https://doi.org/10.48009/1_iis_2005_94-100

ENHANCEMENT OF THE CLASSROOM PERFORMANCE SYSTEM

Chuck West, Bradley University, west@bradley.edu

ABSTRACT

The participants of the IACIS Fall 2004 Conference in Cancun were given the opportunity to experience first hand the Classroom Performance System (CPS) by eInstruction. The CPS environment was initially designed for the K-12 classroom and later marketed to colleges and universities as a means to involve students much as it did the participants of the IACIS conference. The IACIS experience led to the procurement of a basic system and introduction of a new level of technology to the junior/senior information technology class. The initial student feedback led to a design and development project in our senior capstone class. Although the system as demonstrated addresses the basic quiz, test and reporting aspects of a classroom, it fails to take full advantage of the remotes and has several limitations. This paper shows how the capstone students reverse engineered the CPS software and then developed a prototype to demonstrate the enhancements. The enhancements included a real-time feedback mechanism from the student to the instructor to inform the instructor of how the material was understood by the student. A second enhancement eliminated the limitation of only five answers to any question. Lastly, additional functionality enabled the storing student response information on a LAN server.

Keywords: Capstone course, Programming, Visual Basic, System Design, System Development

INTRODUCTION

The Classroom Performance System (CPS) environment enables the instructor to create an on-line quiz or test, present it to the students via a monitor or projection device, and have the student answer the questions using an inexpensive remote control. The CPS environment consists of three components. First, a multi-part computer program that enables the instructor to enter true/false, yes/no and multiple choice questions. These questions can be text, or text with graphic image in nature. A second part of the program presents the questions to the students on a monitor or screen through a video projector. This part of the system visually notifies each of the students that their response has been recorded by displaying the transmitters ID in reverse video below the question and answer form. The student can change their response until the time the next question is presented. After all of the students have answered the question, the system can display the correct answer and what portion of the class answered the question correctly. A third part of the program records the responses from each student, stores the responses in a database and produces numerous reports for both student and instructor use. Attached to the computer that is used to present the questions is a receiver, the second major portion of the system. This receiver is attached to the computer via the USB port. It is a one-way device that multiplexes the signals from the third component, a hand-held IR transmitter. Each student is given, or purchases, an inexpensive IR transmitter. Each transmitter has a unique ID that is registered to the student.

The advertisement for the CPS software [1] states the system involves everyone, facilitating feedback from every student and allows each student to participate simultaneously. In addition the advertisement states that it embarrasses no one, allowing the instructor you to elicit a response from every student without embarrassing anyone for a wrong answer. It also states that it reaches out to a new generation with technology that is dynamic and relevant to today's young people [3]. During the fall 2004 semester, the CPS system was used in three 300 level information technology classes. Approximately 85 students used the devices on three occasions to take quizzes over assigned material. Again during the spring 2005 semester, 41 students in two 300 level classes used the devices on three separate occasions. The system performed as advertised. The students were taken by the technology. They were excited and intrigued. At the end of the semester, they were challenged to use the CPS to write quiz questions for their research paper presentations. This led to team building and much interaction but also uncovered some shortcomings and usability issues.

Resolution of the shortcomings was seen as an opportunity for the MIS capstone class. Inputs from the instructor and the students were used to develop the design specification for a complementary system that would address the shortcomings of the existing system. The capstone class expanded upon the specifications and developed a working prototype that demonstrated the feasibility of the system enhancements. This paper reflects on some pertinent details of the existing system, explains some of the development challenges, reviews the outcome and summarized additional future direction.

SYSTEM SHORTCOMINGS

As is, CPS lacks several functions desired in a university classroom. First, CPS does not allow the creation of a quiz or survey question with more than five choices. There are eight buttons on the controller but only five can be used for answers. This limitation constrains the use of CPS to multiple-choice quiz or test questions with five or fewer possible answers. It is desirable to use the system for more complex questions and for other classroom events such as voting or ranking student team performance. During the information technology class, the students gave presentations on their research topics. It was impossible to use CPS to allow the students to vote on which of the seven teams they thought performed best. A system enhancement needed to be developed to allow the instructor to put together a multiple-choice question with six, seven or even eight possible answers which would add the functionality of classroom voting.

