A WEB-BASED KNOWLEDGE MANAGEMENT SYSTEM FOR INFORMATION TECHNOLOGY EDUCATION

HONORS THESIS

Presented to the Honors Committee of

Texas State University-San Marcos

In Partial Fulfillment of

The Requirements

For Graduation in the Mitte Honors Program

By

Pamela Gayle Kernstock

San Marcos, Texas

December 2006

ACKNOWLEDGEMENTS

I’d like to thank Dr. Sam Lee for being such a patient, caring thesis supervisor; one who is dedicated to student success. Live long and prosper.

I also want to thank Dr. Mayur Mehta, Dr. Ju Long, Dr. John Plummer, Dr. Jaymeen Shah, Dr. Vivek Shah, and Dr. Marcos Sivitanides for being such great customers.

Special thanks go to my husband, Chris for being incredibly supportive in every way. Hugs and thanks go to my son, Charles, for his magnanimous help around the house.

CONTENTS

ABSTRACT

This paper describes a dynamic knowledge management system for use in an educational setting. It is an attempt to transform tacit knowledge into explicit knowledge in order to benefit students in an academic program. The goal of this Web-based system is to focus on providing help for the most common issues, relevant points and frequently asked questions in specific CIS courses and projects at Texas State University, San Marcos.

INTRODUCTION

Students entering information technology classes possess a range of computer skills. The Internet is more popular than television or radio for 13 to 24 year olds (Harris Interactive 2003). Although the majority use the Internet (Lenhart, Madden, and Hitlin 2005), not all students know how to use Microsoft Office application software, or read and write programming code. When students study information technology, they usually take an introductory programming course. As they progress through the program, they encounter typical problems any new information technology student might experience. This includes troubleshooting software problems and reading code, grasping concepts and various other technical skills issues.

Time and money pressures for the undergraduate students are increasing. Tuition has increased at a faster rate than the resources to pay for it (Ryu and Doyle 2004). Currently almost 30 percent of full time students work more than 20 hours a week (Shireman 2005). Students increasingly want answers to questions after hours, when an instructor may not be available. Immediately available answers are more efficient for the student, and create value for an educational institution.

Possessing computer skills can make the learning experience more gratifying and can help the student focus on the concepts taught in the classroom. These skills include software installation, basic computer problem resolution, reading and troubleshooting programming code, general programming concepts, debugging skills, and software installation. Learning how to use a debugging tool is a separate skill. Debugging can be a time sink for the beginning programmer. Downloading and installing the developer software tools introduces another hurdle for students. The software development environment can contain idiosyncrasies that require a steep learning curve. An experienced software developer knows the difference between tool issues versus “buggy” code, while an inexperienced student can spend hours solving one problem.

Since educational institutions foster knowledge and learning, it seems natural to use a knowledge management system (KMS) to leverage competitive advantage (Rowley 2000) and enhance learning. Since students have less time to complete assignments, a knowledge management (KM) system can help them succeed. An instructor-centered Web portal is proposed. For student users of the KMS, access to fresh, relevant information is important. As courses and projects change every semester, some of the knowledge gathered becomes outdated. An updated online knowledge base for an educational institution can give students access to the most current help information**.**

CONTENTS

Knowledge Management

Mack and Byrd (2001) refer to KM as: "the methods and tools for capturing, storing, organizing and making accessible knowledge and expertise within and across communities. Communities may be scientific, academic, and business oriented, or government based." Explicit knowledge is precise, defined knowledge. Tacit knowledge is experiential, and not easily communicated, especially to a distributed student population. Michael Polanyi (1961) uses several anecdotes to describe tacit knowledge in his essay *Knowing and Being.* One anecdote is the allegory of perspective and tacit knowledge. Airplane pilots first discovered the outline of an ancient settlement in the early 20th century. Even after knowledge existed of this settlement, the outline could not be discerned from standing on the settlement; it could only be seen from the perspective of altitude. It is this perspective that makes tacit knowledge valuable to a student. The ability to access information and knowledge from the perspective of experience can be a valuable benefit in education programs.

Treating knowledge as an asset is not a new concept. Intellectual capital consists of the skills, tacit knowledge and experience a company possesses through the virtue of its employees (Coakes and Bradburn 2005). An experienced employee is worth more than a new employee because of the experience he or she possesses. Job experience has historically been an intangible asset. In the past twenty years, corporations worldwide have begun to see the potential competitive advantage in managing their intellectual capital (Mitchell 2000). The ability to leverage intellectual capital and human experience drives funding of many organizations' KM programs (Rowley 2000).

There are many issues to consider in KM, from the core question of the nature or type of knowledge, to managing this elusive asset. Philosophers, economists, computer scientists and other scholars regularly delve into KM.

Knowledge based web sites abound for computer users and programmers alike. Many small web sites are language-version-specific, down-level versions that are not helpful. Other sites promise more than can be delivered. It is the nature of technology to evolve. As it does, the web sites dedicated to various aspects are not necessarily kept up to date, yet are first to come up in a web search. On the other end of the spectrum are large, high visibility online knowledge bases that can be tremendously large and difficult for a beginner to navigate. These online knowledge bases are necessarily vast, meeting many customers' needs. Microsoft's Knowledge Base (Microsoft 2006), which is available at http://support.microsoft.com/search, is an example of a huge online repository where users can search by a product or keyword. A search on Microsoft's knowledge base performed in September 2006 using the keywords "pivot table" and "Excel 2003" returned 28 responses. Selecting one relevant answer from the many responses returned by such a search is daunting for beginners who are up against due dates on their assignments. Sifting through so much information uses a student's valuable time. Finding specific, up to date information on the internet can be time consuming. To ensure that the knowledge-based system is useful to students, the focus is kept small. Keeping the scope to a particular course or curriculum will keep the knowledge base manageable and easier for students to navigate and for administrators to update. It should focus on their specific needs, which will reduce the time spent by students wading through irrelevant information.

