CPI: CS Undergraduate Curriculum Enhancement Through Multi-Year Coordinated Project *

Kurt J. Maly, Dennis E. Ray, J. Christian Wild, Irwin B. Levinstein,
Stephan Olariu, C. Michael Overstreet, Deane Sibol and George Panayides
Department of Computer Science
Old Dominion University
Norfolk, VA 23529-0162

Nageswara S. V. Rao
Center for Engineering Systems Advanced Research
Oak Ridge National Laboratory
Oak Ridge, TN 37831-6364

August 26, 1994

Address for correspondence:
Prof. Kurt J. Maly
Department of Computer Science
Old Dominion University
Norfolk, VA 23529-0162
phone: (804) 683-3915
email: cpi@cs.odu.edu

*This work is funded the by National Science Foundation under grant #CDA-9214930 and Old Dominion University.

CPI: CS Undergraduate Curriculum Enhancement Through Multi-Year Coordinated Project

Abstract

Over the last three decades computer science has evolved into a mature discipline with a well-defined curriculum. Recently many have come to realize that computer science, as a discipline, must transcend its own confines of subject area. Most computer science curricula teach principles and programming skills in isolation from an application perspective, provide limited laboratory experience, and introduce inadequate integration with non-CS components. The Computer Productivity Initiative (CPI) proposes to alleviate these problems by integrating into the curriculum a multi-year project that includes hardware and software issues and also addresses engineering, business, and other non-CS topics. The students apply principles of productivity and employ leading-edge technologies both in the process of project development and its execution. In the process, they also hone essential career-oriented skills in the areas of management, formal presentations, and group problem solving. This paper is a progress report on the implementation issues and the integration into our curriculum. It also describes the methods for the evaluation of the project and for the dissemination of the products that could gracefully integrate CPI into existing curricula at other institutions.

CR Categories: *K. Computing Milieux*: K.3 Computers and Education; K.4 Computers and Society

1 Introduction

The Computer Productivity Initiative (CPI) is a result of a concerted effort to enhance the traditional computer science undergraduate curriculum to better prepare the graduates to meet the challenges of the careers awaiting them and thereby make more positive societal and economic contributions. The reports by the ACM/IEEE Joint Curriculum Task Force [2] and NSF Computer Science Education Workshop Report (Foley *et al.* [6]), British Computer Society and Institution of Electrical Engineers [1], and articles by Denning [7], Parnas [12], Shaw [13] and Wulf [16] discuss several aspects of computer science education. The issues of what higher education institutions should teach the new computer scientist are far from being settled and are currently being pursued in several studies (Parnas [12], Shaw [13]). However there seems to be a general sense of agreement for making changes in the current curriculum to meet the needs of year 2000 and beyond (see the report of the Commonwealth of Virginia's Commission on The University of the 21st Century [3]). A recent study of our graduates, who have been practicing professionals for one to five years, reveals that they went through several months of company training programs before taking on active company responsibilities. Discussions with industry as well as government representatives, including those on our advisory board, have revealed similar conclusions. Fresh graduates are often found to be functionally incapable of productive work without significant training beyond their university education. Interestingly enough, the same graduates are perceived by the employers as having had a sound education and holding a great promise of significant productive efforts.

In this paper we describe the CPI effort that is currently in progress under the joint support of National Science Foundation and Old Dominion University. The present curriculum is enhanced by adding a multi-year integrated project: two additional courses are added and two existing courses are adapted. This initiative is currently entering in its third year of operation with culminating efforts aimed at producing a package that enables other universities to incorporate CPI into their curricula. The process of incorporating CPI into our curriculum is riddled with logistic and organizational challenges in addition to the academic concerns. These issues are also briefly outlined here.

The remainder of this paper is organized as follows. In Section 2 we identify some of the shortcomings in our computer science undergraduate program. A brief outline of our approach to incorporate a multi-year project into the computer science curriculum is described in Section 3. Our plan and its execution is based on a design described in Section 4. Section 5 describes our evaluation approach. We report the current status and future work in Section 6.

