Computer Productivity Initiative

Kurt Maly Chris Wild Dennis Ray Hussein Abdel-Wahab
Irwin Levinstein Stephan Olariu Mike Overstreet
Nageswara Rao

March 16, 1992

Contents

I. Executive Summary

1. The Problem

Most students with a computer science education become software technologists, that is, they apply concepts and skills learned in their academic careers to generate or maintain software products. This technology is used either to improve the overall effectiveness of businesses or provide for an increase of personal productivity. However software technology is rarely developed isolated from a host of other concerns. While the traditional computer science curriculum adequately teaches principles and programming skills, it often ignores other dimensions impacted by software technology. The impact of these other dimensions by new graduates is absorbed over a period of time through a kind of apprenticeship. They are generally recognized as mature software professionals only when they demonstrate that they can successfully address non-computer related issues within the context of the overall project. This proposal presents a Computer Productivity Initiative (CPI) which has as its main objective a modification of the computer science curriculum to educate students so that they will not only have a thorough understanding of the fundamental principles of computer science but who can effectively use this knowledge to increase productivity. In this initiative students will address a specific problem, which we will refer to as the Productivity Project, and produce a Systems Engineering Specification. The Systems Engineering Specification will include a detailed hardware and software specification, a cost benefit analysis, an impact study, and an implementation management plan. Parts of the Productivity Project will be completed in several courses over multiple years.

We use as an example productivity project a ship terminal loading/unloading operation, referred to as the VIT (Viriginia International Terminal) project. In the VIT project, ships from around the world unload and load cargo packed in containers at the Viriginia International Terminal's docks. These containers are handled by automated cranes which move the cargo between the ship and waiting trucks and local storage yards. This operation involves the cooperation of many parties including the shipping and trucking companies, several organized labor unions and the owners of the docks and yards. For example, in the VIT project computers may be used to develop plans for stacking the containers to reduce the amount of loading and unloading. Computers could provide a graphical interface for the crane operator to pinpoint the location of the next container to be handled. However to make the best use of computers many non-computer science issues must be addressed. These issues include the education level of the crane operator, constraints from union contracts and company policies and the variability of ship and trucking schedules.

In order to achieve the overall objective of teaching students how to effectively insert software technology within the overall context of a customer's operation, we have indentified the following objectives.

• *Realistic Application of Principles Learned:* Although well specified problems are appropriate for learning principles, students need to learn how to apply these principles in a less structured environment.
• *Systems Problem Solving:* Students need to understand the role that problem solving plays in the solution of systems. They need to define the requirements for the overall problem so-

lution including software components within the framework of non-computer science issues.
• *Programming in the Large:* In traditional curricula large projects are mostly restricted to such courses (or sequence of courses) as Software Engineering, Operating Systems or Compilers. These well structured projects do not address several of the fundamental steps of effective problem solving. The Productivity Project will deal with unstructured problems in a context of a business application. System documentation is not just produced for the instructor to grade but will be used by students in other classes. Requirements analysis, verification and prototyping will involve communications with the "customer" to define to problem structure. The system solution will involve the efforts of students in several courses. This requires that a management plan be generated and implemented to coordinate these efforts.
• *Effective Technology Utilization:* While the main emphasis of this initiative is to provide our students with the skills to increase the productivity of industry through the effective use of computer technology, students must also learn how to increase their own effectiveness, both individually and as teams. Thus the term productivity takes on two meanings in this proposal, i.e., the productivity of our students in using advanced software technology and the ability of our students to propose computer solutions to make other people more productive.

2. The Solution

Our plan is to meet the objectives of the proposed initiative through a formal laboratory experience that spans the four year undergraduate program. The curriculum to support the CPI involves the selection and refocussing of existing courses and laboratories and the introduction of new courses and laboratories. The curriculum structure was derived from a model of the activities to be undertaken in the Productivity Project. Fig. 1 shows the process flow for the Productivity Project in terms of activity modules.

