ABSTRACT

This paper presents a capstone course given to CIS seniors involving projects which utilize programming, database design, requirements analysis, use cases, object modeling, project management, and written and oral communication skills in an active learning environment. The learning objectives for this course correspond with most of the learning units described for the upper level courses in the IS 2002 model curriculum. The course content and capstone project are based on an object-oriented methodology that centers on component-based design. Project elements emphasize both creative and critical thinking processes and are designed to utilize and challenge the technical, managerial, and communication skills of students, who also engage in reflective analysis by constructing project workbooks and journals.

Keywords: Java programming, database design, systems analysis, project management, component-based design, IS 2002 learning units, journal writing.

INTRODUCTION

One of the most difficult skills for IT students to master is software development, especially in database-oriented, client-server environments. This is particularly true for CIS majors, whose academic background tends to be more business oriented and less mathematical and technical than computer science majors. Nevertheless, a mastery of or at least strong understanding of software development skills 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.

Although many information systems students do not intend to focus on computer programming during their IT careers, and many programming tasks are outsourced to foreign workers, programming in modern languages such as Java, VB.NET, C++, etc. remains among the top IT skills identified by employers in the major Internet job portals; calls for expertise in this skill are usually integrated with calls for Web development and database skills [9, 10]. This, however, is contrasted with empirical survey findings that employers place more emphasis on the people and business-related skills than on technical skills when hiring IS talent [3, 12]. One possible explanation of these conflicting findings is that technical skills may form the necessary minimum requirements for candidate qualifications and thereby are used as filtering criteria by recruiters, whereas organizational and communication skills are used to select from the shortened list of candidates when making final hiring decisions [12]. At any rate, it is evident that both “hard” and “soft” skills are necessary for IS professionals, and therefore development of both skill types should be facilitated for IS students during their college careers.

One criticism of academic IS programs cited by employers is the lack of integration of technical and personal skills attainment in the curriculum; different skills (database, networks, systems analysis, programming, project management, communication) are typically taught in separate courses, and students do not gain sufficient experience combining these skills in a single course or project [6]. Thus, in order to better meet the needs of employers and to make IS a more attractive major for students, it is important to provide a realistic experience covering all the skills involved in information systems.

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. Many IS programs include capstone courses and projects, and there are various descriptions of these in the academic IS literature [16, 19, 21]. 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. [7]” Although the emphasis is on project management and systems integration, skills related to programming, database design, and systems analysis are all prerequisites for, and are utilized in, this recommended course.

The capstone project described in this paper was designed to facilitate development of the skills that satisfy the major criteria for successful IT professionals. In particular, 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 should develop maturity and the ability to work with others as part of their group projects [20].

Prior to taking the capstone course, students will have learned all the fundamental programming, analysis, design, and database skills required for an integrated systems development project, having completed courses covering: Visual Basic, object-oriented programming with Java, systems analysis and design, and database design, as well as hardware, operating systems, and networking. During the capstone course, students advance their programming and database skills in individual assignments, and then apply these to a group capstone project. The skills developed and used in the capstone course are related to learning units ascribed to the following courses in the IS 2002 model curriculum [7]; IS 2002.5 — Programming, Data, File and Object Structures, IS 2002.7 — Analysis and Logical Design, IS 2002.8 — Physical Design and Implementation with DBMS, IS 2002.9 — Physical Design and Implementation in Emerging Environments, and IS 2002.10 — Project Management and Practice. Specific learning
units of these courses are mapped onto the phases of the capstone project, as will be described throughout this paper.

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 [17]. The CBD approach is gradually working its way into programming curricula [1, 8]. But it is quite challenging to educate IS majors on system component design in an object-oriented, and some argue that mixing object-oriented with more traditional structured methodologies confuses the students [18].

The capstone project described in this paper relies heavily on component development and assembly. Although some components are vendor supplied (such as Java objects and middleware), others are developed in individual assignments by the students themselves during the first half of the semester, as they learn advanced programming techniques pertaining to database processing via JDBC, multithreading, socket programming for telecommunications, and graphics programming.

