Computer Science Capstone Course Senior Projects: From Project Idea to Prototype Implementation

Sergiu M. Dascalu, Yaakov L. Varol, Frederick C. Harris, Brian T. Westphal
University of Nevada, Reno, Department of Computer Science and Engineering
Reno, NV USA 89557 {dascalus, varol, fredh, westphal}@cse.unr.edu

Abstract - **This paper gives an overview of a capstone course, CS426-Senior Projects, and focuses on the streamlined software engineering process followed to take student projects from initial concept to prototype implementation. The paper provides details on course structure, pointers to students' work, and observations on applying various educational tools. It also presents details on the software process designed and followed in CS426 to support high quality project development in the short timeframe of an academic semester. A discussion of criteria used for deciding on project phases, activities and artifacts, together with an analysis on how this shorter, streamlined development process could be applied to other engineering courses is included. The number of publications and other accomplishments that resulted from CS426 student projects are also reported. Furthermore, we compare our approach to this capstone course with others', and include a summary of planned enhancements to the course's content, delivery, and outcomes.**

Index Terms – Senior projects, Capstone course, Software engineering, Software process.

1. Background

CS426-Senior Projects is the second in a sequence of two senior level capstone courses taught in the Computer Science and Engineering Department of the University of Nevada, Reno [1]. The first course, CS425-Software Engineering, provides students with the necessary background for applying sound software engineering practices. In CS426, the emphasis is on team collaboration with the primary objective being the development of an original, functional, industry-strength software-intensive product [2]. This is deemed a coronation of the students' academic work, bringing to light a significant portion of their software development expertise acquired while in school.

Not only CS426-Senior Projects requires and reinforces material learned in previous courses, but it also involves significant new studying and learning. In CS426, the instructors present lectures on the Unified Process (UP) [3] and the Unified Modeling Language (UML) [4, 5] and its application to software analysis and design, and the student teams report on their projects' progress at the following stages: topic proposal (concept), software specification (requirements), design (model), and implemented software (prototype). A public Senior Projects Workshop with presentations, demos and posters concludes the course.

In December 2003, CS426-Senior Projects received the designation *major capstone course* from UNR's Core Curriculum Committee. During the review process leading to this designation, the criteria used for evaluating the course included [6]:

- Building upon core curriculum
- Integrative, broadly-focused, multi-disciplinary, and cross-cultural
- Focused on ethical and substantial issues and themes
- Promoting critical thinking
- Containing a rigorous writing component and a strong computational component
- Providing a unique opportunity for innovative teaching

CS426-Senior Projects is a required course in the Bachelor of Science in Computer Science program at UNR.

As part of the course overview, we provide in this paper details on course requirements and organization, refer to students' work, and comment on the experience gained by applying various pedagogical tools (e.g., invited talks by experts from the industry). We also present details (phases, activities, and artifacts) of the simplified (streamlined) software engineering process applied in CS426 to develop high quality projects within the short time period of a single academic semester. In addition, we indicate the criteria used for deciding on project phases, activities, and artifacts. A discussion on how this streamlined software development process could be applied to other engineering courses is also included. Furthermore, we give a brief report on publications and other achievements resulted from student projects, and compare our approach to others'. The paper also includes a set of directions of future work considered for enhancing the course's content, delivery, and outcomes.

The remainder of this paper is organized as follows: Section 2 presents details of course structure and requirements, Section 3 describes the streamlined software engineering process followed in CS426 for student project development, Section 4 focuses on specific educational strategies and tools used for delivering the course, Section 5 provides a brief comparison with similar work, Section 6 reports on results obtained so far and shows samples of students' work, Section 7 discusses the possibility of applying the suggested development process to other courses, Section 8 points to planned enhancements, and Section 9 presents our conclusions.

2. Course Structure And Requirements

A continuation of the CS425 Software Engineering course (taught in Fall), CS426 Senior Projects (taught in Spring), covers advanced topics on the UP and the UML. While the former provides the main software engineering process example for object-oriented development, the latter is its supporting notation and the widely accepted standard for object-oriented modeling and visualization. In addition, several lectures are dedicated to the human-computer interaction aspects of the projects developed, as the instructors firmly believe in user-centric approaches [7] for creating software-intensive products. For this purpose, short video clips of projects developed in well-known human-computer interaction laboratories such as those at IBM, HP, Stanford University, MIT, and the University of Maryland are also presented.