Secondly, CPS does not allow for a linear real-time feedback from the student to let the instructor know, as they are lecturing, that the students do not understand the current topic under discussion. In large classes where student feedback is minimal, it is extremely difficult to get a reading on how well the instruction is understood. This was an opportunity overlooked by the eInstruction developers. A feature needed to be developed to allow the student to use the remote controls for more than quiz and test taking.

A third issue is that CPS lacks the ability to retrieve quizzes from a central LAN based database and to store the results of quizzes and tests in a network environment. CPS is designed to be executed on the same system where the quiz or test resides. In addition, the results are stored on the same system. In an environment where classroom computers are shared, this poses a

problem. Networking environments are common in classrooms. Taking advantage of this is a needed enhancement.

The fourth issue was one of usability. Issues of usability arose with the creation of quiz or test materials. Use of the remotes by the student did not pose any significant issues. Students and instructors found the CPS system cumbersome to use and far less than intuitive to create quiz or test materials. They had trouble entering questions and answers, converting from one type of question to another and going from one question to another without losing their work. The students had significant trouble understanding the directory structure used by the CPS system to connect text with graphics. Over 80% of the student developed quizzes that included graphic images failed to display properly because the graphic image is not stored in the quiz but in a separate space that requires a specific directory structure. When the students tested their quizzes on their computers the question and graphic displayed properly. When they transferred the quiz to a common machine, the graphic image was lost. In addition, when the students created quiz questions, they had a difficult time going back and modifying the question or the possible answers. Over half of the students complained they lost question information. Little or no training should be needed in a system of this type. Extensive usability testing should be performed to make sure the system meets this requirement.

Lastly, the students in the spring 2005 classes suggested a hybrid system that included the use of the remotes to answer questions that would be distributed in paper form. This would allow the students to individually spend more time on the questions they did not completely understand and to spend less time on those they immediately knew the answer to. CPS has one mode of presenting the questions in which each question is allotted the same amount of time. Giving the students the ability to split the time up as they needed was seen to be a great advantage to the student while at the same time it actually reduced overall time needed to take the quiz.

DEVELOPMENT CHALLENGES AND OUTCOMES

An opportunity to develop a system that demonstrated how the shortcomings could be overcome and how enhancements could be implemented arose during the spring semester. The MIS capstone course needed significant projects. Through this course the students learn how to analyze and solve business problems. They form teams and address real-world problems. During the spring 2005 semester, a team of four graduating seniors undertook the challenge to demonstrate how the functionality of the current CPS software could be enhanced. The champion of the use of CPS became the client for this team. Through a process of interviewing, sharing the experiences of the 126 students who had classroom experience with CPS, and getting first-hand experience with the remotes, receiver and CPS software, they developed the Teacher's Assistant Pro [2]. The Teacher's Assistant Pro (TAP) is a prototype that demonstrates how each of the shortcomings can be overcome and how additional functionality could be added to CPS. This software augments the existing software. It does not attempt to replace the entire system; it just presents a solution to overcome the various shortcomings. The goal for the prototype was to demonstrate in a working model a solution to the perceived need for enhancements and to demonstrate the desired increased functionality. All the functionality in the respective portions of the eInstruction system was included in the prototype. The system development cycle

implemented a continuous usability testing format. One of the requirements of the system was to require no training on the part of the student or the instructor.

The original intent was to work with eInstruction to implement the enhancements. Once the system was developed TAP functionality could be demonstrated to eInstruction for incorporation in future releases. Due to timing issues and ongoing development projects at eInstruction, TAP evolved into a working standalone system.

The decision to not completely rewrite the existing system and to concentrate on addressing the shortcomings and enhancements significantly reduced the magnitude of the project and made it possible to complete in a one semester timeframe. This decision eliminated the need for rewriting the true/false and yes/no formatted question sections. The team decided to write the new system in Visual Basic .NET. All of the basic functionality like feedback and recording provided in the original eInstruction system would be developed in the new system. The areas the student team and client agreed to concentrate on were the 1) creation of multiple-choice questions with up to eight possible answers, 2) a real-time feature for student feedback that would provide the students with a mechanism that would inform the instructor of how well the students were understanding the material being presented, 3) a LAN based storage system for quizzes and responses, and 4) an interface that would allow the students to enter the answer via the remote devices for questions presented on paper. In all cases the team would concentrate on the intuitiveness and usability of the system interface.