Students spend the first half of the semester in individual assignments, during which they develop a multithreaded server that allows clients to submit database requests and sends the results of these requests back to the clients. In addition, students build components that (a) connect to and interact with a database, (b) communicate with the multithreaded server over the Internet, and (c) display a graphical representation of numerical data in the form of a bar chart. Thus, at the end of the first half of the semester, each student has written, as individual programming assignments, the major components required for the final applications of the capstone project. These components are written to satisfy a specified API, with the intent that they will be used in future applications (particularly during 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.

During the capstone project itself, an object-oriented systems analysis and design (OOSAD) approach [5] is utilized. George describes OOSAD as having four main phases: inception, elaboration, construction, and transition. During inception, developers define the scope of the project and its business case. In elaboration, the project needs are analyzed in detail. Application design and source code development is emphasized during the construction phase. And transition involves delivery of the system to users. The key difference between this approach and the traditional SDLC is that the cycle of planning, analysis, design, and implementation is repeated multiple times in each of the OOSAD phases, and the phases each involve actual construction of system components. This is referred to as an iterative, incremental development methodology, and relies heavily on rapid application development techniques. Although all SDLC stages iterate over each of the OOSAD phases, the degree of effort of the SDLC stages vary in different OOSAD phases. For example, analysis is emphasized more heavily in inception and elaboration, and drops off in emphasis during construction and transition. Design is at its heaviest during elaboration and construction, whereas implementation is at its heaviest during construction and transition. Management and planning, however, remains constant throughout the entire project [5 p66].

Similarly, 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

<table>
<thead>
<tr>
<th>Components and Applications developed during first half of semester</th>
<th>Purpose</th>
<th>Programming Techniques and Java Constructs</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBSource (component)</td>
<td>Provide a simple interface for an application to connect to and perform queries/updates on a database.</td>
<td>JDBC. Connection, ResultSet, Statement classes and interfaces.</td>
</tr>
<tr>
<td>Multithread Server (application)</td>
<td>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.</td>
<td>ServerSocket and Socket classes. Thread class. DataStream class.</td>
</tr>
<tr>
<td>ClientConnector (component)</td>
<td>Component used by client applications to communicate with the multithread server. Sends SQL statements to the server and places the results into two-dimensional String arrays.</td>
<td>Socket and InetAddress classes, multi-dimensional arrays, use of StringTokenizer for breaking up string results.</td>
</tr>
<tr>
<td>Bar Chart Panel (GUI component)</td>
<td>Display a bar chart based to visually represent numerical data, along with descriptive labels for bars.</td>
<td>Graphics abstract class, Rectangle class, JPanel class, X-Y coordinate system, Color class, line, shape, and string drawing. Calculating bar heights and widths.</td>
</tr>
</tbody>
</table>

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.
is applied iteratively throughout the phases of the capstone project, as described in the following sections of this paper.

DESCRIPTION OF CAPSTONE PROJECTS

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. Each team is given a different project; a total of four projects have been developed for this course. These are listed and described in Table 2.

When completed, the projects will result in integrated systems whose general architecture is depicted in Figure 1. Each group is required to create a database and three separate application programs geared to fulfilling the needs described in their cases. The application programs will communicate with the multithreaded server, and will use the client connector and bar chart components (described in Table 1). The italicized components in the figure are created by students during the first half of the semester as individual programming assignments. 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 Planning Services, Inc.

PPS, 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.

### BeastOfBurden.Com

BeastOfBurden.Com is a startup online service that matches companies with prospective job applicants. This company services two target audiences. First it services businesses who want to post job opportunities and search for and identify the job applicants who meet the skill sets required for their job opportunities. Second, it services job applicants, who want to search for job opportunities that match their skill sets. BeastOfBurden requires an information system to manage the services it provides by keeping track of the companies it services as well as the job applicants who use the system. Your task is to develop the information system to satisfy these needs.