The teams of students have one academic semester (about 15 weeks) to develop their projects. Typically, the projects have a predominant software component (in Booch *et al.* terminology, they are "software intensive [4]). However, some hardware development has also been included lately and is encouraged (to encompass the challenging category of embedded systems). Throughout the semester the teams report on their work through documents (deliverables) submitted to the instructors and through project presentations delivered during regular classes. A project website is set up at the beginning of the work and updated regularly during the semester. An internal project demo ("dry run") is presented to the instructors during the last week of the semester. Few days later, at the public CS Senior Projects Workshop a final presentation is delivered, accompanied by a poster and, on-request, by a public demo. In Spring 2004 the CS Senior Projects Workshop took place on the 30th of April and enjoyed an audience of about 200 attendees throughout the day, from both inside and outside the university.

Each project team consists of either three or four students and has at least one "external advisor" and/or sponsor. External advisors for CS426 teams have been faculty members from UNR, as well as professionals from research organizations and industry. Several graduate students have also been involved in project advisement. Most of the external advisors have been from USA, but several experts from other countries (specifically Canada and Japan) have also been involved in advising the projects. All projects are guided by the course instructors, who supervise the entire project cycle, from topic selection to workshop demonstration. In both 2004 and 2005 there were 16 teams of students taking the Senior Projects course (55 and 53 students, respectively).

Project concept topics can be proposed by the students, by the instructors, and by the external advisors. The concepts are validated and the projects receive the "green light" (typically, in the third week of the course) only after brainstorming and analysis meetings of the teams with the instructors are held (these take place outside regular classes). The course encourages the application of computer technology to various areas of human activity including, for example, biochemistry, manufacturing, mining, military, home security, education, sports, and gaming [2, 8]. Examples of CS426 projects are presented in Section 7 of the paper.

Invited presentations by speakers from business, industry, and research are part of this course's structure. They are meant to provide insight in technical and business matters and foster the entrepreneurial initiative of the students. In both 2004 and 2005 three such talks were delivered during the course [2, 8].

Peer review, constructive critique, and active exchange of ideas are strongly encouraged in CS426. Based on class presentations, the students evaluate other teams' work. Teamwork, collaboration, peer feedback, instructor and external advisor guidance play essential roles in project development, refinement, and enhancement.

The grading scheme used currently in the Senior Projects course is shown in Table I. More details on the software development process designed specifically for this course, including the phases followed and the artifacts produced are presented next.

3. Streamlined Software Engineering Process

Because the pre-requisite course, Software Engineering, has a great deal of material to be covered, starting work on senior projects in this course is not an option we have followed (we look at possibly introducing it in some form in the future, though). Therefore, due to time constraints (15 weeks from start to finish), a simplified version of a traditional software development process has been used in our Senior Projects.

We call this process "streamlined," as it is made up of the most necessary development phases and activities. With the exception of the final phase, in which a complete product is delivered (that is, all its items are mandatory – to satisfy the top level project requirements), all other phases produce an "essential set" of artifacts (deliverables), which consists of a subset of mandatory items and a subset of sample (that is, incomplete) non-mandatory items. The phases followed and the artifacts produced are shown in Fig. 1.

As timeline for phases, we allocate roughly 3 weeks for deciding on project topics and writing the project concept, 3 weeks for specification, 3 for design, and 6 weeks for implementation, integration, and testing. For detailed activities within each phase we refer the reader to the project handouts posted at [2]. We mention only that such activities include setting main project goals, deciding on external advisors, identifying process resources (in the project concept phase), specifying and prioritizing requirements, modeling use cases, creating the project glossary (in specification phase), defining the high-level architecture, producing the medium and low-level design, and detailing the user-interface (in the design phase.)

Activities and artifacts require a significant amount of work: for example, a team of four students estimated about 700 hours work in total, over a time span of 15 weeks. Thus, we urge the reader not to associate "streamlined" with "easy" or "light".

