Response to Microsoft Research Tablet PC and Computing Curriculum Request for Proposal, November 2004

Name: Roy P. Pargas

Title: Associate Professor of Computer Science

University: Clemson University

Email: pargas@cs.clemson.edu

Phone 864-656-5855

Address: 204 McAdams Hall, Department of Computer Science, Clemson University
Clemson, SC 29634

Proposal Title: Development of a Targeted Tablet PC Software Development Course

1. Introduction

New technologies can excite the imagination. This is true not just with a finished product but equally so during the *development* of a product. We propose to capture and infuse students with this excitement through a new course entitled *CPSC 481/681: Tablet PC Software Development*, designed for senior undergraduate / first-year graduate computer science students and offered by the Clemson University Computer Science Department.

This course achieves two interrelated goals. (1) The course teaches both undergraduate and graduate students the skills necessary to develop exciting and useful Tablet PC software. (2) The software developed will provide *new* instructional material targeting an existing laptop-enhanced course being taught at Clemson University. In a laptop-enhanced course, each student comes to class with a laptop equipped with wireless access to the Internet. These laptop-enhanced courses are being taught by instructors (from many different disciplines) who are experimenting with novel pedagogical techniques in an attempt to use technology, i.e., laptops, to deliver course content more effectively. The Tablet PC software produced by *CPSC 481/681* students will supplement the material currently being used in these courses.

Section 2 details the proposed course and shows how the two goals are achieved. Section 3 describes the administration of the proposed course, including assessment, dissemination of information, student outcomes, a schedule of tasks, and deliverables expected from this course. Section 4 gives a budget and justification of costs. Section 5 provides a brief summary.

2. Proposed Course

2.1. Overview

CPSC 481/681: Tablet PC Software Development will be taught as a 3 credit hour computer science *elective* to senior undergraduate / first-year graduate computer science students. Students who enroll are advanced computer science students who have successfully completed at least six computer science courses, including four programming courses. Because they are advanced students, they will be able to handle the relatively rapid pace and high-level instruction envisioned for this course.

A typical fifteen-week semester is divided into three phases. In Phase 1 (weeks 1-6), the instructor lectures on topics such as: (a) Visual Studio.NET, (b) C#, and (c) APIs specifically designed for Tablet PC software development. Students are given 5-6 small (one-week) programming assignments, each focusing on one or more of the concepts from the topics above. The objective is to very quickly provide students with as many C# programming examples involving Tablet PC APIs using Visual Studio.NET as possible.

In Phase 2 (weeks 7-12), the students select, design, and implement a large project. Students (a) pick from a list of project topics provided by the instructor, (b) propose a design of a solution, (c) develop a list of software development tasks with completion dates, and (d) implement solutions to each of the tasks according to the schedule.. Students meet with the instructor at least once a week, oftentimes more frequently, to discuss ideas and to provide evidence of progress. The instructor monitors each student's progress and, after each meeting, specifies what the students are expected to have accomplished by the next meeting. Students may work individually or in teams of two or three with the understanding that team projects are expected to be proportionally more difficult than individual projects.

This phase is a *mentoring* or *coaching* period in which the instructor spends a *large* amount of one-on-one time with each student, carefully guiding each student's progress. It is also during this time that the class, as a whole is expected to make significant progress, as explained in Section 2.3 below. In the final Phase 2 meeting with the instructor at the end of week 12, the students are expected to demonstrate at minimum a *working draft* of the complete project.

Phase 3 takes place during weeks 13-15 of the semester. A *mini-conference* is organized by the instructor. Over one or two days (depending on the number of projects) students give a 20-minute presentation of their work including a 10-minute overview, a 5-minute demonstration of their software, and

a 5-minute question and answer period. Students are asked to dress in Sunday attire as if they were giving a formal presentation at a conference. The instructor acts as session chair, friends are encouraged to attend, and every effort is made to give the students the experience of presenting a paper at a professional meeting.

All students are expected to *attend* every presentation. Moreover, students in the audience evaluate each presenter's performance. The instructor reviews the evaluation forms and then gives them to the presenter thus providing valuable feedback to the student about his or her performance.

The final requirement of Phase 3 is written documentation: (a) a 5- to 10-page Technical Reference Manual, and (b) a 5- to 10-page User's Manual. Students are instructed to include in the Technical Reference manual information about their projects that a future student will need to extend and build on their work. This information should minimally contain: (a) a hierarchy of C# classes used, and (b) a description of the major data structures used in the system. Students are told that the User's Manual should guide the user through the program and should, therefore, contain screenshots depicting the various functions offered by the software system.

Students are evaluated in the following way:

1.	(Phase 1, 6 wks)	20% Five to seven small programming assignments
2.	(Phase 2, 6 wks)	10% Quality of project proposal; 10% Progress during Phase 2 meetings
3.	(Phase 3, 3 wks)	10% Presentation; 5% Peer-evaluation of presentation
		25% Completely working project; 20% Documentation

There are *no* tests or quizzes in this course, only programming assignments and project-related activities.