### SysDoc Services, Inc.

SysDoc Services, Inc. is an IT documentation service provider that offers systems analysis and documentation for enterprise-wide business systems. The documentation services it provides keeps track of the various software and data components of an information system, the personnel involved in supporting these components, the problems identified by users when working with the system, and the actions taken by programmers to fix these problems. SysDoc Services is in need of an automated system to keep track of all these items. This problem tracking system will be used by programmer, system users, and project managers. Your task is to develop the information system to satisfy these needs.

### YouBuyIt.Com

YouBuyIt.Com is a startup online retail company that provides products to customers over the Internet; in other words, an e-commerce company providing Web-based sales and services. The company has a need to develop a database and an accompanying application system to capture, store, and manipulate data for its customer order processing operations. In particular, YouBuyIt requires a system that provides customers with online access to its catalog of products and enables them to purchase products online. In addition, the system must provide customer service representatives with the information required in order to fulfill the orders placed by customers. Your task is to develop the information system to satisfy these needs.

**Table 2:**
Figure 1: System architecture of capstone project application. Italicized components are developed as individual programming assignments during the first half of the semester.

Figure 2: Gantt chart and work breakdown structure showing the main tasks of the capstone project.
<table>
<thead>
<tr>
<th>Phase</th>
<th>Tasks</th>
<th>OOSAD Stages Emphasized</th>
<th>IS2002 Learning Units Involved [5]</th>
</tr>
</thead>
</table>
| Phase 1 Database Design | Construct ER model, Construct relational database, Populate database via utility program | inception, elaboration construction transition | 88. develop skills in data modeling  
92. in application and physical implementation of database systems  
95. design a conceptual data model and logical database model, convert the logical database designs to physical designs, develop the physical database  
53. operate on data and file structures  
54. to develop structures using abstract data types representing arrays, lists, trees, records, and files  
57. applications of data structures and file processing techniques  
58. problem solving involving files and database representations  
95. generate test data  |
| Phase 2 Requirements Analysis | Conduct interviews, interviews for Requirements Gathering Use Case Modeling User Interface Design | inception, elaboration | 52. development tools, and graphics development environments  
74. collect and structure information in the development of requirements and specifications  
96. develop functional specifications for an information system, develop a detailed information system design  
111. develop requirements and specifications for multi-user information system based on a database  |
| Phase 3 Software | Application programming and testing, UML Class diagramming | elaboration, construction | 46. systems view of object representations  
47. developing an algorithmic solution to a problem  
48. top-down implementation  
49. object implementation  
50. modular design, cohesion, and coupling  
51. verification and validation  
52. programming environments, development tools, and graphics development environments  
54. using abstract data types representing arrays, lists, trees  
58. problem solving involving . . . database representations  
60. . . testing, and debugging of IS related programs  
81. application skills for implementing databases and applications  
93. use of a combination of code generators and language facilities to implement multi-user departmental or simple enterprise level systems  
98. develop detailed program specifications, develop programs, set up system test parameters, install, and test the new system  
100. development of application software using a programming environment  
101. structured, event-driven, and object-oriented application design  
103. develop program tests and system tests  |
| Phase 4 Documentation and Presentation | Technical and user documentation, Prepare for and conduct presentation | transition | 94. to implement a documentation plan  
109. implement, and discuss project close down  
117. to present a system design, test plan, implementation plan, and evaluation, in written and oral form  |
| Throughout all Phases | Project scheduling, maintaining project workbook, journaling, peer evaluation | inception, elaboration construction, transition | 79. develop skills for effective interpersonal communication  
80. small group dynamics as related to working with users  
84. quality metrics for assessment of customer satisfaction  
85. the use of a professional code of ethics to evaluate specific IS actions  
87. develop agreements describing work to be done, and to commit, rigorously complete and self-evaluate agreed work  
94. use project management, project standards, and a system implementation plan  
105. design a project development and implementation plan  
106. develop and practice essential project management skills  
107. use of project management tools and methods within the context of an information systems project  
112. proactive principled behavior and time management  
113. ensure attitudes necessary for successful team behavior including empathetic listening, consensus negotiation, conflict resolution, and synergistic solution finding, and to apply the concept of commitment and rigorous completion  
114. ensure goal setting and alignment of team activities with project obligations  
115. interactions with higher levels of management in selling project objectives and performing project management tasks  
116. explain life cycle concepts, and apply them to the course project  
126. management of time and interpersonal relations  
127. discuss performance evaluation consistent with quality management and continuous improvement  |