TABLE I
GRADING SCHEME CS426-SENIOR PROJECTS, UNR, SPRING 2005

REQUIREMENTS	ITEM	WEIGHT [%]	CONTENTS
PROJECT DELIVERABLES	Concept	4	Description of project idea (maximum 1200 words).
	Specification	12	Software Requirements Specification document.
	Design	14	Design document with details of architectural design, detailed design, and user interface design.
	Implementation, Integration, and Testing	28	Documented program code with details of program structure, modules, interfaces, and test cases. Also, project demo.
PROJECT PRESENTATIONS AND PUBLICATIONS	Presentations	8	Three presentations: concept, design, and final product (the first two during regular classes, the third at the public Senior Project Workshop).
	Project Poster	6	Poster with project description, presented at the Workshop.
	Project Website	4	Regularly updated website showing project progress.
TEST	Midterm Exam	16	Test of UML and UP knowledge, as applied to individual projects (about 50% of the questions are project-related).
PARTICIPATION	Regular Classes & Workshop Participation	8	Participation to lectures and Workshop: Q&A, comments, suggestions, peer evaluation, attendance.

The first criterion applied, when deciding on phases and activities, is to include only the process steps without which a product cannot be completed. All software engineering processes include project concept, specification, design, implementation, integration, and testing [9]. Our approach maps roughly to the broader ("industrial") UP, which also consists of four phases: inception, elaboration, construction, and transition [3, 10]. Due to time constraints, we do not cover software evolution (software maintenance) [9].

Secondly, given the complex nature of the projects and the elements of software engineering risks that cannot be ignored (technical and organizational), we enforce throughout the project a prioritized treatment of requirements, on three levels. Level one translates to "must be implemented", level two to "might be implemented", and level three to "would like to implement, but most likely will not have time for it".

Thirdly, another criterion used is to emphasize a user-centric approach to the entire development. Thus, user interface prototyping is required in all phases of the project.

Finally, refinement is encouraged throughout the project. Each project document submitted to the instructors has an introductory part in which the teams relate to prior phases, re-state the goals of the project (if necessary), give a concise account on the progress made, and indicate modifications made to previous documents .

As a result, the process shown in Figure 1 has elements of both classical waterfall – the four phases that follow each other in strict sequence – and evolutionary development, as requirements and previous models can be (partially) changed as the project progresses and its priorities and technical solutions become clearer.

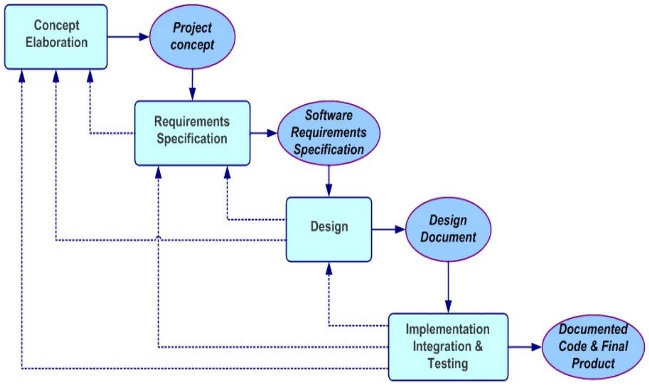

Concept Elaboration
Project concept
Requirements Specification
Software Requirements Specification
Design
Design Document
Implementation Integration & Testing
Documented Code & Final Product

FIGURE 1
PHASES AND DELIVERABLES IN CS-426 SENIOR PROJECTS, UNR

In terms of artifacts requested, the decision we made is to require completeness for those items deemed essential (functional and non-functional requirements, system architecture, and implementation of first level priority requirements) and enforce a minimum number of samples for the non-mandatory artifact items (e.g., scenarios, flowcharts, activity diagrams, and terms included in the project glossary). In other words, for project completion all mandatory artifacts must be delivered, while for gaining additional knowledge in UML modeling and software development a minimum number of "non-mandatory" artifacts is specified in each project phase. Worth noting, some teams have been very diligent in creating all types of artifact items, often exceeding by far the requirements we set for sample non-mandatory items.

4. EDUCATIONAL TOOLS

Because of its abbreviated timeline, the Senior Projects course must make use of the most efficient pedagogical strategies and tools available. Over the past several years the course has made use of both stable and experimental methodologies, evolving to better suit the needs of our students, department, and community.