Knowledge Transformation

The acronym KMS stands for Knowledge Management System. It is an information technology system that stores information and makes it available as knowledge (Alavi and Leidner 2001). When the need to store information arises in an organization, a KMS is the solution many companies deploy.

Knowledge takes on different forms. When knowledge is considered explicit, it is often in the form of facts and figures. Tacit knowledge is the realm of experiences and unique observations resulting from possessing explicit knowledge and acting upon it in a given situation. Explicit knowledge could be the fact that a particular development tool is used for developing an application. The tacit knowledge would be the “feel” of the tool; the idiosyncrasies that are not documented but are very important to efficiently using that tool. Information technology systems can perform in specific ways, well within specifications, but consistently idiosyncratic nonetheless. Two machines may be entirely within specifications, yet the response time may vary. If system A, for example, is always slow to boot, that might be considered normal for that machine. Machine B may speed through the boot process. If an engineer knows this, when machine B is slow to boot, the engineer possessing tacit knowledge can diagnose the problem quickly. In the CIS program, WebSphere software will throw a certain type of error when a particular type of programming error is made. The tacit dimension to this error is that it is not obviously related to the mistake made. It is not documented, but it happens with consistency nonetheless. Experienced users know that this error really holds another meaning, and use that tacit knowledge to quickly fix the real error.

Sharing this type of knowledge may have a positive impact on students. It is hoped that the shared knowledge in this KMS can be a timesaver for CIS students.

Knowledge sharing, in general, involves transforming the tacit knowledge and explicit knowledge three ways. Marwick (2001) categorizes the ways in which knowledge transformation takes place in a business setting. Tacit knowledge transforms into explicit knowledge through a process called externalization. Socialization spreads tacit knowledge. Meetings, classes, and impromptu discussions all provide an opportunity for people to share their experiences. Before and after meetings are prime times to exchange information informally with peers. The golf course or other social encounters are additional classic venues for sharing tacit knowledge in addition to forming business relationships. Tacit knowledge transforms to externalized explicit knowledge through written documents. Internalization of explicit knowledge to tacit knowledge occurs during the critical reading of a document, and application of the explicit knowledge to unique situations in life. Reading and using the explicit knowledge creates more tacit knowledge, which is communicated yet again to others; to be documented and used to generate more tacit knowledge.

These avenues of transformation exist in the educational setting as well. Instructors use tacit knowledge to create explicit knowledge. Students share this explicit knowledge, where it transforms again into tacit knowledge upon application in assignments. Instructors and students share tacit knowledge during class, and in discussions. Knowledge changes its form, flowing through the educational community, from instructor to student and back again. This transformation process is illustrated in figure 1 (see figure 1).

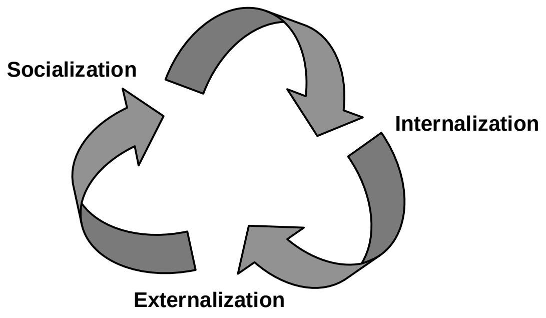


Figure 1. Knowledge transformation flow

Web-Based Knowledge Portal

This project is an attempt to quantify the collective experiences of students and instructors for the benefit of students in the Computer Information Systems and Quantitative Methods (CIS) classes at Texas State University – San Marcos. The aim is to provide help, direction, and answers to the most common and perplexing issues facing CIS students. Tacit knowledge becomes explicit when captured and made available to another student. Transforming tacit knowledge to explicit is at the heart of this project.

Jones, Provost, and Pascale (2006) explore a similar concept, which is the basis for this research. They describe a KMS portal designed for use by university researchers. This system focuses on the needs of a small group of students, resulting in a sharper focus on relevant knowledge. Personally interviewing the users (in this case, the students) gives the designer a higher level of interaction. During personal interviews, capturing nuances and reactions can give the designer greater guidance in developing the knowledge-based system.

Requirements Analysis

During the analysis phase of the prototype, six CIS faculty members and three students were interviewed to discover their needs in a KMS. Speaking directly to them gave them a chance to share their wants, wishes, and opinions. A discovery was made that a small database (MS Access) is being used for problems encountered with online test software in the Introduction to Computers (1323) course. A faculty member communicated the fact that the database is near capacity. This system design would be ideal for that application, if a more robust database is used. However, integrating the database with this prototype is beyond the scope of this project. The faculty interviewed communicated a desire for the application to cover generic concepts, main issues, and particular sticking points in the CIS program. General parameters include keeping topics relevant to current courses and projects and accessibility of this KMS to everyone enrolled in a course. One faculty member expressed a desire to communicate the importance of time management skills in regard to developing software programs for class projects. The KMS should encourage reading the textbook, not be a substitute for a textbook. Often the textbook material compliments the lectures, containing a substantial amount of course material. Reading a textbook for a technical class is a good habit that serves as a foundation for reading technical manuals, an essential skill for any IT professional. Other suggestions include multimedia clips instructing students on software image installation and configuration. The students should have the opportunity to collaborate on input; this will increase acceptance and use of the KMS.