Table 3: Description of the four phases of the capstone project. Activities in each phase correspond to OOSAD stages, and satisfy specific IS 2002 learning objectives.
writes a brief summary of the activities performed on the project, along with any problems or issues encountered and the number of hours worked.

3) Technical documentation, including an entity-relationship (ER) diagram for data model, transcripts of requirements gathering notes and interviews, use case diagrams and descriptions, and UML class diagrams. These are typically done in Microsoft Visio and Word.

4) A Microsoft Access database (.mdb file) or Oracle or mySQL database scripts.

5) Java source code files in appropriate package folders.

6) A user manual in a Microsoft Word document.

7) JavaDoc technical documentation of all classes and packages created and used during the project.

8) PowerPoint file of student presentations.

The project workbook is a portfolio that can be used later for assessment purposes, and that students also use for demonstrating their skills to prospective employers upon graduation.

---

**Project Planning Services, Inc.**

**Description of Entities, Attributes, and Relationships**

PPS is a nationwide company, and has several home offices. Information about each home office will need to be stored in the database. This information includes the home office address, including street, city, state, zip, and PO box number. In addition, the home office’s main telephone number and fax number, as well as the name and email address of the home office’s division manager should be included. (NOTE: division managers are not consultants per se, they serve a purely administrative role).

Consultants are identified by a unique employee ID. Each consultant has a name and rank (the ranks can be senior, associate, or junior). Each consultant also has an email address and phone number. In addition, each consultant has a yearly salary. Consultants are housed in home offices. Each consultant belongs to exactly one home office and each home office will have at least one and probably many consultants.

Consultants work on projects. Each project is composed of one or more tasks. Tasks are essentially components of projects, so they cannot exist by themselves. Each project has at least one task, and each task is associated with one and only one project. A project has a unique project ID, a name, a description, a start date, and an expected end date. A project also has an indication of its status. Project status could be ongoing, completed, on-hold, or abandoned. Each project is managed by exactly one project manager (who is a consultant). Therefore, each consultant may be a project manager for any number of projects, or may not be managing any projects at all.

Each of a project’s tasks has a task number that uniquely identifies that task within the project. The task also has a name and description. In addition, the task has a priority level (vital, important, optional, or unimportant). Each task has a task status (not-yet-started, active, or completed). Finally, each task has an indication of its expected duration in number of days.

Projects are performed for clients. Each project is associated with exactly one client. PP may be performing any number of projects for a particular client. Client information includes the unique client ID, the client’s company name, and a full address composed of street, city, state, country, zip code and PO box. In addition, the main telephone number and fax number of the client should be stored in the database. Finally, each client will include information of the main contact person, including the contact person’s name, email address, and phone number.

As mentioned earlier, one of the purposes of this system will be to track a consultant’s time spent on projects and business expenses while working on projects. Thus, time-log data must be kept for a consultant. A consultant will have any number (at least one) of time-log entries. Each time-log entry is associated with exactly one consultant (in fact, time-log entries cannot exist without being associated with a consultant). Each time-log entry includes the date that the work was done, the number of hours worked, and a description of the work that was done. A time log entry is usually (but not always) associated with a project’s task. In other words, a time-log entry may be associated with exactly one task, or may not be associated with any tasks. Of course, a project’s task could be associated with any number of consultants’ time-log entries.