In module M1 students are given a problem statement selected by the executive committee. They must analyze and understand the problem statement and identify the major issues to be resolved.They must then define a set of task activities to resolve these issues. It is in the performance of this module that the students will interact with the customer and domain experts. These activities are achieved by tasking courses within the curriculum. The tasks include impact studies, prototyping and simulation and theoretical analysis. In module M2 students generate these specifications for a software study and a management plan to be implemented by the *Software Engineering Lab.* The students define problems for use in the *Problem Solving Lab.* The requirements document and the management plan will undergo an extensive review by the students themselves, the lab supervisors responsible for the affected courses, and the Project Coordinator. In module M3 students implement the management plan by supervising major aspects of the *Software Engineering Lab.* In module M4 students analyze the results of previous studies and generate a set of feasible solutions. The results of this analysis is presented to the customer and domain experts and the potential productivity improvements are assessed. In module M5 students generate the system specification, the problem posed as input to M1. This specification addresses computer hardware and software components, non-computer hardware (such as cranes, sensors) and the interfaces among these components (this comprises the systems architecture for addressing the Productivity Project). In addition, the System Specification includes cost/benefit

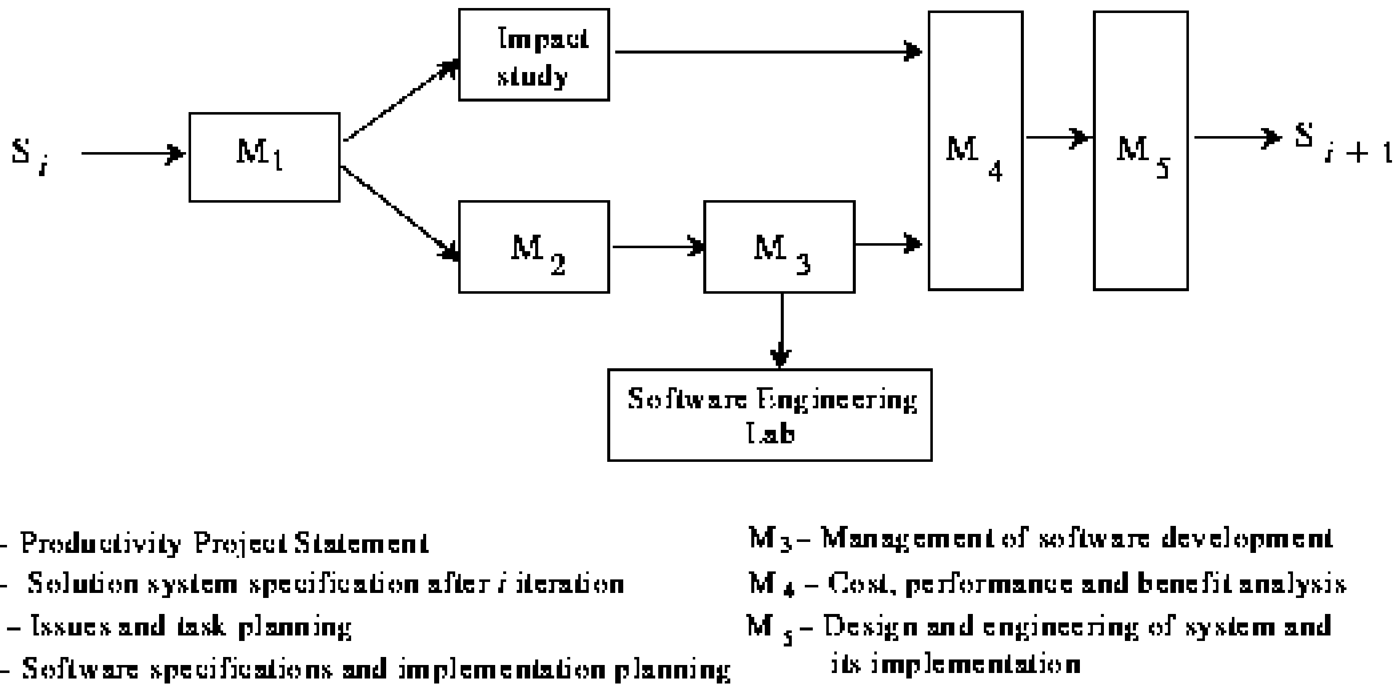

Figure 1: Process Flow for Major Course Modules

and performance analysis, prototypes and simulations, operator and training manuals and a management plan for the implementation of the systems solution.

The implementation of the CPI consists of four formal laboratories, each attached to a computer science lecture course, one study that is part of an existing course, along with two senior laboratory courses. These seven courses are organized into two tiers. The upper tier consists of the following three courses:

• Project Design Lab (CS400, new course). This course implements activities modules M1, M2 and M3.

• Systems Engineering Lab (CS401, new course). This course implements activity modules M4 and M5.