From its inception, some concepts of the course's design were apparent. To simulate the commercial workplace, effective use of teamwork has been emphasized as the most important resource that students can employ. Teams are composed of three or four students, no more or less. This specific range allows for substantive diversity among group members (important for broad-topic projects that require an array of skills) while minimizing overhead for communication and schedule coordination. Students are given constructive feedback continuously throughout the academic semester. This feedback comes from a multitude of sources including peer comments on group presentations (three presentations per group in the semester: specification, design, and final product), and outside of class project meetings between the instructors, external advisors, and student teams. In addition to verbal and written feedback from peers and advisors, we have recently begun the video recording of student presentations. There are few more effective approaches for improving one's presentation abilities, than reviewing one's own presentations recorded on video.

Besides the technical abilities that students use in implementing their projects, students also gain relevant experience in business. The specification and design processes that the students undertake make them familiar with industrial product design processes. Their presentations benefit their communication skills and their peer reviews help students get used to receiving constructive feedback and also help them to formulate positive feedback of others' work. These endeavors are compounded through the opportunity for students to use their creativity. In the 2005 course, students have the option of using hi-fidelity audio/visual equipment to create advertisements or public information video clips for their projects. For 2006, this will become a requirement as we have found that it adds another dimension to project development – how does one "sell" the product to the customer?

CS426-Senior Projects is a course highly anticipated by students, a course in which they combine the knowledge they have gathered throughout their college experiences. The course is supported by the Northern Nevada community as a whole including funding from the Department of Computer Science and Engineering, the College of Engineering, the University of Nevada, Reno, and local and regional industry. In addition to funding from these sources, industrial partners provide students with external advisors that give a commercial and industrial perspective on projects. Members of industry are also commonly asked to speak as guest lecturers for the course. In the past, speakers have addressed issues such as open source business models and developing commercially successful games, among other things. A tradition that began with the 2004 course, alumni have stepped up to offer prizes that help further motivate students in the Senior Projects course to excel. Monetary prizes are currently given for best in-class presentation and best overall products. We feel that these prizes are a good and fun way to additionally motivate the students with a healthy dose of competition, and we are pleased that the list of supporting alumni is growing each year.

5. COMPARISON WITH SIMILAR CAPSTONE COURSES

Capstone courses at other universities throughout the United States have similar goals to those of the CS-426 Senior Projects course at UNR (more details on the goals of CS426 can be found in the course syllabus [2]). The implementation specifics of our course, however, are different from others'. In particular, the organization and responsibilities of our student teams emphasize independent team entities. In addition, our teams have a great deal of flexibility in choosing and implementing their topics.

At other universities it is common for capstone courses to focus on particular domains, such as game development [11]. Students in Senior Projects at UNR are arranged as teams that work independently (of other teams) on individualized projects. Student teams are allowed to choose their own topics or may be given ideas for projects from advisors (subject to course instructors' approval).

In some other capstone courses, the entire class works on a larger single project [12]. In such a case, students interested in specific aspects of development are assigned to various tasks. While this might enable one to produce a larger product, it may not give undergraduate students the breadth-experience that they are expected to have upon graduating.

Like similar capstone courses, ours is designed to provide a simulated real-world experience based on industrial techniques and processes. Besides industry-oriented projects, students also have the option, if they are so inclined, to approach research-oriented topics. Integrating a research and analysis component is important for students that are interested in continuing onto advanced degrees.

The course at [13] provides students a similar opportunity, but in a limited scope as students work on specific projects rather than creating unique research and development projects such as some of ours [14, 15, 16, 17]. In general, students will find that they have more freedom to explore both a depth (their chosen topic) and a breadth (the overall development process) of experiences, than in other universities' capstone courses.

6. DISCUSSION OF PROCESS APPLICABILITY TO OTHER ENGINEERING COURSES

The Senior Projects in Computer Science course exposes students to a specification, design, and implementation process that can be applied to projects in most engineering disciplines. The process, which is stripped down to the essentials, emphasizes rapid prototyping and user-centric design where possibilities for periodic refinement make up for a lack of time spent on each stage (concept, specification, design, and implementation). The 15-week course gives students the hands-on experience they need to walk through the same process with virtually any other project they will complete under such a limited timeframe.