As for business expenses, these will be tracked in expense log entries. Similar to a time-log, an expense log entry exists only with respect to its association with a consultant. An expense log could be one of four possible types. It could either be an air travel expense item, an automobile rental, a meal expenses, a hotel room, or a generic “other” category. Each expense log entry must include the date the expense was incurred and a description of the expense. Finally, every expense log entry must be associated with exactly one project. A project may have any number of expense items associated with it.

Your task is now to generate an entity-relationship diagram that matches the above description, and then to construct a normalized database for Project Planning Services.

---

**Figure 3:** Description of entities, attributes, and relationships used in Phase 1 of capstone project.

---

6  Journal of Computer Information Systems  Spring 2008
Figure 4: ER diagram created by student team based on written case description.

Figure 5: Database design constructed based on ER diagram.
PROJECT SCHEDULE

As mentioned earlier, 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 preliminary 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). Figure 2 shows a Gantt chart with the expected schedule of the project. A finalized version of this Gantt chart with actual completion of tasks must be included in a team’s project workbook.

Each phase of the capstone project is designed to assess students’ mastery of a number of IS 2002 learning objectives. Table 3 lists the tasks involved in each of the phases, identifies the OOSAD stages emphasized for each phase, and includes the IS2002 learning units that are involved for each phase.

Phase 1: Database Development

In the first phase, student teams are presented with a verbal description of the business entities and their relationships. Figure 3 shows the description for one of the cases, Project Planning Services, Inc. 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, as shown in Figure 4.

Based on the ER diagram, a relational database is constructed, such as the one shown in Figure 5. This must be a normalized database structure, and will typically be constructed using MS Access, Oracle, or mySQL.

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, tokenize the read-in lines of text into strings representing the individual fields, 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), so that students begin to gain CBD experience.

In terms of OOSAD, Phase 1’s SDLC can be broken down as follows. The planning and selection stage involves reading and understanding the case description, and then selecting the database package to use (students generally choose Microsoft Access, although Oracle and mySQL are other options). The analysis phase involves identification of all the entities, attributes, and relationships based on reading of the entity/relationship/attribute description (Figure 3), and creation of the ER diagram. The design phase involves construction of the relational schema in the DBMS. Finally, in implementation, students write and
execute the utility program for populating the database, making use of their DBSource component and gaining experience in component usage. The finished product, a fully developed and populated database, becomes the input to the next phases of the project, in which teams design and develop the applications that will interact with the database.

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 two weeks. 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, as shown in Figure 6. Each use case must be described in detail in a written use case document.

In addition to the use case construction, project teams also develop their initial user interface designs during this phase. They must do this using some electronic medium. Some students may use a drawing program. Most students, however, use the Java IDE’s (e.g. Netbeans, Eclipse, etc.) form-building features and/or actually write the Java code in order to construct the user interface, realizing that this gives them a head start on the software construction in Phase 3. During the user interface design, student
must ensure that the user interface is consistent with the use case descriptions; that is, all steps of the use case must correspond with buttons, menu options, or other appropriate controls in the user interface. Typically, each use case will correspond with one screen of the user interface design.

In terms of OOSAD, systems analysis and design are the primary activities in this phase. Phase 2’s SDLC is broken down as follows. The planning and selection stage involves assigning tasks among the team members and scheduling interviews with the user (i.e. the course instructor). The analysis phase involves conducting the actual interviews for requirement gathering, and construction of use case diagrams and descriptions based on the interview results. The design phase involves construction of the user interfaces. In this phase, there is typically little in the way of implementation and operation, although some programming may be done in order to construct the user interface.

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. As George et al [5] state, construction is the longest phase of the OOSAD cycle. This involves actual computer programming and debugging, which is a particular challenge to most students. Based on analysis of the students’ reading logs, it is evident that more hours are spent on this phase than any other, and it is scheduled as the longest in elapsed time for the project (see figure 2).