Every semester, several issues repeatedly arise while students first learn to program. Students have trouble reconciling the code execution versus the intention of the code execution. Reading and following code through its logical path is a useful skill, yet many students don't realize the importance of this skill. Understanding object oriented concepts is another stumbling block for new programming students. The skill of stepping through an array or loop is a rough spot for inexperienced programmers. The ability to use the development environment and the accompanying debug tool adds another burden on top of learning to write and troubleshoot code. Above all, the KMS should not compromise the university honor code.

The students interviewed asked for tutorials on loading development tools onto their personal computer, as well as a deeper understanding of those development environments. They also expressed a desire to understand concepts more thoroughly. Students expressed a desire for a smaller and more focused knowledge base than MSDN. They also want an FAQ section. Students want to learn how to troubleshoot programming code methodically. Stepping through a multiple array and other complex functions seems to be a perennial rough spot for novice programmers. Another suggestion involved a way for the instructor to push clarifications and announcements to the class using the site during lab exams. This was suggested as an alternative to verbally making an announcement during the exam, thus breaking students' concentration. Table 1 lists the common issues that are particular to each of the courses covered in this project.

Table 1. -- Difficulties in particular CIS courses

Course	Name	Students	Common issues
1323	Introduction to Microcomputer Applications	TXState	• Circular reference errors
2324	Visual Programming I	CIS	• Value statements • Development Environment Software download and installation • Reading and interpreting code
3325	Visual Programming II	CIS	• Object oriented concepts and approaches. • Development Environment Software download and installation
3360	E-Business Application Design and Development	CIS	• Development Environment Software download and installation
3389	Business Application Programming III	CIS	• Java error message interpretation and trace stack interpretation • Logic errors, debug techniques & tips general and tool specific • Loops and Array combinations
4318	Advanced Business Application Development	CIS	• Advanced Object concepts
4360	Web Server Application Development	CIS	• File and permission management

System Capability

To obtain the anticipated benefits, the KMS shall include these capabilities:

- Knowledge DBMS contains bug issues both general and specific to projects and development environments
- Dynamic in nature to respond to new critical issues
- Supports queries through a website
- Website is straightforward and simple to use
- Maintains history of sessions
- Data is relatively secure

In addition to the Java classes in the KMS, Java testing modules will be used. Software testing serves several purposes. First, to ensure code execution results are appropriate. Second, regression testing helps detect bugs introduced during code iterations. JUnit is used to test the Java classes in this project. JUnit tests use assertions to set up the test, and if the test is successful, WebSphere displays a green bar.

Architecture

The Model-View-Controller architecture design pattern is used in this project. It was first described by Trygve M. H. Reenskaug while he was a visiting scientist at the Xerox Palo Alto Research Center in 1979 (Reenskaug 1979). The View layer is the website with which the user interacts. The Model layer encapsulates database access. One of the advantages to the MVC is that the data can be reorganized without impacting the user interface, and vice-versa. The Controller handles the events fired by the view, executing the model layer (Java Beans) which retrieves data from the database, returning the result to the user through the View (see figure 6).

System Development

Object oriented development processes will be used to design the prototype. The development process for the proposed knowledge-based system involves gathering requirements from the customers, in this case, instructors and students. Personal interviews are the type of "requirements gathering" activity used in this project. Other types involve group meetings, surveys and observing business processes. A relational database will be designed for storing transformed knowledge. Components developed using Java programming language will comprise the middleware. MS FrontPage will be used to develop the web site for the portal (see figure 5). The portal will have a keyword search function to find documented bugs and any workarounds needed to deal with them. The keywords will be based on project names, course numbers, class sections, and general keywords unique to course concepts. There are links to other relevant sites as well. A "frequently asked questions" page (FAQ) is included. The course textbook companion sites, the tutoring department, the CIS department home page, and a site map round out the choices for the web site user. The potential user (student) will use the Internet to access the portal hosted on a web server, which will access the database server (see figure 5). The prototype knowledge-base system will contain couple of examples of each type of knowledge made available: tutorials, video clips, project specific knowledge, and frequently asked questions. Instructors and CIS tutors will provide the prototype content. The database will hold information that the CIS student can access and use to help them learn new concepts.

The application will support lecture material and concepts for each of these courses:

- Introduction to Microcomputer Applications CIS 1323
- Visual Programming I CIS 2324
- Visual Programming II CIS 3325
- E-Business Application Design and Development CIS 3360
- Business Application Programming III CIS 3389
- Advanced Java Programming CIS 4318
- Web Server Application Development CIS 4360

Object Oriented System Development

Object oriented system development is the process of finding logical solutions to a problem and applying the solution in the form of objects. OOD is the system design that results from the analysis of user requirements, and is used to write the programming code that makes the system function.

Objects can be defined in several ways, but objects generally act upon each other, having roles and responsibilities. Objects are units of computer memory that contain the characteristics of a real object. An object has a state (attribute) and a behavior (method). Object reuse is a hallmark of object oriented programming. Once the execution has completed, objects can be cleared out of memory and used again in another routine. Java relies on reuse to streamline operations by assigning a value to an object until it is released by memory, then using the object to contain another value. This cuts down on the amount of code that needs to be maintained by programmers. Reuse also increases quality, as proven code is being repurposed. The modular nature of classes makes it possible to make changes to parts of the design without having to re-design the whole application.