If one were to remove the term “software” from the description in Section 3, one would find that the engineering process could be applied in other fields. This model is especially applicable for concept projects (such as many student projects) that do not involve critical levels of safety or security. As one gets further into the process, the prototype detail increases but the focus is continuously aimed at satisfying the needs of the end user. User interface design is one of the most important tasks at each stage as it helps clarify the organization of the implementation and represents what the final product will look like.

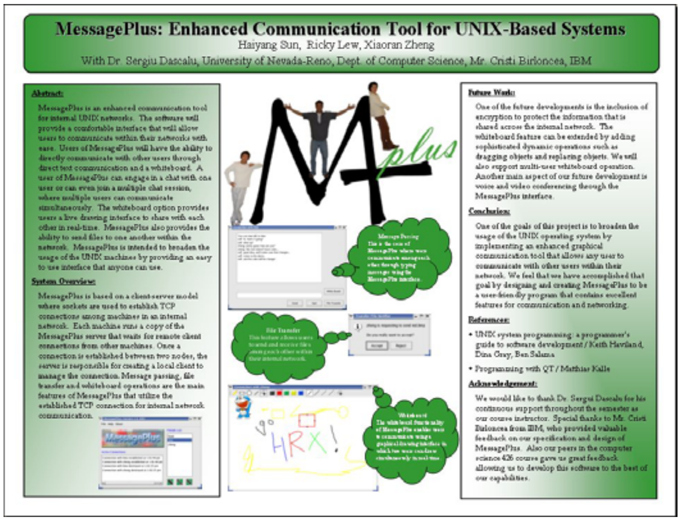
As an example, in the field of electrical engineering one might apply a similar process to the one used in our Senior Projects course to design a circuit. The “user interface” in such a case is the parts of the circuit that take input from or provide feedback to the “user”. The user may be an actual human user in some cases, or it may be another electronic system. Prototypes can be block diagrams, computer simulated circuits, or partially implemented circuits. Because of time constraints in this process, one can only verify an implementation at a high-level. Even though this is ideal only when working on non-critical applications that need rapid, but high quality prototypes, most student projects (and many commercial concept projects) fit in this category. For more critical applications, rigorous prototype verification should be completed at each phase.

7. RESULTS AND SAMPLE STUDENT WORK

This course is about software engineering and project development and the results in this area have been very good. Figures 2 and 3 show samples of posters (actual size, 3 x 4 feet) prepared by Senior Projects groups for public presentations (more samples, as well as complete project descriptions, can be found at [2] and [8]). An exciting byproduct of CS426 has been the unexpected research thrust that this course has created. This thrust which has been fostered by student interest and enthusiasm has led in 2003 and 2004 to 17 refereed publications in 8 international forums, with 46 out of the 90 Senior Projects students among the co-authors. We expect this trend to continue as we already have several drafts this year being assembled.

These papers have been published in a variety of places such as the Springer-Verlag series Lecture Notes in Computer Science [15], obtained two best paper of the session awards at other conferences [18, 19], and several were in a special session at SERP-2004 (IMCSCE-2004) that “was ranked third (out of 98 [sessions] for the entire event)” [20].

These projects have offered students a taste of requirements engineering, design, implementation, and research work and have led several of them to continue their education in computer science. Some of these students have chosen to take their group project to the next level of development while pursuing their MS degrees and we have seen several great enhancements, some of which have led to more publications, such as [21].



MessagePlus: Enhanced Communication Tool for UNIX-Based Systems

Haiyang Sun, Ricky Lew, Xiaoran Zheng

With Dr. Sergiu Dascalu, University of Nevada-Reno, Dept. of Computer Science, Mr. Cristi Birloncea, IBM

Abstract:

MessagePlus is an enhanced communication tool for internal UNIX networks. The software will provide a comfortable interface that will allow users to communicate within their networks with ease. Users of MessagePlus will have the ability to directly communicate with other users through direct text communication and a whiteboard. A user of MessagePlus can engage in a chat with one user or can even join a multiple chat session, where multiple users can communicate simultaneously. The whiteboard option provides users a live drawing interface to share with each other in real-time. MessagePlus also provides the ability to send files to one another within the network. MessagePlus is intended to broaden the usage of the UNIX machines by providing an easy to use interface that anyone can use.