Here, students write the code for the three application programs. As mentioned earlier, 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). Once again, students benefit from use of CBD during their development efforts. 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, providing yet another experience of CBD to the students. All applications challenge the students by requiring implementation of complex SQL queries embedded in their code; most of these queries involve joins and/or subqueries, and many include aggregate functions (especially those pertaining to bar chart functionality).

Figures 7-9 show screen shots of one of the applications written by students in the group project. As can be seen, several Java GUI components are used, including tabbed controls, list...
boxes, combo boxes, buttons, labels, radio buttons, and text fields. In addition, the custom bar chart component that was created in the first half of the semester plays an important role in the application. Virtually all button clicks and list selections trigger events resulting in SQL queries that are sent to the multithreaded server (via the ClientConnector component) which processes them (using the DBSource component) via JDBC calls and sends results back to the clients (see Figure 1).

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. This puts considerable time requirements on the shoulders of the instructor, who is often inundated with long lines of students coming for assistance. Lecture time is considerably less during the group project than earlier in the semester; however, actual student-instructor contact time increases, as the instructor plays the role of customer, project manager, and mentor during several fact-to-face meetings with groups.

The bar chart component presents an opportunity for students to implement OLAP-like behaviors to their applications. Indeed many students make use of the event-handling capabilities of their bar chart components to provide drill-down functionality. For example, in the project planning services example, a user can initially be presented with a bar chart where each bar represents overall expenses for the list of clients, and then a bar can be clicked which brings another bar chart showing a breakdown of expense categories for that client. This is an effective and satisfying experience for students, because not only do they create the component that enables this drill-down in the first place, but they then actually use that component in a practical setting to facilitate drill-down for a realistic business application. Figure 9 shows an example of a 2nd-level bar-chart arrived at via drill-down on the bar chart from figure 8.

In terms of OOSAD, Phase 3 of the capstone project involves a heavy emphasis on coding and testing (i.e. the construction stage). The final deliverables for Phase 3 include the source code for all the applications and components developed by the students.

**Phase 4: Documentation and Presentation**

In this final phase of the project, students complete a user manual, the technical documentation, and a PowerPoint
Figure 9: Drilling down to get more detailed information about a particular client’s expenses.

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.

Students must also create a full technical document. The Java Development Kit (see java.sun.com) provides a useful tool called JavaDoc, which generates a series of web pages based on the source code of the Java project. These web pages contain an index and skeleton descriptions of each of the packages, classes, and public member variables and methods. If students have commented their code correctly, as they are expected to do, the comments are also part of the JavaDoc-generated documentation.

Finally, students prepare a presentation in PowerPoint, and present it to their classmates and to CIS faculty; this presentation is done instead of a final exam. Therefore, oral communication in a realistic professional setting is emphasized in this phase.

In terms of OOSAD, this is primarily a transition phase, as the system is deployed for the “user” (i.e. the instructor), and documentation and “training” (i.e. presentation) is emphasized.

JOURNAL WRITING

Research in education, particularly in complex problem domains, indicates that reflective observation is an important component of experiential learning [14]. In such situations, reflective journals or diaries can be useful learning tools [13]. In the computing realm, instructors have used this technique for a variety of purposes, including: assisting with learning and assessment in introductory programming classes [4], tracking attitudes of older students in end-user computing classes [15], and enabling students to document their problem-solving processes while engaging in programming tasks [11]. These writing efforts assist learning and teaching by forcing students to reflect on their problem-solving processes as well as the difficulties they encounter, providing a communication medium between
student and instructor in order to solve immediate problems, and providing information for instructors that can help with future course improvements.

During the capstone project, students are required to keep a journal of their systems development experiences. Students must write an entry for each day, and must (a) log the number of hours worked, (b) describe the work that was done, and (c) provide some sort of evaluation of the experience. This journal is turned in every week, and reviewed by the instructor. As a result of reviewing the journal entries for the week, the instructor may respond with advice or concerns. In addition, by logging time worked on projects and detailing the specifics of the efforts, these journals keep the students honest, as they know they will have to account for their time. Finally, the results are used to identify trends in student performance and concerns.