A characteristic of OOD is the use of an iterative design process. After the first round of business requirements are gathered, the designer moves through the design inception, elaboration, construction, and transition. As more requirements are gathered or discovered, a new iteration of design is undertaken. This is a change from the traditional design approach. Programming routines are updated to support the requirements. This process of iterative development can be repeated as often as necessary, and some aspects of the design can be more iterative than others. This is where the modular design of Java classes is practical. If a Java class needs to be either added or removed, it does not substantially change the entire design. An analogy of this modularity can be expressed as Lego® building blocks, where each block is a Java class. OOD uses the Unified Modeling Language (UML) is a result of a collaborative effort by James Rumbaugh, Grady Booch, and Ivar Jacobson; three pioneers in the world of object oriented design methodologies. UML is used to model objects throughout the design cycle as well as describe the various components, views and functions of a system. In UML, there are functional, static, and dynamic views of the system. Use cases, class diagrams and state charts all communicate different views of the system. Two use cases are detailed in this project (see tables 2 and 3). A use case models the behavior of one task.

Why choose Java over another language, like .NET? A Java application can be used in any operating system. The modular nature of a Java application means functionality can be expanded in the future by adding libraries of specialized code. Third party libraries can be imported into the Java application when a new feature needs to be added. Some examples of third party libraries are those written to enable portable devices such as personal digital assistants using Bluetooth® technology (Spinellis 2006).

As Sun® Microsystems moves toward an open source paradigm for Java, more developers will have access to the language, and more companies will adopt it to save money.
A disadvantage to using Java is that excessive hiding can affect bug discovery. Heavy reliance on inheritance can cause problems with maintenance, as inheritance can sometimes be misapplied in the absence of sound design principles (Goth 2002).

Software Used

Database

Although Microsoft Access is used for this prototype, it is not recommended for a fully deployed version of this KMS. Microsoft SQL Server is recommended for a deployed KMS application. The differences between the two databases can explain why SQL Server is a better choice. MS Access is best used in a Windows desktop environment, not a live web environment. It is made for situations where the lifecycle of the project is short, the use is local, and is geared toward a small, limited database solution. This makes it ideal for a small prototype. Although it can also be developed to a certain extent by professional developers, it is oriented toward end users (see figure 2). The Access JET database engine is prone to corruption, especially when stretched to its limits of 2 Gigabytes of data. MS Access is not as secure as SQL Server. On the other hand, SQL Server is a better choice for a distributed architecture such as a web application. It must be developed by IT professionals, and costs more, but offers more capacity and functionality. The primary language used in Access and SQL Server is Structured Query Language, or SQL.

Application Development Tool

WebSphere Studio Application Developer 5.1.2 (WSAD) is the Java application development environment used. Java 1.3 is used, with some XML and HTML. Of the Java solutions available, Servlets, Java Server Pages (JSP), and Java Beans are the types used in this project. Each has a distinctive set of tasks in the MVC architecture. XML, Extensible Markup Language, is a framework to define a standard way of adding markup to documents. XML is used to share data and its format across heterogeneous information systems using the HTTP protocol. It is generated automatically with dynamic web pages designed in WebSphere.

User Interface Development Tool

Front Page 2003 was used in the initial development of the user interface. Cascading Style Sheets were used for the layout design. The template choice was originally designed by Andreas Viklund. The primary language of FrontPage is HTML. Cascading style sheets are also used. CSS is a way to apply a design to all the site pages. If a design change is needed, one text file can be changed and it will cascade throughout the document. It can be over-ridden and changed in particular places when the situation warrants (see figure 3).

Tutorial Software

Camtasia Studio is a software package for recording, editing and sharing high-quality screen video over a range of media. It was used in creating the multimedia tutorial for this prototype.

Challenges

One of the hardest aspects of the design phase was the database design. Being an average database student, this afforded an opportunity to work on design skills. The database design was too complex at the beginning of the project. Although a bug tracking tool was the original project idea, a knowledge base is a better solution to the customer requirements. It also became apparent that the scope of the project could only include a knowledge management system and not a bug tracking system. Through the first iteration, many tables were discarded and the design was revised. Database normalization was a challenge; clarifying each table's function in the context of the business rules helped normalization. Numerous design decisions were made. Knowing why a table had a one-to-many relationship with another table, when to use association tables and other database design are two examples. After the database was functional, adjustments to the design were still being made. In the Java code development, data retrieval using Java Beans presented an exercise in detailed work. Setting the values of the input parameters for the result set object must match the prepared statement object exactly. This is an easy concept, yet one minor error results in a malfunction of the entire method. Using the CSS template in the FrontPage software worked well; however, porting it to WebSphere posed another challenge. The file path to the style sheet defaulted to a relative file path in WebSphere, yet CSS would not cascade in WebSphere without an absolute file path. Also, WebSphere threw an error on the upper case HTML tags used in FrontPage so all the tags had to be changed to lower case, a time consuming endeavor.

CONCLUSION

Instructors and students share knowledge. New tacit knowledge is formed in the process of sharing and applying explicit knowledge. Students come from varied backgrounds; needing quick help in subjects in which they may not have prior experience.

Education can leverage the power of internet applications to share tacit knowledge between instructors and students, helping the students and creating competitive advantage for the institution. If a student is struggling with a particular topic in a course, a focused knowledge base may be of great help. This model can be applied in many academic settings; the small scale of such a knowledge base is ideal for a department or a program.