System Overview:

MessagePlus is based on a client-server model where sockets are used to establish TCP connections among machines in an internal network. Each machine runs a copy of the MessagePlus server that waits for remote client connections from other machines. Once a connection is established between two nodes, the server is responsible for creating a local client to manage the connection. Message passing, file transfer and whiteboard operations are the main features of MessagePlus that utilize the established TCP connection for internal network communication.

Message Passing
This is the core of MessagePlus where users communicate among each other through typing messages using the MessagePlus interface.

File Transfer
This feature allows users to send and receive files among each other within their internal network.

Whiteboard
The whiteboard functionality of MessagePlus enables users to communicate using a graphical drawing interface in which two users can draw simultaneously in real-time.

Future Work:

One of the future developments is the inclusion of encryption to protect the information that is shared across the internal network. The whiteboard feature can be extended by adding sophisticated dynamic operations such as dragging objects and replacing objects. We will also support multi-user whiteboard operation. Another main aspect of our future development is voice and video conferencing through the MessagePlus interface.

Conclusion:

One of the goals of this project is to broaden the usage of the UNIX operating system by implementing an enhanced graphical communication tool that allows any user to communicate with other users within their network. We feel that we have accomplished that goal by designing and creating MessagePlus to be a user-friendly program that contains excellent features for communication and networking.

References:

- UNIX system programming: a programmer's guide to software development / Keith Haviland, Dina Gray, Ben Salama
- Programming with QT / Matthias Kalle

Acknowledgement:

We would like to thank Dr. Sergui Dascalu for his continuous support throughout the semester as our course instructor. Special thanks to Mr. Cristi Birloncea from IBM, who provided valuable feedback on our specification and design of MessagePlus. Also our peers in the computer science 426 course gave us great feedback allowing us to develop this software to the best of our capabilities.

FIGURE 2
SAMPLE POSTER (2004): MESSAGE PLUS

There are several things we have been surprised about in this course. First is the fact that the students do not want to work on multi-year on-going projects. They are much more interested in selecting their own project and seeing it all the way to conclusion. Second is the fact that we have not been able to generate as many business connections and initiatives as we have desired. Third, as several of the students have expressed interest in commercializing their projects we wish that there was more room for a business plan development. The first and third may lead to some changes in the structure of the first semester course (CS425-Software Engineering), perhaps allowing us to spread the topics of the two courses over the two semesters and thereby giving more time for each phase of project development.

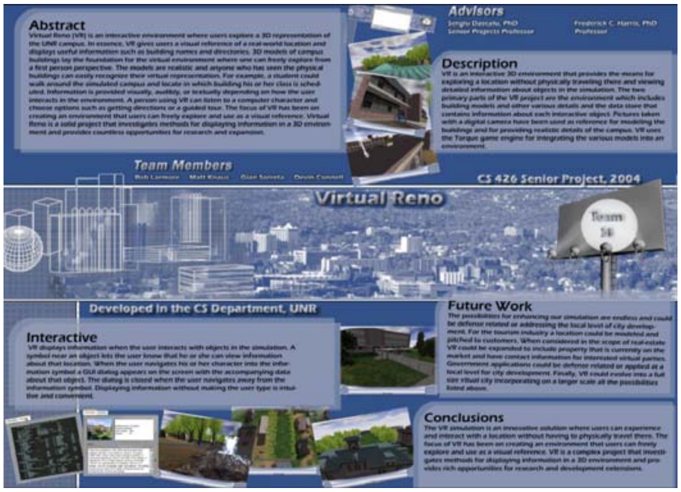
8. PLANNED ENHANCEMENTS

The capstone course discussed in this paper, Senior Projects in Computer Science, is a very demanding, challenging, and rewarding course for the instructors. This is due to the many application domains the projects cover (e.g., biochemistry, physics, atmospheric science, forestry, mining, network communications, and many more), the tight schedule followed, and the complexity of the work that is evaluated.

From the students’ perspective, this course is also very demanding. However, it has been exciting to see that the students are working very hard and show great enthusiasm in their work. We believe that this is due to the very practical nature of the course, as well as the fact that the projects were selected by the student and not thrust upon them.

We see enhancement of this course coming in several different directions. First, we see an increasing need for more feedback to the students. This can be accomplished through questionnaires and evaluations filled out by industry specialists and software engineering professionals, as well as by potential users of the products being developed.

Second, we see the need for video recording of the presentations given by groups (we have started this recently).