2 Limitations of Current Curricula

The traditional computer science curriculum is successful in addressing the fundamental concepts of the field, as described in several ACM/IEEE reports on the contents of the CS curriculum [2]. In a typical curriculum, the students are carefully guided through the illustrations of problem solving methodologies and programming skills. These illustrations generally include clear problem specifications, and as a result students are rarely exposed (either in classes or in laboratories) to the development process which precedes these clear specifications. Suitable enhancements to the conventional curriculum could provide a significant exposure to the process, beginning with a broad, unstructured problem and aiming at a structured solution. These enhancements will enable the students to go beyond the following limitations associated with traditional curricula of computer science at the undergraduate level:

1. Teaching of principles and programming skills in isolation from application perspective;

2. Limited laboratory experience;

3. Restricted comprehensive framework that does not provide for enough integration with non-CS components; and

4. Inadequate emphasis on life-long skills such as justification presentations for projects or its subsystems to groups, project planning, cost estimation, group interaction and cooperation, etc.

Some of these limitations have been identified in the past by several educators. Various universities have efforts underway to address these problems [4, 5, 14].

3 The CPI Solution

The primary objective of this initiative is to increase the effectiveness of the undergraduate learning experience. This objective will be achieved by integrating into the curriculum multi-year, real-life projects along with a coordinated infrastructure of formal laboratories. The principle features of this initiative can be summarized as foci of coordinated effort.

Focus 1. *Multi-year project experience:* Students have the opportunity to solve well-specified problems in the regular course work and also to apply the principles in a less structured environment of a project. The following topics will be emphasized:

(a) planning, specification and development; design, implementation, evaluation and improvements;

(b) use of material from different courses; and

(c) management skills, presentation skills, and group problem solving and coordination.

Focus 2. *Experience beyond computer science:* By the choice of a real-life project, the students are exposed to various issues such as engineering, societal and economic impact, and other similar concerns that are not covered in mainstream CS curricula.

Focus 3. *Exposure to leading-edge technologies:* While the main emphasis of this initiative is to develop graduates who are capable of increasing the productivity of the users of computing technology, students learn to increase their own effectiveness, both as individuals and as teams, through the use of leading edge computer technology, e.g. multi-media, hyper-case, and software engineering tools.

Focus 4. *Principles of productivity:* Students are exposed to concepts to improve productivity using computers and measure the productivity gains in the context of a concrete project.

Focus 5. *Portable Curriculum:* As a tangible output, we produce a portable curriculum package so that CS departments at other universities can implement CPI without incurring all of the initial development costs. The proposed package consists of instructional material, organizational material and software.

4 Implementation

A typical industrial project has many factors such as: development of functional requirements and system specifications, selling of the proposal to higher management, construction of a development plan with schedules, budgets, and personnel requirements, recognition and resolution of perceived risks, and development of detailed system specifications. Our implementation adapts the existing curriculum so that a large scale real-life project is developed by students in several courses during several semesters.

During the development phase, the CPI is administered by a committee of seven faculty and two graduate research assistants. One faculty member serves as the coordinator of project elements. Others function not only in an advisory capacity but individually as course coordinators for each of the courses associated with the project. In a steady state implementation, one coordinator and two faculty members in charge of the courses will be adequate in the CPI. The CPI committee is augmented by an advisory board composed of seven members from various government and industrial organizations. The advisory board and committee meet three to four times a year to further define the function and operation of CPI.

With the concurrence of the advisory board, the committee chose to include two lower level courses along with the senior level Computer Based Productivity laboratory courses (CS410 and CS411). CS250 (Problem Solving and Programming) was selected as the sophomore level course primarily because of its long standing use of a formal laboratory and its fundamental problem solving course concept. CS300U (Computers in Society) was selected as the junior level course because of its two laboratory like activities (paper and class presentations) and for its source of critical non-CS resource information for support of CS410 and CS411 laboratories. Additional courses, such as Data Structures, Graphics, and Software Engineering, may be incorporated into the CPI concept but only as informal, non-participating courses for evaluation purposes. The advisory committee will provide feedback to the student presentations, thereby closely simulating a real-life industrial and/or government scenario.