Future studies related to the use of a focused knowledge base should include the following:

- Comparisons of course completion rates using a KMS against no KMS.
- Studies focused on the effects of the knowledge base upon student grades.
- Impact of the knowledge base on teaching effectiveness of concepts and skills.
- Ease of use of the web interface used in the knowledge-base system.
- Measurements of the impact the knowledge-base system on instructors.

An area for future development could involve integrating other modules with the KMS. An example if a module is the database for the online test program being used in the Introduction to Computers (1323) class mentioned previously. In addition to expanding the size, future knowledge bases could be based on collaborative software; where students also contribute to the knowledge base. Instructors should still be the only ones authorized to edit and add the content to the knowledge base.

This will keep the student’s focus on the course content instead of learning to use another software tool. A benefit of a collaborative effort could include increased student use of the knowledge base and provide a venue for students to articulate their knowledge. No matter who contributes, the instructors will be responsible for approving the content. This ensures the knowledge contained is accurate and relevant to the program. Sharing this knowledge could help struggling students succeed.

BIBLIOGRAPHY

Alavi, Maryam, and Dorothy Leidner. "Review: Knowledge Management and Knowledge Management Systems: Conceptual Foundations and Research Issues." *MIS Quarterly* 25, no. 1 (2001): 107.

Coakes, Elayne, and Anton Bradburn. "What is the Value of Intellectual Capital?" *Knowledge Management Research & Practice* 3, no. 2 (May 2005 2005): 60. Proquest.

"Teens and Young Adults Now Spend More Time Online than Watching Television." in Carat North America [database online]. Sunnyvale, CA May 2003 [cited 2006]. Available from http://docs.yahoo.com/docs/pr/release1107.html.

Jones, Nory B., Darylyne Provost, and David Pascale. "Developing a University Research Web-Based Knowledge Portal." (2001).

Mack, R., Y. Ravin, and R. J. Byrd. "Knowledge Portals and the Emerging Digital Knowledge Workplace." *IBM Systems Journal* 40, no. 4 (2001).

Marwick, Alan D. "Knowledge Management Technology." *IBM Systems Journal* 40, no. 4 (2001).

Mitchell, K. D. "Knowledge Management: The Next Big Thing." *Public Manager,* Summer 2000. Database on-line. Available from Proquest document ID: 62029833.

Polanyi, Michael. "Knowing and being." *Mind*, *New Series* (1961).

Rowley, Jennifer. "Is Higher Education Ready for Knowledge Management?" *The International Journal of Educational Management* 14, no. 7 (2000): 325. Available from Proquest Document ID: 115923838.

Rumbaugh, Jacobson, Booch. *The Unified Software Development Process.* 1st ed. Boston, Mass.: Addison-Wesley Pearson Education, 1999.

Ryu, Mikyung, and William Doyle. *Measuring Up 2004: The National Report Card on Higher Education.* National Center for Public Policy and Higher Education, 2004.

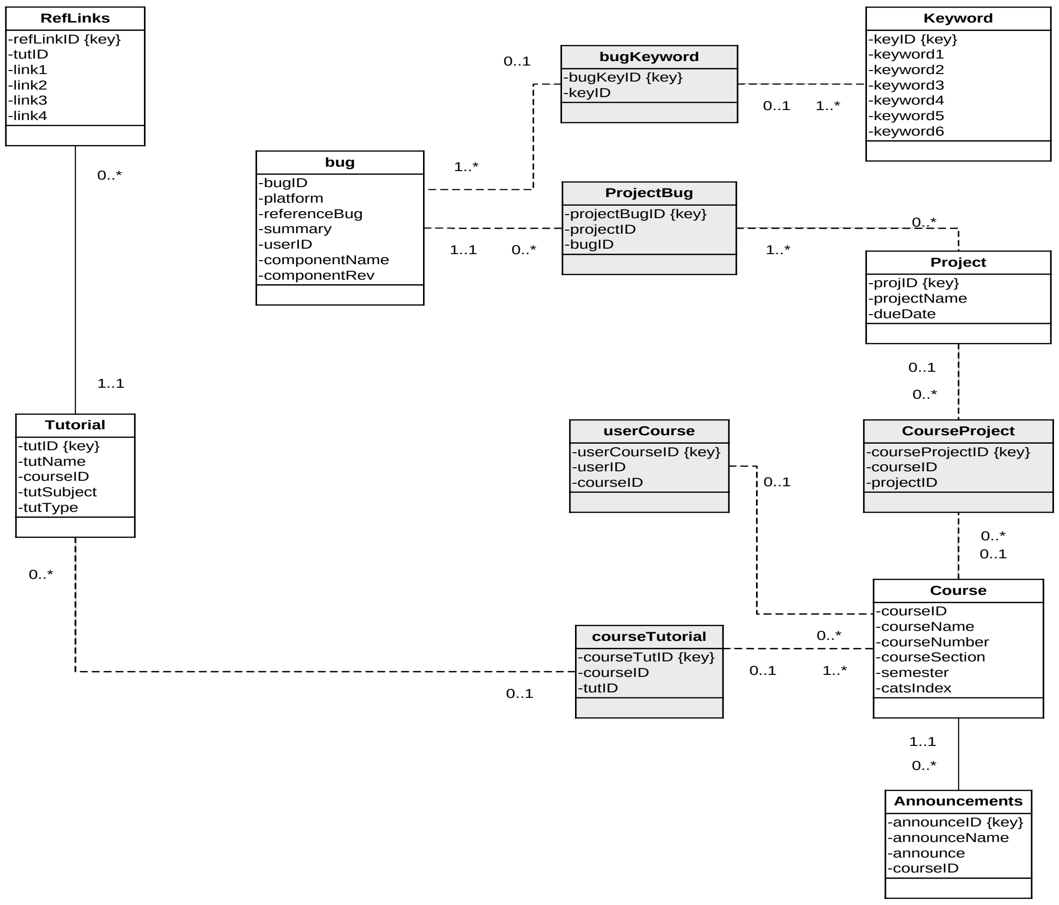
Satzinger, Jackson, Burd. *Systems Analysis and Design in a Changing World.* 3rd ed. Boston, Mass.: Thompson Learning, Inc., 2004.