Reverse-engineering the protocol between the remote, receiver and the computer was the first challenge for the students. To make this part of the development less difficult, a consultant, Christopher Taylor, was brought in. He was a previous graduate of the University and was willing to work with the students to develop their product. Since the development of the system was the students' responsibility, Christopher Taylor was limited in his consulting activities to advising the team and prohibited from actually coding the solution. Unfortunately the students did not know were to begin. They came back to the client and asked for assistance. Through a web search for a serial monitor, the client discovered the software USM Monitor version 2.37 [5] on the tucows.com [4] web site. Through the use of USM Monitor software and the help of the consultant, the team reverse-engineered the interface protocol. It was determined when a button on the remote was pressed, that a packet of information was sent from the receiver to the computer that contained control codes and the unique ID of the remote and the button pressed. Because of the lack of understanding of communications protocols on the part of the student team, this portion of the development started first and ended last. It consumed the majority of the development time.

The next challenge to the team was to develop a mechanism by which the instructor could input a multiple-choice question with up to eight answers. The CPS system contained facilities for only text and graphics. The design for the new system included text, graphics and ultimately audio and video. This was a significant enhancement to the CPS product. Through a division of labor, the programmer also became the designer for this phase. This phase actually started before the remote, receiver and computer interface protocol was completed. Initially the programmer designed what he thought was a very straight forward method for entering the questions. Unfortunately it lacked the usability tenants desired. The team was familiar with usability testing

and designed a series of usability test to address the various implementation issues. Through usability testing, this as well as other problems, were identified and the interface re-written. Ultimately the interface became extremely intuitive.

The lack of a working interface protocol module stalled the development of the feedback mechanism and the alternate method for taking a quiz. The students were unable to develop an alternate method for testing their program without the actual remote/receiver interface. This had a cascading effect on the system development. The development of a LAN database was stalled. From a positive perspective, the delay gave the students time to appreciate what they had learned with regard to usability. When they finally did present a working feedback mechanism and alternate method for taking a quiz, both were intuitive in nature.

The alternate method of taking a quiz where the students were given the quiz in paper form and uses the remotes to enter their answers was simple in function. A matrix was developed with the ID of the remote along the vertical axis and the question number along the horizontal axis. The application knows which remote is being pressed and thereby does not require the student to position their remote on the appropriate row. Using the "<" (right/G button) and ">" (left/H button) on the remote, the student positions their cursor above the question they wanted to answer. They can answer the questions in any order they desire. Once they answer the desired question, the question number turns to yellow. After they verify their answer by pressing their selection again, the question number is color coded to indicate if their answer was correct (green) or not (red). The individual student's answers are not shared with the other students, only the correctness of the answer is displayed. Because of the limitation of only eight buttons on the control, questions are limited to six multiple choice answers for this type of quiz.

The design for the student feedback to the instructor is similar to the alternate method of taking a quiz. The "<" and ">" keys are simply pressed by the student to indicate, they do not understand or they do understand. The program counts the number of students that pressed a key over a period of time. A large, highly visible bar appears on the instructor's monitor. It reflects a period of time and the student responses during that period. The feedback mechanism is color coded using green to represent understandable, yellow as a warning for up to three negative responses and red to indicate the material was not understood. The amplitude of the bar reflects the number of students responding. It is possible to have a green bar and a yellow or red bar present. A combination simply means that some understand and some do not.

The students decided to use a LAN based implementation of Microsoft Access as the storage mechanism. The questions, answers and pointers to graphic, audio and video files were easily stored in a database. Responses were also stored in an Access database. Unfortunately in the prototype environment, anyone that had access to the server had access to the questions and results. This was a significant shortcoming. In the case of a catastrophic failure in which the quiz or test session is terminated, only the answers to the last question are potentially lost. A bigger problem was identified with the system. If the LAN connection was not available, the system does not work on the stand-lone system.

The design and implementation phases of the process were interlaced with usability tests. This iterative approach to testing resulted in a very intuitive system. The frequency of the usability

testing and of the interplay between team members and the client added to the "no surprise" finish.

As part of the development process, the students were required to develop user and program documentation. The user documentation was presented in a colorful seven part, 11 page user manual. The manual included an overview of the user interface, how to create and edit a class, how to add or edit student information, how to create a quiz, how to conduct a quiz, how to set and edit the roster, and how to conduct a class. The program documentation was presented in a 62 page document with screen dumps, program logic code and code documentation. Both exceeded the expectations of the client.