The selected CPI project is carried out over the period of one year (or more years if the problem is sufficiently large) in various courses. The two main courses at senior level, CS410 and CS411, deal with the generation of a solution to the project; this step involves many issues not traditionally considered in software development for university course projects. The goal of this initiative is not to develop commercially acceptable systems but rather to develop system specifications and a plan for its production. The specifications include a detailed design description, the expected functional performance, cost budget analysis, and a production plan based on analyses, simulations, and prototyping. Ideally the product should be such that it could be submitted as an SBIR grant proposal. A CPI project is selected to require a solution that consists of not just computer software or hardware, but requires the design of a system where a computer is only one solution factor albeit an important one. The initial project, Logically Integrated Family Entertainment system (LIFE), as described by the initial 10 students of CS410 involves the hardware and software modules that support many aspects of family life using the family television as a graphic display and communication device (see [8] for details).

The courses involved in the CPI project are linked through the tasking of one class by another. The tasking class is always one of the senior CPI courses (CS410 or CS411). The tasked classes are the sophomore level course (CS250) and the junior level course (CS300U). After some experience, we decided that tasking must be: (i) initiated early in the semester, (ii) responded to well before the end of the same semester, (iii) explained by the seniors to the lower level course, (iv) actively managed by the seniors, and (v) be in line with the objectives of the lower level course. Our aims are to broaden the vision of the students in the lower level courses so that they understand that their activities support a larger enterprise and to give the seniors some experience of managing others. The seniors decide, early in the semester, on an area in which the sophomores should conduct research. Two or three seniors introduce the CPI project to the sophomores and explain how the area of research fits into the whole project. During the course of this assignment, the seniors are available to the sophomores for consultation on their research. Once the proposed hypotheses and experimental designs are submitted, the seniors (all of them, not merely the liaisons) are responsible for critiquing and evaluating them.

Since the CPI project theme will have been used in several supporting courses, computer science students have a sense of participating in a computer productivity project for several years. By the time the student is presented with the entire task of the project, the majority of the basic computer science material has been seen in various courses. The senior students will then be able to readily integrate material of previous core courses into a single project solution at the senior level.

5 Evaluation

We keep track of all problems encountered in the execution of CPI, their alternate solutions and the evaluation methods for the decisions in a database maintained by DHC [10]. In this and the next section we shall address six major decisions and their justification. Our evaluation data is presented in Table 1.

(a) **Teaching CPI:** Based on the data collected using a control group (see [11, 9] for details) our main conclusion is that there is a hardware component necessary to teach the senior courses but most schools should be able to handle it within their regular equipment budgets (the non-traditional equipment - video cards, TV, camcorders - is on the order of 10K). The software we use for presentation - Islands, Office - may also require additional funds if no equivalent is available in the department. All other software used in the course is available from us over the net without any cost. Based on the data collected from students, we were pleasantly surprised that students do not spend more time on the senior courses than in a normal project oriented course. We note that students indeed learned what we set out to teach them and that they liked the courses. That in itself does not mean that students actually benefit from CPI. We performed a survey of all students who completed the courses of CPI and asked them again if they benefited explicitly from and attributably to CPI. A large majority answered affirmatively.

(b) **Curriculum Requirements:** The three main alternatives we considered were to make the senior courses required, have them honor sections, or have students make proposals to enter the courses. After the first year offering we received a strong support from our industrial members on the board as to the marketability of our students. As a result, we felt compelled to make every effort make the courses required to ensure that all our students have the experience. The biggest obstacle was a concurrent effort by our University to reduce the number of credits necessary for professional degrees to 124 credits (with 120 credits preferred). Since CS250 and CS300 are already required we needed to add two courses without increasing our number of credits. We were able to do so by making one of our electives into the required CS410 and replaced a general education requirement with CS411.

