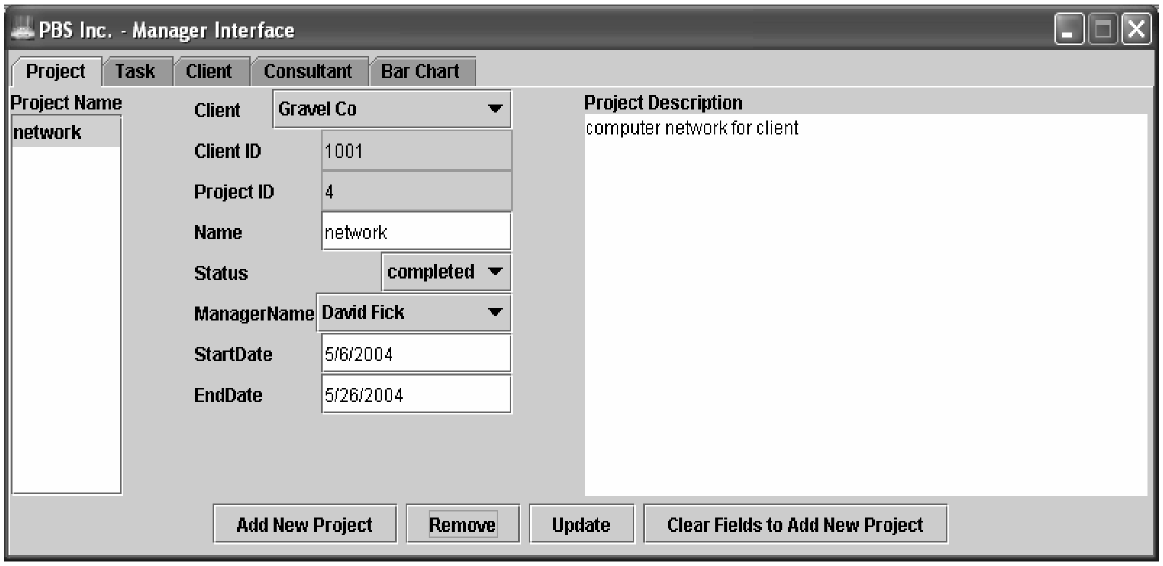
Phase 3 is the most difficult and time-consuming part of the whole project. This involves actual computer programming and debugging, which is a particular challenge to most students. Here, students write the code for the three application programs. All programs interface to the multithreaded server that students had developed earlier in the semester during their individual assignments; this communication is accomplished via the communication ClientComponent that was developed as an individual programming assignment (see Table 1). All three applications also involve an extensive array of GUI programming and event handling routines. In addition, at least one of the applications involves use of the bar chart component (Table 1) students had developed during their individual assignments. All this reinforces the concept of component-based design.

During this phase, students often run into difficulty with bugs in their programs. During previous individual assignments, they had learned debugging techniques, but this is the first time they are experiencing this in the context of a multi-application system with extensive complexity. Whereas testing in the individual assignments is limited to unit tests, testing in the group project extends to systems integration and user acceptance testing.

Figures 3 and 4 show screen shots of one of the applications written by students in the group project.

Phase 4: Documentation and Presentation

In this final phase of the project, students complete a user manual, the technical documentation, and a *PowerPoint* presentation that is presented during the final exam period. Compared to Phase 3, this is a much simpler task and typically is done during the last week of classes as the programming effort is coming to an end. Students are required to create a complete user manual, giving an overview of the system as a whole, describing each of the three applications in terms of purpose and functionality, and fully explaining all the menu options and other interactive controls for each of the applications; thus written communication skills are important in this phase. Screen shots are a required part of the user manual.


PBS Inc. - Manager Interface

Project | Task | Client | Consultant | Bar Chart

Project Name: network

Client: Gravel Co
Client ID: 1001
Project ID: 4
Name: network
Status: completed
ManagerName: David Fick
StartDate: 5/6/2004
EndDate: 5/26/2004

Project Description: computer network for client

Figure 3. One screen of an application built by a student team, utilizing several built-in Java components.

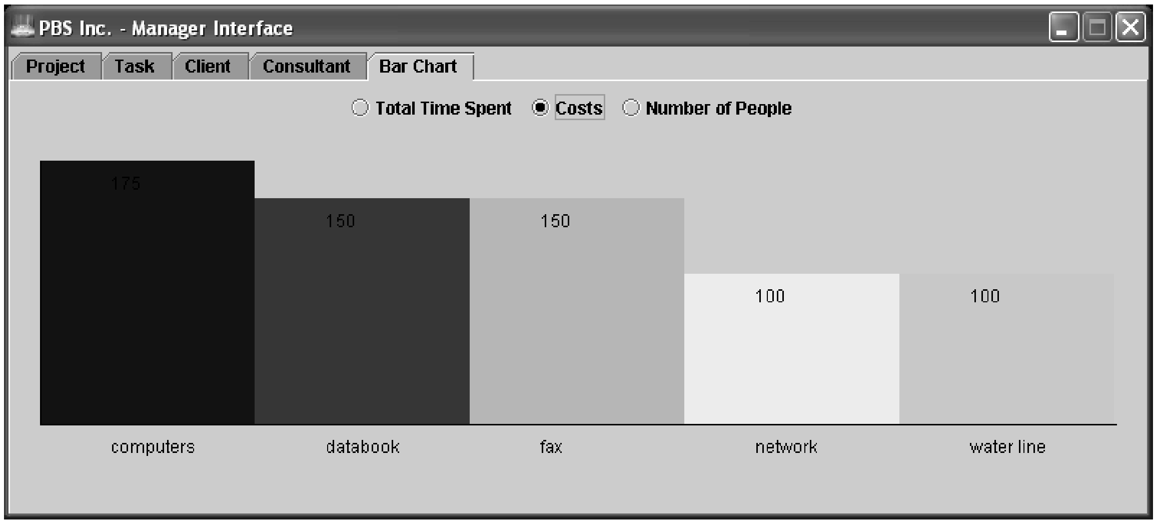


Figure 4. A second screen of an application built by a student team, this one involving the bar chart component developed during the first half of the semester.

REFERENCES

1. Gorgone J.T., Davis G.B., Valacich J.S., Topi H., Feinstein D.L., & Longenecker H.E. (2002). IS 2002 Model Curriculum and Guidelines for Undergraduate Degree Programs in Information Systems, www.is2002.org.
2. Hoffer, J., George, J.F., a7 Valacich, J. (2005). *Modern Systems Analysis and Design*, Upper Saddle River, NJ: Prentice Hall.
3. Ratchivadran, T. & Rothenberger, M. (2003). Software reuse strategies and component markets, *Communications of the ACM*, *46*(8), 109-114.