• Impact Study (attached to CS300, Computers in Society). Students in this class will use the Productivity Project theme as the focal point of their reports. The specific areas of the project to be evaluated will be received as products from the Project Design Lab. The impact report is a semester long effort of detailed research into the impacts of computers on the problem at hand. In the VIT problem, for example, typical reports may be: 'International Business', 'Integrating Different Levels of Technology', 'Automation and Unions', 'The Impact of the Educational Level of Users on the Computer Interface'.

The lower tier consists of the following four courses:

• Software Engineering Lab (attached to CS350, Software Engineering). This laboratory will implement, using teams of students, software subsystems of the Productivity Project, develop prototypes, and use simulations for other subsystems. All of the problems are defined in module M2. The implementation is managed by members of the Project Design Lab.

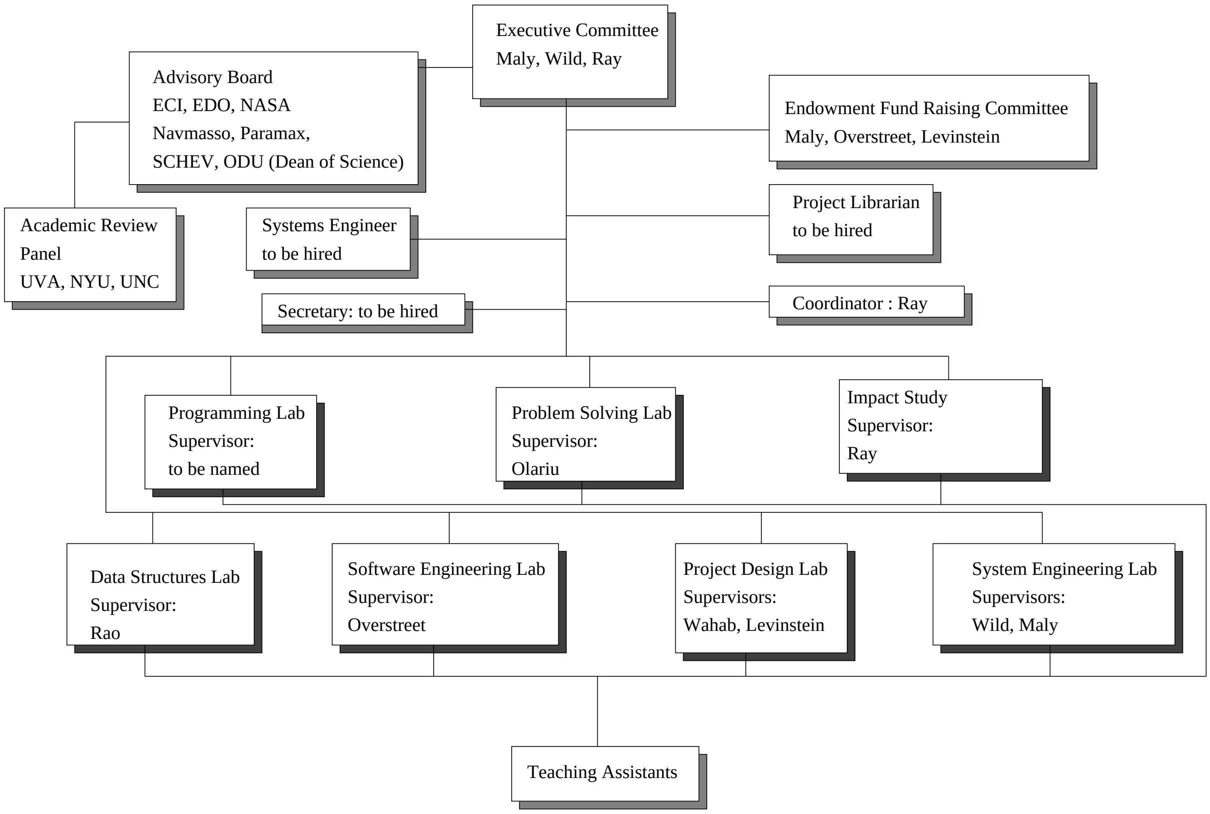

Figure 2: Management Structure

• Data Structures Lab (attached to CS360, Data Structures). The main purpose of this supervised lab is to provide an in-depth hands-on experience in the design of generic modules encapsulating abstract data types. Specifications for needed abstract data types are generated by the Software Engineering Lab and the resulting software components are placed in the software laboratory.
• Programming Lab (attached to CS150, Introduction to Programming). The major objective of this existing lab is to assist students in mastering the fundamental programming skills required in upper level courses. Programming assignments will be derived from the software specifications generated in *Software Engineering Lab* where such specifications coincide with the educational objectives of this lab. This lab is not tasked to produce components needed by the project. The major benefit is the early introduction of the CPI to the students.
• Problem Solving Lab (attached to CS250, Problem Solving). This is an existing lab with well defined educational objectives including introduction to problems solving methodology and advanced programming techniques. Software problem specifications derived from module M2 are used to meet these objectives. The resulting software is added to the project software library for inclusion in the software study. In addition students in this course will run experiments with parts of the project developed in the *Software Engineering Lab*. These experiments will be used to collect testing and performance data for later analysis.