(c) **Tasking:** This problem concerns the coordination necessary to integrate CS250 and CS300 with CS410 and CS411. The key issue was not how to task students in the lower level courses but on a timely delivery of the results of these tasks back to the seniors so that they can be incorporated into their decision process. In the first year, no effort was made to couple these courses. In the second year, the seniors used the material

	CPI				Comparison	
	CS250	CS300	CS410	CS411	CS390 (theory)	CS476 (project)
Enrollment/sem	60	80	20	20	50	35
#offered/year	2	2	1	1	1	1
Contact time hours/week	3+1 (lab)	3	3+1 (lab)	3+1 (lab)	3	3
Student effort hrs/wk				14	6	14
CPI Specific						
#TAs (20hrs/wk)	.5	.5	1	1	.4	.4
#speakers	1	3	3	3	0	0
Hardware	wkst	wkst PCs	4wkst 2PC TV+VCR 2camcds 2way video cards	4wkst 2PC TV+VCR 2camcds 2way video cards	wkst	wkst
Software	Unix Test environ.	Unix/X	Unix/X Island Office DHC Mosaic emacs19	Unix/X Island Office DHC Mosaic emacs19	Unix	Unix
Student satisf. 6-exc. 1-poor at course time after graduation	4.8	4.2	5.8	4.6		
Pretest/postest			38/54			

Table 1: Summary of Evaluation Data for CPI

produced by students in CS300 and CS250 a semester earlier. In steady state, Fall 94, we have achieved a schedule which includes presentations of the tasks by seniors to students in CS300 and CS250 by the third week and results back from CS250 by the twelfth week and from CS300 results will come in a continuous stream starting in the sixth week. During that period individual seniors serve as expert help to these students and also they will attend foray held in CS300 and provide a summary of the presentations there to the rest of the seniors.

(d) **Advanced Tools:** We decided to use the departmental network of workstations and a locally developed multimedia, hyperlinked system development environment (DHC) to maintain our project database. We were very much aware of the problem of the heavy learning curve of using sophisticated tools in a course. We made a special effort to tailor subsets of the tool environment for the course needs and provide self-guiding, multimedia tutorials on DHC, Island (a presentation tool) and the use of multimedia through TV and workstations interchangeably. The biggest addition in terms of advanced technology we made this year is to provide an interface to allow students to use PCs and their presentation software (we supply Office). This decision is driven by the cost factor of presentation tools specifically and multimedia software in general for Unix versus for Windows (or similar OS).

(e) **Project Selection:** Initially, we considered to be this a major problem but, as it turns out, it has become a non-issue. In the original proposal to NSF we had envisioned a rather complicated sequencing of the four courses. It was thought that we would actually have more than one project being worked on by different student groups at the same time. However, it soon became clear that there is an enormous investment in preparing the senior courses for a particular project.

6 Current State and Future Work

Starting in the Fall 1994, CPI will become part of our undergraduate curriculum. However, CPI is still an experiment in progress and must evolve as we gain a better understanding of its role in an undergraduate education. Our immediate goal is the development of a multimedia hyperlinked instructor's manual which includes slide packages for each curriculum module, example documents supporting the course objectives and the decision space which describes how we arrived at the particulars of the course contents. This manual will be available through MOSAIC over the Internet as part of our effort to disseminate the CPI project to other educational institutions.

Although the course objectives for CPI have stabilized, we will need to continually monitor skills learned and to assess the value of these skills against others which could be substituted in their place. We are aware that any addition to the curriculum means that something else must be sacrificed if the time to complete the undergraduate degree is to remain constant. Thus, while we feel that the set of skills offered in CPI are vitally important to our students and that they are not available elsewhere in the curriculum, we are sensitive to fact that other skills will not be learned. One unforeseen side effect of CPI is the emphasis on user interface design issues which are only briefly covered elsewhere.

Evaluation of CPI will be an ongoing activity as the curriculum evolves and matures. We have found that the external board of advisors provides invaluable insights into current and future directions. Having the students present their project to the board provides a very real incentive towards excellence and can reflect poorly on the institution if poorly done. Although it can be a nerve-wracking experience, it builds the student's confidence in a way that would be difficult to do so otherwise.