Shireman, Robert. *Reducing the Dangers of Debt - Student Loans could be a More Positive Tool in College Access Efforts.* National Center for Public Policy and Higher Education, 2005.

Viklund, Andreas. "Cascading Style Sheet, andreas08." (2006) Database on-line. Available from OSWD.org.

APPENDIX A

Figure 2. Database Class Diagram showing database schema

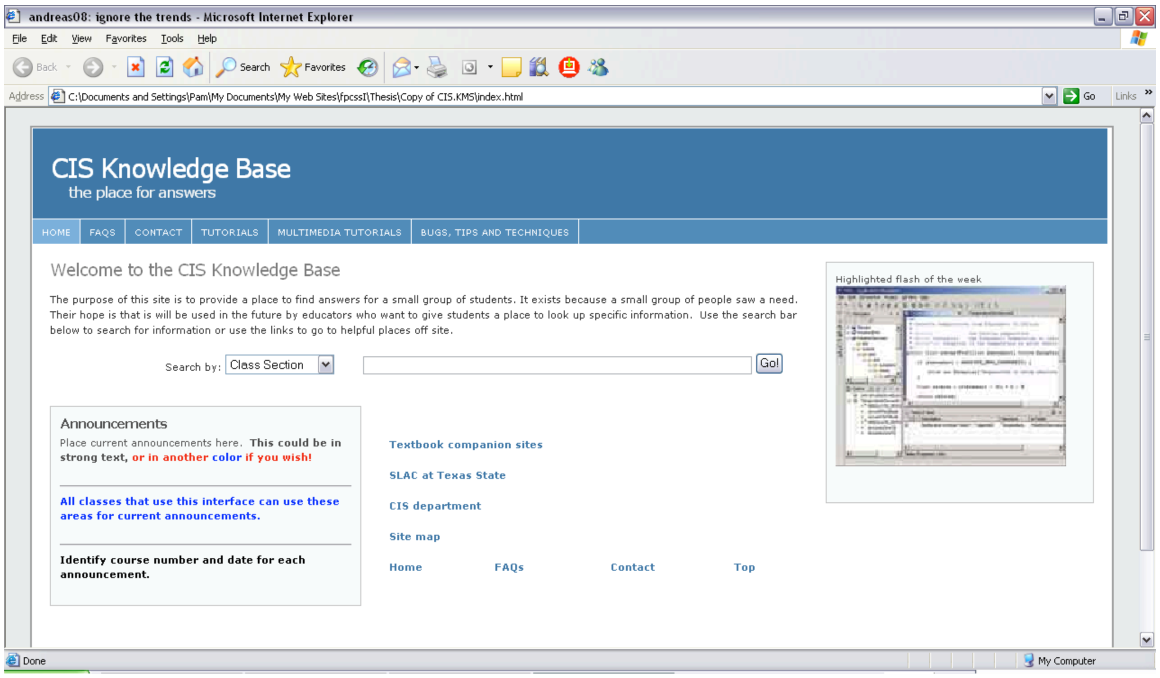


Use Case Name:	1. Look up a bug	
Scenario:	User looks up a bug on the web portal	
Triggering Event:	User navigates internet browser to portal	
Brief Description:	User navigates onto bug site and initiates a query. The user chooses a query filter if desired, and performs a search by keyword	
Actors	Users	
Related Use Cases:		
Stakeholders:	Students, who look up bugs Instructors, who make bugs available for viewing.	
Preconditions:	Access to portal DB server up DB populated with bug data	
Postconditions:	Search must be performed. Bug record must be related to keyword. Result displayed on web site	
Flow of Events:	Actor	System
	1. Navigates to web site 2. Chooses filter option from drop down box, Enters keyword (s) into search bar, Selects enter (go) button 3. Chooses one result	1.1 Home page presents search choices 2.1 Filters by option choice 2.2 Form accepts input and requests a filtered query through controller, to the DB 2.3 Controller retrieves keyword result from DB, sends to browser through JSP view. 3.1 Displays results to user on web site
Exception Conditions:	If bug is not listed, student will contact instructor to communicate a possible bug and gain assistance. Student can seek help at SLAC.	

Table 2. UML Use case 1, modeling a “Look up a Bug” task

Use Case Name:	2. Look up a tutorial or multimedia tutorial	
Scenario:	User looks up a tutorial on the web portal	
Triggering Event:	Student navigates internet browser to portal	
Brief Description:	Student navigates internet browser to portal and chooses the tutorial button. Student selects appropriate tutorial, or a multimedia tutorial	
Actors	Students	
Related Use Cases:	Look up Multimedia Tutorial Link to textbook companion website	
Stakeholders:	Students, who look up tutorials Instructors, who may update tutorials	
Preconditions:	Access to portal DB server up DB populated with tutorial data	
Postconditions:	Button links to tutorial page. Tutorial page displayed in browser window	
Flow of Events:	Actor	System
	1. Navigates to web site 2. Selects tutorial or multimedia tutorial page. 3. Chooses a tutorial or a multimedia tutorial.	1.1 Home page presents tutorial link 2.1 Hyperlink to tutorial pages 3.1 Retrieves tutorial from DB and Displays tutorial to user in browser window
Exception Conditions:	If tutorial not found, student can seek assistance from SLAC or instructor, or choose the link to the textbook website.	