The final step in the capstone project was demonstrating the system to the client. The team demonstrated each of the pieces individually and then demonstrated the entire system to the client and an advanced Visual Basic .NET programming class. The students were questioned about the project in general and what they learned from it. The presentation and the question and answer session were videotaped for future use.

LESSONS LEARNED AND CONCLUSIONS

Many lessons were learned during the project. Throughout the project, the coordinator of the capstone project questioned if there really was a project and if the students could really accomplish the tasks. This project was one of the more ambitious projects entertained as a capstone project. The coordination of the students, hardware, specifications and consultant was difficult. The students lacked experience with projects this complex. The remotes and receiver were needed in the classroom as well as by the development team. This created a scheduling problem not unlike those in the real world. The specifications for the system were vague and not totally understood by all until the waning moments of the project. The students were introduced to scope creep. The consultant's time was at a premium. He had a very demanding full time position. In addition, the students did not successfully contact him until well after half of the way through the project. The complexity of the project was not the major issue. It was difficult but doable. The project did point out the level of education and experience needed to complete such a task in a timely manner. A second set of hardware would have made the process less challenging. The vagueness of the specifications was a bit too real-world. It was the first time the students had to deal with such a moving target. Scope creep and matching the talent to the task had a very negative impact.

From the client's perspective the communication with the team was the greatest challenge. The team met with the client early in the semester. The second meeting was not until the middle of the semester. During this time several of the specifications had changed and scope creep had begun. Subsequently, it was agreed upon to have meetings at least every other week. This resulted in a much smoother relationship. No formal mechanism was put in place to communicate the changing needs to the team. The need for communication between client and developer needed to be stressed in the initial few weeks of the project. In addition, communication between team and client must be stressed and possibly practiced before the capstone project begins.

The consultant found it difficult to work with the team [C.J. Taylor, May, 2005] and conversely the team had a difficult time working with the consultant. It was the first time the team had the opportunity to work with an outside consultant. There was a significant communication problem. The student tasked with communicating with the consultant tried so for several weeks via e-mail but to no avail. The consultant did not receive any the e-mail communications. It was agreed to by all that communications is paramount in a project, especially one with such limiting time constraints. The students did not know how to press the consultant for time without insulting him. The consultant was wondering why the team had not contacted him. This is a real-world problem and a possible area for additional training prior to the capstone class. The use of an outside consultant complicates the management of such a project. From a technical perspective, the consultant was needed to resolve some major problems but from the student's perspective, it was difficult working with someone outside their environment.

Due to the complexity and scope of the project, the students operated in a near panic mode for the last half of the semester. A key component of the project, interfacing with the remote, was getting in the way of the entire project. Unfortunately, the students focused on the reverse-engineering of the protocol used to interface the program with the remote and receiver and left other tasks unattended [M. Herff, April 2005]. They were not accustomed to working on a project in a modular fashion. The time needed to complete the project exceeded their expectations [N. Abderholden, May 9, 2005]. Even though they had experience with the resources necessary to solve a problem, when challenged with a problem that appeared foreign to them they did not know where to start. In addition, the capstone project has typically been the student's first experience with team programming. The curriculum may need to be adjusted to address these issues. The user manual was complete and very readable. The technical manual demonstrated their proficiency in documenting and explaining over 1200 lines of Visual Basic .NET code.

FUTURE DIRECTIONS

Although the project accomplished the agreed upon goals, enhancing the LAN database functionality is needed. Database security and efficiency issues remain. Additional testing and refinement of the user interface is possible. The system functions adequately but improvements are possible. There is enough work to support a second capstone project for next spring.

REFERENCES

1. Classroom Performance System (Version 3.40.0291) [Computer Software]. (1999-2005). e.Instruction Corporation, Denton, TX.
2. Teacher's Assistant Pro (Version 1.0) [Computer Software]. (2005). Pollack, Herff, Abderholden and Kaplan, Peoria, IL.
3. The Classroom Performance System. (2005). Retrieved March, 5, 2005, from http://www.eInstruction.com
4. tucows Downloads. (2005). Retrieved April 1, 2005, from http://tucows.com/preview/332433&ignore=1
5. USB Monitor 2.37 [Computer Software]. (2002-2003). HHD Software.