Because group interactions are an essential part of the CPI experience, assigning individual grades becomes a difficult problem. In other classes in which we have team projects, the tasks have been divided in a way which allows individual credit to be assigned even if some members of the team fail to perform their tasks. However, with CPI, one of the lessons to be learned is that the success of the team is important and that part of the student's grade will depend on the success of the team. We feel strongly about this since in their careers, students will also be judged in part on the success of their team. However, we also recognize each student needs to be evaluated separately from the team. Therefore, for the Fall 1994, we are planning to use self evaluations to assign individual grades. At the beginning of the semester, the class will be told what the objectives of the course are, and that they must maintain a portfolio which documents their individual efforts in meeting those objectives. At the end of the semester, each student will present this portfolio to the instructor and must persuade the instructor that they have earned whatever grade they feel their efforts justify. It is planned that part of the portfolio will be monthly progress reports.

Although the number of students who have finished the CPI project and graduated is not very large, their satisfaction, obtained from postgraduate interviews, with the experience and the degree to which it prepared them for their careers seems to validate the concept so far.

Apart from the usual activities such as writing papers, making presentation at appropriate conferences, and putting the material under MOSAIC, two dissemination efforts are under consideration: (i) workshops (as held by Clemson University [14] for their NSF project), and (ii) inviting faculty for a sabbatical (as is being done by UVA [15]). A third method we are attempting is to make dissemination a part of another proposal we have made to NSF as part of the NIE initiative. Old Dominion University is in the process of building a statewide highly interactive remote instruction testbed based on networked workstations. Plans are underway to integrate CPI into this initiative to disseminate it to the schools inside the state of Virginia.

References

[1] A report on undergraduate curricula for software engineeing. British Computer Society and Institution of Electrical Engineers, 1989.

[2] ACM/IEEE-CS Joint Curriculum Task Force, 1991. Computing Curricula 1991.

[3] The Case for Change. Commonwealth of Virginia, Commision on the University of the 21st Century, Richmond, Virginia, 1992.

[4] Computer science curriculum. Department of Computer Science, University of Virginia, Charlottsville, Virginia, 1993.

[5] P. A. Bailes and E. J. Salzman. A proposal for a bachelor's degree program in software engineering. In N. E. Gibbs, editor, *Software Engineering Education; SEI Conference 1989*. Springer-Verlag, New York, 1989.

[6] J. Foley *et al.* Report of the NSF Computer Science Education Workshop. *SIGCSE Bulletin*, 20(3), 1988.

[7] P. Denning *et al.* Computing as a discipline. *Communications of the ACM*, 1, 1989.

[8] K. Maly, D. Ray, I. Levinstein, S. Olariu, C. M. Overstreet, N. Rao, T. Ireland, and G. Kantsios. Computer productivity initiative. *Proc. Seventh SEI Conference on Software Engineering Education*, San Antonio, Texas:441–459, January 1994.

[9] K. Maly, D. Ray, I. Levinstein, S. Olariu, C. M. Overstreet, N. Rao, D. Sibol, and G. Panayides. Computer productivity initiative: Past, present and future. *Proc. Eight SEI Conference on Software Engineering Education*, March 1995. Submitted.

[10] K. Maly and C. Wild. Software life cycle support-desicion based software development. *IFIP '92*, pages 72–79, September 1992.

[11] K. Maly, C. Wild, C. Zhang, D. Ray, I. Levinstein, S. Olariu, C. M. Overstreet, N. Rao, D. Rosca, T. Taylor, T. Ireland, and D. Sibol. A decision-based hyper-multimedia case environment for the computer productivity initiative, a large undergraduate project. accepted for publication, June 1994. Proc. World Conference on Educational Multimedia and Hypermedia, Vancouver Canada.

[12] D. L. Parnas. Education for computing professionals. *IEEE Computer*, 23(1):17–22, 1990.

[13] M. Shaw. We can improve the way we teach CS students. *Computing Research News*, 4(1):2–3, 1992. letter to the editor.

[14] J. Turner. private communication. Clemson University, 1994.

[15] W. Wulf. private communication. University of Virginia, 1994.

[16] W. A. Wulf. SE programs won't solve our problems. *Computing Research News*, 3(5):2, 1991. letter to the editor.