Table 3. UML Use case 2, Look up a tutorial or multimedia tutorial


andreas08: ignore the trends - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Search Favorites

Address C:\Documents and Settings\Pam\My Documents\My Web Sites\fpcssl\Thesis\Copy of CIS.KMS\index.html Go Links

CIS Knowledge Base

the place for answers

HOME | FAQS | CONTACT | TUTORIALS | MULTIMEDIA TUTORIALS | BUGS, TIPS AND TECHNIQUES

Welcome to the CIS Knowledge Base

The purpose of this site is to provide a place to find answers for a small group of students. It exists because a small group of people saw a need. Their hope is that is will be used in the future by educators who want to give students a place to look up specific information. Use the search bar below to search for information or use the links to go to helpful places off site.

Search by: Class Section Go!

Highlighted flash of the week

Announcements

Place current announcements here. **This could be in strong text, or in another color if you wish!**

All classes that use this interface can use these areas for current announcements.

Identify course number and date for each announcement.

Textbook companion sites

SLAC at Texas State

CIS department

Site map

Home FAQs Contact Top

Done My Computer

Figure 4. Web-Based Knowledge Portal Home Page
Cascading Style Sheet Courtesy of Andreas Viklund

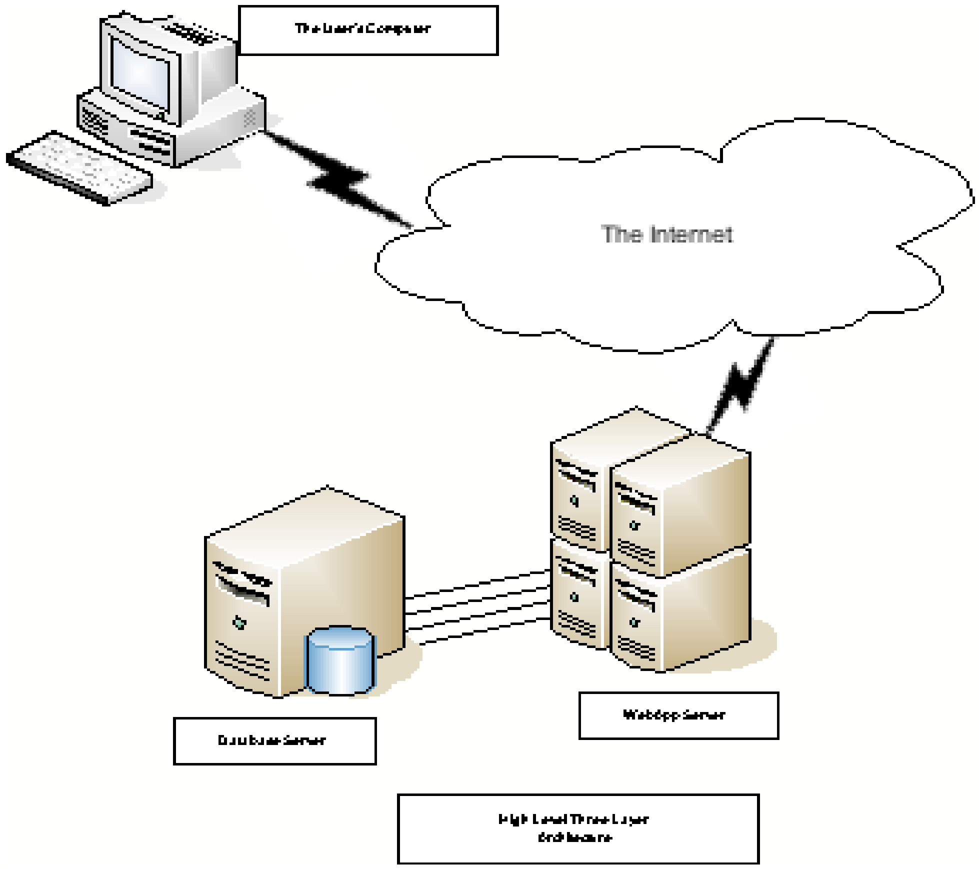


Figure 5. System Architecture of the Web-Based Knowledge Repository, Showing Major Components