Abstract

Virtual Reno (VR) is an interactive environment where users explore a 3D representation of the UNR campus. In essence, VR gives users a visual reference of a real-world location and displays useful information such as building names and directories. 3D models of campus buildings lay the foundation for the virtual environment where one can freely explore from a first person perspective. The models are realistic and anyone who has seen the physical buildings can easily recognize their virtual representation. For example, a student could walk around the simulated campus and locate in which building his or her class is scheduled. Information is provided visually, audibly, or textually depending on how the user interacts in the environment. A person using VR can listen to a computer character and choose options such as getting directions or a guided tour. The focus of VR has been on creating an environment that users can freely explore and use as a visual reference. Virtual Reno is a solid project that investigates methods for displaying information in a 3D environment and provides countless opportunities for research and expansion.

Team Members

Rob Larmore Matt Knaus Gian Sorreta Devin Connell

Advisors

Sergiu Dascalu, PhD
Senior Projects Professor

Frederick C. Harris, PhD
Professor

Description

VR is an interactive 3D environment that provides the means for exploring a location without physically traveling there and viewing detailed information about objects in the simulation. The two primary parts of the VR project are the environment which includes building models and other various details and the data store that contains information about each interactive object. Pictures taken with a digital camera have been used as reference for modeling the buildings and for providing realistic details of the campus. VR uses the Torque game engine for integrating the various models into an environment.

CS 426 Senior Project, 2004

Virtual Reno

Developed in the CS Department, UNR

Interactive

VR displays information when the user interacts with objects in the simulation. A symbol near an object lets the user know that he or she can view information about that location. When the user navigates his or her character into the information symbol a GUI dialog appears on the screen with the accompanying data about that object. The dialog is closed when the user navigates away from the information symbol. Displaying information without making the user type is intuitive and convenient.

Future Work

The possibilities for enhancing our simulation are endless and could be defense related or addressing the local level of city development. For the tourism industry a location could be modeled and pitched to customers. When considered in the scope of real-estate VR could be expanded to include property that is currently on the market and have contact information for interested virtual parties. Government applications could be defense related or applied at a local level for city development. Finally, VR could evolve into a full size visual city incorporating on a larger scale all the possibilities listed above.

Conclusions

The VR simulation is an innovative solution where users can experience and interact with a location without having to physically travel there. The focus of VR has been on creating an environment that users can freely explore and use as a visual reference. VR is a complex project that investigates methods for displaying information in a 3D environment and provides rich opportunities for research and development extensions.

FIGURE 3
SAMPLE POSTER (2004): VIRTUAL RENO

This will allow the students to do a self-critique of their presentations and will also allow us to implement the first item in a way more convenient to the outside evaluators (who might find it difficult to attend class project presentations).

Third, we need an increased participation from companies and research groups outside of the department. It was interesting last year to see the excited responses to real projects that the students could see the practicality of.

Fourth, we seek an increase in industry support as well as sponsorship of initial project budgets for supplies needed by the student groups to start their projects.

Fifth, we would like to increase the student's motivation through the sponsorship of larger prizes for best projects and best presentations. We have already begun pursuing this direction, but we would like to see it substantially enhanced.

9. CONCLUSIONS

This paper has presented details of the UNR capstone course Senior Projects in Computer Science. This is an exciting course, for which we have seen many of the department's faculty take ownership through their work with student groups. This course has helped lay the foundation for a proper software engineering process that is systematic and precise, albeit simplified (streamlined).

Not only this course can significantly enhance the students' experience in terms of software development but it can also immerse them into certain research activities, thus preparing them for challenging and innovative work on complex real-word applications. The practicality of this course has been borne out in a variety of fashions, ranging from the visits by startup capital investors to Workshop presentations, to the fact that over the past two years we have seen more than 15 refereed papers emerge from the student groups.

ACKNOWLEDGMENT

The Senior Projects course has received funding from several sources. A grant from the UNR Student Technology Fee and another from the Office of the Provost has helped construct a Software Engineering Lab that students in Senior Projects use to work on their projects. At UNR, the College of Engineering and the Department of Computer Science and Engineering have consistently provided funding for equipment and software, as well as for sponsoring public events. We would like to express our gratitude for their generosity.