2.3. Promoting an Environment of Cooperation

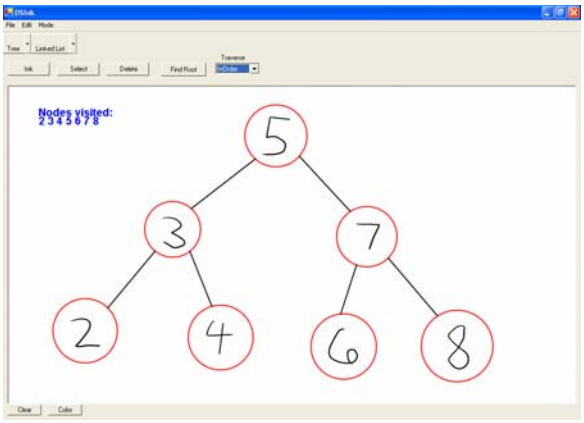
Unlike other programming courses in which students are prohibited from collaborating or from using code written by others, in this course students are *encouraged* to share with their peers what they have learned about Tablet PC programming. This is accomplished, during Phase 2, by each student maintaining a web-page detailing his or her progress. When a student achieves a programming breakthrough, the instructor will ask the student to "publish" his or her result on the web in order that others in the class can learn and use the programming technique. Students who share their work are rewarded with bonus credits. Our hypothesis is that encouraging students to share programming successes eliminates the feeling of competition in the classroom and allows *all* students to move their projects forward collectively. It

cultivates a spirit of cooperation that spurs students to work even harder on their projects. Students are eager to utilize techniques that others may have discovered in order to improve their own projects. And because the projects are *all different*, there is little danger that a student can simply coast. This approach has, in fact, been tried successfully in another course taught by the author [1].

2.4. Project Themes and Laptop-Enhanced Courses

Each semester, the instructor defines the set of software projects from which students select during Phase 2. The projects share a common theme and collectively target and support a course already being taught at Clemson. As an example, the first time this course is taught the projects will be Tablet PC software that supports a standard computer science data structures (CS4) course. That is, CPSC 481/681 students will develop Tablet PC software that allows the CS4 user to draw linked lists, stacks, trees, and graphs. Associated algorithms will demonstrate tree traversal, insertion into and deletion from binary search trees, AVL trees, splay trees, B-trees, binary heaps, leftist and skew heaps, Dijkstra's algorithm for shortest paths, Prim and Kruskal's algorithm for minimum spanning trees, depth- and breadth-first graph searches, message encoding and decoding using Huffman trees, etc.

Work along these lines has already begun [2]. Current progress includes Tablet PC programs for drawing linked lists and trees and algorithms demonstrating inorder, preorder, and postorder tree traversals. A user, for example, can draw an arbitrary binary tree, label the nodes with numeric or alphabetic values, click a button, and observe the nodes of the tree being visited in inorder (the diagram shows the final state after an animated Tablet PC program demonstrates the order in which nodes are visited). The plan is to use examples (such as that shown in the diagram) during Phase 1 to demonstrate how Tablet PC programs are developed. After learning the basics, students will develop their own Tablet PC programs for the data structures and algorithms they selected from the project list.



The result, at the end of the semester, is a collection of Tablet PC algorithms that demonstrate a variety of data structures and their behaviors. The benefit to a CS4 instructor is substantial. He or she can, for

example, use the Tablet PC algorithms to demonstrate how to insert nodes into an initially empty AVL tree and show how the tree self-balances. Or the instructor can draw a graph and compare and contrast Prim's and Kruskal's algorithms for finding minimum spanning trees. The simpler algorithms, such as tree traversal, may also be used in CS1 and CS2.

The set of algorithms is particularly useful to the instructor of a laptop-enhanced class. The students of a laptop-enhanced CS4 class can download the Tablet PC algorithm corresponding to the data structure of the day and can use the algorithm in a structured class activity designed by the instructor. Students can be actively learning about *heapsort*, for example, rather than passively listening to the instructor lecture on it.

After the semester ends, the *best* student will be invited, either for pay or as a senior thesis project, to collect all of the projects and organize all source code and documentation into a single website. Part of this task will also be to complete any of the projects that were either unsuccessful or not attempted. The course website will be a future reference for anyone (CS1, CS2, CS4 and CPSC 481/681 instructors and students) who is interested in working with the Tablet PC programs. The URL will be provided to Microsoft to include in any database or webpage the company chooses.

2.5. Subsequent semesters

The best part about the course webpage produced is that it becomes a *foundation* for the second time CPSC 481/681 is taught. *All* of the code from the first class is now available to the instructor and students of the second class. The second time CPSC 481/681 is taught, the target course will be *Computer Organization (CS3)* and projects will include, for example, Tablet PC programs that allow *CS3* students to build complex devices from simple components (such as programs that enable a *CS3* student to build *N-bit* incrementers, decrementers, and adders by correctly coupling *N* 1-bit adders). The point, though, is that with each passing semester, the body of Tablet PC software and knowledge grows. And because all code and documentation is provided online, that knowledge is passed on and is available to subsequent classes.