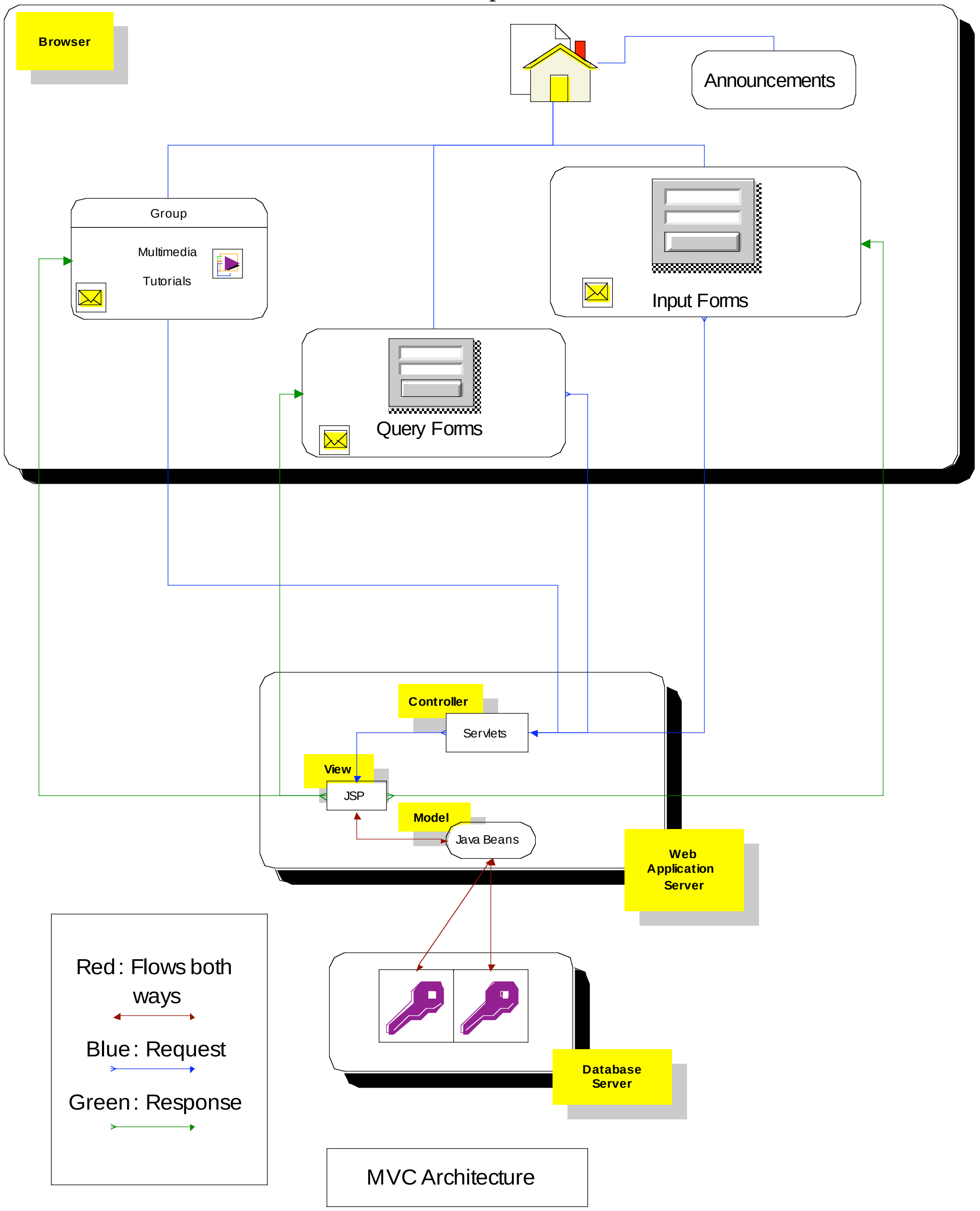


Figure 6. Model-View-Controller Architecture Shows Data Flow Between Components

APPENDIX B

The complete source code is on the accompanying CD.

SQL

```
SELECT project.projectName, bug.bugId, bug.referenceBug, bug.componentName,
bug.componentRev, bug.platform, bug.summary
FROM (bug INNER JOIN projBug ON bug.bugId=projBug.bugId) INNER JOIN project ON
projBug.projectId=project.projectId
WHERE project.projectName='Soundstream';
```

CSS

```
/**************** Body and tag styles ****************/

*{margin:0; padding:0;}

body{
font:76% Verdana,Tahoma,Arial,sans-serif;
line-height:1.4em;
text-align:center;
color:#303030;
background:#e8eaec;
}

a{
color:#467aa7;
font-weight:bold;
text-decoration:none;
background-color:inherit;
}

a:hover{color:#2a5a8a; text-decoration:none; background-color:inherit;}
a img{border:none;}

p{padding:0 0 1.6em 0;}
p form{margin-top:0; margin-bottom:20px;}

img.left,img.center,img.right{padding:4px; border:1px solid #a0a0a0;}
img.left{float:left; margin:0 12px 5px 0;}
img.center{display:block; margin:0 auto 5px auto;}
img.right{float:right; margin:0 0 5px 12px;}
```

Java Bean

```
public ArrayList findByProjectName(String argProjectName){

        Connection con  = null;
        ArrayList list = new ArrayList();
        boolean ok = false;

        try {
                //register the JDBC driver
                DriverManager.registerDriver(
                new com.inzoom.jdbcado.Driver());
                //connect to a database
                con =
DriverManager.getConnection("jdbc:izmado:Bug;IzmDllPath=C:\\jadozoom\\izmjniado.dll");
                PreparedStatement stmtProjectN = con.prepareStatement("SELECT
                project.projectName, bug.componentName, bug.componentRev, bug.platform,
                projBug.summary FROM project INNER JOIN (bug INNER JOIN projBug ON bug.bugId =
                projBug.bugId) ON project.projectID = projBug.projectId WHERE
                (((project.projectName)=?))");
                stmtProjectN.setString(1, argProjectName);
                ResultSet rsProjectName = stmtProjectN.executeQuery();


                while (rsProjectName.next()){
                        Bug object = new Bug();
                        object.setProjectName(argProjectName);
                                        object.setComponentName(rsProjectName.getString(2));
                        object.setComponentRev(rsProjectName.getString(3));
                        object.setPlatform(rsProjectName.getString(4));
                        object.setSummary(rsProjectName.getString(5));
                        list.add(object);

                        }

                                } catch (Exception ex) {
                                        ex.printStackTrace();
                                } finally {
                                        try {
                                        con.close();
                                        } catch (Exception ex ){}
                                 }
                                return list;   }
```