EVALUATING STUDENT WORK

Student work in the group project is evaluated on two dimensions, the group as a whole and the individual student. The group as a whole is graded on the quality of the software created, as well as accompanying documentation and the final presentation. Individually, students are evaluated on the quality and depth of their journal log entries. In addition, students are evaluated by their peers, and this peer evaluation has a measurable impact on the individual student’s overall grade on the project.

Evaluation of group performance is done during each of the four phases. In phase 1, the ER model, relational design, and database populating program are graded. After phase 2, the use case documentation and the user interfaces are graded. After phase 3, the programs are tested and graded on completeness, correctness, and user-friendliness. Finally, after phase 4, the remaining user documentation and the final presentation are graded. Grading in phases 1-3 are accomplished by the individual instructor of the course. Grading of the final presentation in phase 4 is done by consensus by a committee of faculty members who observe the presentation, ask questions of the student presenters, and run the programs.

As mentioned earlier, students evaluate each of their teammates. This is done to ensure that all members do their fair share of the work, and to give due credit to the team members who shoulder the lion’s share of technical work. It is typical in capstone group projects for effort levels to be unevenly distributed among team members. Also, it should be noted that in most teams, there will be one or two members who are particularly gifted at programming, and these are the ones who usually wind up doing most of the coding; this is typical of capstone group projects in computing courses [2, 19]. Similarly, there may be a subset of the team with particularly strong database skills, and these are the ones who shoulder most of the database design burden. So, it is important that the others pick up the less technical tasks, such as requirements gathering, use case modeling, and technical and user documentation.

CONCLUSIONS AND FUTURE DIRECTIONS

This capstone course and others like it that are springing up in IS programs around the world have great potential for teaching us as IS educators how to optimize the learning experience of our students. Ideally, a capstone course should accommodate a variety of learning styles and comprehensively cover the typical tasks of a learning cycle, while emphasizing the learning styles that typify IS majors [14]. A capstone course that integrates a wide variety of hard and soft skills in a rigorous environment, and that also incorporates documentary and assessment measures of data collection in the process of course delivery, can be used both to train the students and to empirically study the results of such training. This capstone course is used to facilitate both the training itself and the associated empirical analysis.

To summarize, the capstone course presented in this paper emphasizes the following elements: (1) component based development (CBD), (2) object oriented systems analysis and design (OOSAD), (3) integration of programming, database, systems analysis, and project management skills, (4) maintenance of a project workbook, and (4) use of journal writing for self-reflection. The course and its capstone project is geared to meet most of the learning units defined in the IS 2002 model curriculum, and to result in a student portfolio (implemented as the project workbook) that can be used by students during their job search and by faculty for the purpose of program assessment.

This course was piloted during the spring 2005 semester, and has been used as the capstone experience for CIS majors in every semester since. Although the course has naturally evolved over that time, its basic structure and format remains the same. The course with its capstone project has consistently shown itself to be a positive and effective learning experience, as evidenced by assessment findings regarding student performance, student survey and focus group results, and employer comments, which are collected and analyzed yearly as part of the normal assessment process.

Indeed, these assessment findings are often used to make improvements to the course, as one would expect. Several improvements and enhancements are currently planned and underway. First, the interface between the multithreaded server and the client connector component will be modified to involve an XML format for the data transfer; this is in keeping with the current standard for Internet-based communication between applications. Second, more emphasis will be placed on building components during the early part of the semester, so that the development effort during the group project can involve more component assembly and less algorithm coding. Third, the option for Java Server Pages or Servlets will be included for the server-side programming component, to generate a genuine web application experience. Fourth, UML modeling will get more emphasis, adding sequence diagrams to the already required use case and class diagrams. Finally, the journal writing portion of the assignment will be enhanced to make it more relevant for the student’s learning experience.

REFERENCES