3. Course Administration

3.1. Schedule of Tasks

The following schedule describes the tasks to be accomplished as part of this project:

- Design course, develop Phase 1 exercises and Phase 2 project descriptions (summer 2005)
- Teach CPSC 481/681 first time, collect formative and summative assessments (fall 2005)

- Collect all Tablet PC algorithms, organize course website, write paper(s) for computer education conference(s) (spring/summer 2006)

3.2 Formative and Summative Assessments

At the end of each of Phases 1, 2, and 3, *formative* assessment of the course will be taken. Students will be asked to provide anonymous evaluation of the course along four general lines:

1. Does the course live up to its billing? Are you learning what you expected to be learning?
2. What are the strengths of the course? Which topics do you feel are of greatest interest?
3. What are the weaknesses of the course? Which topics do you feel are of least interest?
4. What suggestions do you have to improve the course?

At the end of the semester, the students will be asked to provide a final *summative* assessment of the course. The questionnaires to be used have not been finalized. Specific questions will be designed before the start of the Fall 2005 semester.

3.3. Dissemination of Information

Dissemination of information will be accomplished through: (a) the development of a course website containing all code and documentation produced in the course (see [1] for an example of such a website developed by the author for another course), (b) the use of the Tablet PC software developed in fall 2005 in the data structures course to be taught by the author in spring 2006 (the author has regularly taught the data structures course in the Computer Science Department at Clemson University for the past ten years), (c) at least one paper describing this project to be submitted to a computer education conference such as ACM SIGCSE or ASEE/IEEE CS FIE, and (d) two reports per year to be submitted to Microsoft during 2005 and 2006 summarizing the successes and any failures of the course, including a summary of the formative and summative assessments. In addition, an integral part of this proposal is a commitment by the author to attend a Tablet PC Computing Curriculum Workshop and present all knowledge learned in this project.

3.4. Student Outcomes

The CPSC 481/681 student can expect to learn several important skills: (a) Tablet PC programming using C# and Visual Studio.NET, (b) project management and development (possibly working as a member of a team), (c) formal presentation and communication, and (d) document development.

3.5. Qualifications of the Instructor

The instructor of the proposed course and author of this proposal is an Associate Professor of Computer Science at Clemson University and is a recipient of a 2003 Microsoft Curriculum and Content Development Grant (Project title: Development of a Network Applications Course Using Microsoft .NET, SQL Server, and IIS; Tom Healy, Content and Curriculum Manager, Microsoft Research University Relations). His publications include recent papers on using and developing tools to support laptop-enhanced courses [3-6]. He received a 2003 award for innovative teaching [7]. He is currently working on the development of a web-based program called *MessageGrid* [8] which facilitates instructor-student interaction in laptop-enhanced courses. Although useful in any laptop-enhanced class, *MessageGrid* is particularly useful in a *large* class because it provides the ability for the instructor to simultaneously interact through the web with many more students during the class period than otherwise possible. A preliminary version of *MessageGrid* is currently being used in the fall 2004 semester by six faculty members from widely differing disciplines (Music Appreciation, Spanish, Nursing, Psychology, English, and Computer Science).

4. Budget and Budget Notes

5. Summary

This document proposes to develop a new course entitled *CPSC 481/681: Tablet PC Software Development*, designed for senior undergraduate / first-year graduate computer science students and offered by the Clemson University Computer Science Department. This course achieves two interrelated goals. (1) The course teaches both undergraduate and graduate students the skills necessary to develop exciting and useful Tablet PC software. (2) The software developed will provide *new* instructional material targeting an existing laptop-enhanced course being taught at Clemson University.

References

1. CPSC 463/663 website, Department of Computer Science, Clemson University, Clemson, SC 29634 http://www.cs.clemson.edu/~pargas/463web/
2. Culler, Robert., *On Developing Tablet PC Software*, M.S. paper, Department of Computer Science, Clemson University, (*in preparation*)

3. W.F. Moss, R.P.Pargas, L.W. Grimes, B. Weaver, 2003, Technology + Innovation = Pedgogy. *Proceedings of the International Conference on Multimedia and Information and Communication Technologies in Education*, Badajoz, Spain, pp. 1065-1069.
4. A.B. Campbell and R.P. Pargas, 2003, Laptops in the Classroom, *Proceedings of the ACM 34th SIGCSE Technical Symposium on Computer Science Education*, Reno, Nevada, pp. 98-102.
5. J. Herron and R.P. Pargas, 2004, Blending Technology into Review Sessions, *Proceedings of the 15th International Conference on College Teaching and Learning*, Jacksonville, FL, pp. 47-55.
6. R.P. Pargas, 2004, A Different Pedagogical Approach to Teaching a Web Applications Course, *Proceedings of the 2004 IADIS International Conference, WWW/Internet 2004*, Madrid Spain, v. 1, pp. 193-200.
7. 2003 Award for Innovative Excellence in Teaching, Learning, and Technology, given by the Center for Advancement of Teaching and Learning, Jacksonville, FL.
8. R.P. Pargas, A.R. Levin, and J.D. Austin, Every Voice Heard: Serving Class Participation Needs, *(in preparation)*