We would also like to extend our thanks to the students who, as part of their coursework, prepared the posters shown in this paper: Xiaoran Zheng, Ricky Lew, Haiyang Sun (Message Plus), Robert Larmore, Matthew Knaus, Gianpaulo Sorreta, and Devin Connell (Virtual Reno).

REFERENCES

[1] Department of Computer Science and Engineering, University of Nevada, Reno (UNR) website: http://www.cse.unr.edu (March 2005).

[2] UNR CS 426 Senior Projects 2005 website: www.cs.unr.edu/~ sp2005

[3] Jacobson, I., Booch, G., and Rumbaugh, J., *The Unified Software Development Process*, Addison-Wesley, 1998.

[4] Booch, G., Rumbaugh, J., and Jacobson, I., *The Unified Modeling Language: User Guide*, Addison-Wesley, 1998.

[5] OMG's UML Resource Page: http://www.omg.org/uml/ (March 2005)

[6] UNR Core Curriculum: www.unr.edu/core-curriculum/ (March 2005)

[7] Norman, D., *Emotional Design*, Basic Books, 2004.

[8] Pointers to UNR CS Senior Projects developed between 2002 to 2004: http://www.cs.unr.edu/~dascalus/sp.html (March 2005).

[9] Sommerville, I., *Software Engineering*, 6th Ed., Addison-Wesley, 2000.

[10] Arlow, J., and Neustadt, I., *UML and the Unified Process: Practical Object-Oriented Analysis and Design*, Addison-Wesley, 2002.

[11] Parberry, I, Roden, T., and Kazemzadeh, M.B. "Experience with An Industry-Driven Capstone Course on Gaming", *Proceedings of ACM SIGCSE'05*, pp. 91-95.

[12] Way, T. P. "A Company-based Framework for a Software Engineering Course", *Proceedings of ACM SIGCSE'05*, pp. 132-136.

[13] Pastel, R., "Integrating Science and Research in a HCI Design Course", *Proceedings of ACM SIGCSE'05*, pp. 31-35.

[14] Dascalu, S., Pasculescu, A., Woolever, J., Fritzinger, E., and Sharan, V., "Stratified Programming Integrated Development Environment", *Procs. of the 12th Intl. Conf. on Intelligent and Adaptive Systems and Software Engineering (IASSE-2003)*, July 2003, San Francisco, CA, pp. 227-232.

[15] Kallman, J., Minaie, P., Truppi, J., Dascalu, S., and Harris, F., "Software Modeling for Open Distributed Network Monitoring Systems", *Lecture Notes in Computer Sci.*, LNCS-3126, Springer-Verlag, 2004, 158-169.

[16] Dascalu, S., Harris, F., Knaus, M., Larmore, R., Sorreta, G., and Connell, D., "Virtual UNR Campus: The Specification Process", *Procs. of the Intl. Conference on Software Engineering Research and Practice (SERP-2004)*, June 2004, Las Vegas, NV, vol. 1, pp. 290-296.

[17] Montulli, L., Collier, D., Gilliland, S., and Reeves, J., "Specification of the Blaze-A-Trail Interactive Trail Mapping System", *Procs. of the Intl. Conf. SERP-2004*, June 2004, Las Vegas, NV, vol. I., pp. 255-261.

[18] McDole, T., Cua, H., Huang, C., Kania, L., Dascalu, S., and Harris, F.C., "Software Specification of the GORT Environment for 3D Modeling", *Procs. of the 7th World Multiconference on Systemics, Cybernetics & Informatics (SCI-2003)*, July 2003, Orlando, FL, vol. VI, pp. 254-259.

[19] Minaie, P., Kallman, J., Truppi, J., Dascalu, S., and Harris, F., "Practical Educational Uses of the Open Distributed Network Monitor", *Procs. of the Intl. Conf. on Education and Information Systems, Technologies and Applications (EISTA-2004)*, July 2004, Orlando, FL, vol. I, pp. 98-103.

[20] Arabnia, H. (General Chair of the International Las Vegas Multi-Conference on CS & CE), email communication, February 2005.

[21] Larmore, R., Knaus, M., Dascalu, S., and Harris, F.C., "Virtual Environment for On-Campus Orientation", *Procs. of IEEE Symposium on Collaborative Techn. and Systems (CTS-2005)*, St. Louis, May 2005.