While the rest of the courses in the CS and CE curriculum are not formally in this proposal, we believe that individual course projects already in place may draw inspiration from the Productivity Project, using it both as the background for examples and a fertile ground for experimentation with principles learned.

Management Plan:

The CPI will be managed by an executive committee that is responsible for overall project planning, financial disbursements, personnel assignments, and CPI review. The committee is assisted by an industry/government advisory board (gives strategic directions, assists in project selection, and evaluation of effectiveness), and an academic review panel (guides curriculum structure, evaluates project curricular impact).The endowment committee is changed to establish a permanent endowment to provide financial support for both outside speakers and to fund students to visit customer sites. This structure is presented in Fig. 2.

3. Results

By the end of the third year we expect to achieve the following three goals:

1. Our students will be more effective in addressing the complex issues arising in the solution of large scale problems. By refocussing the existing curriculum the additional effort required by our students to achieve this increased effectiveness will be relatively modest.

2. We would have established a curriculum which can serve as a model for the computer science curriculum of the twenty first century.

3. The experience and expertise gained through this initiative will enable us to serve as a leading university for transferring the CPI concept nationwide.

II. Proposal Narrative

1 Introduction

Computer Science is fundamentally a laboratory science. Its concepts are best illustrated with a repetitive application of a sequence of investigative actions: analysis, proposed solution, experimentation, observation, and new analysis. The traditional computer science curriculum is successful in addressing the fundamental concepts of the field, as described in several IEEE reports on the contents of the Computer Science Curriculum. Students are carefully guided through the development of problem solving methodologies and programming skills. They are exposed to theoretical concepts which underlie computer science. These theoretical concepts are illustrated using well understood and well structured applications such as compilers, operating systems and databases. What is missing, however, is an adequate exposure to the process that originates with a broad, unstructured problem and whose goal is to arrive at a structured solution. In order to address the decisions the Department of Computer Science proposes the Computer Productivity Initiative (CPI) for the undergraduate curriculum.

Our primary objective is to produce students who are capable of understanding the needs of users of computing technology and of producing solutions which will increase the productivity of these users through the application of computer technology. As history has taught us, the availability of technology is not enough. We must also learn to make effective use of that technology [9, 5]. We propose a curriculum that extends the utilization of principles and skills learned in the traditional computer science courses by the addition of a lab infrastructure that incompasses most of the undergraduate computer science core courses. This infrastructure will be coordinated to focus on a Productivity Project. The project will be of sufficient breadth and depth that will address issues involved in solving a large scale problem that uses software technology. Of necessity, the successful prosecution of a Productivity Project will require a multi-year coordinated effort.

The CPI addresses the solution of problems such as automating toll booth collection on a highway, speeding the mail handling in a university post office and the utilization of computers to increase the productivity of a manufacturing business process. Students will address a specific problem which we will refer to as the Productivity Project and produce a Systems Engineering Specification. In order to clarify this presentation, we will use as a running example the problem described in Appendix B - a container ship loading/unloading process referred to as the VIT (Virginia International Terminals) project. The generation of a solution to a Productivity Project will address many issues not traditionally considered in software development for university course projects but which are critical components of the productive use of software technology. For example, in the VIT project computers may be used to develop plans for efficient stacking of containers to reduce the amount of loading and unloading. Computers may be used to provide a graphical interface for the crane operator to pinpoint the location of the next container to be handled. However to make the best use of computers several non-computer science issues must be addressed; these include the education level of the crane operator, the effect of union contracts and policies, and the variability of ship arrival schedules.

The goal of this initiative is not to develop commercially acceptable systems but rather to develop a System Specification. This specification will include detailed design, its expected performance, its cost budget and a production plan based on analyses, simulations, prototyping and reports its cost budget, and a plan to produce the system can be developed. In many computer science curricula, programming activities require inordinate effort on the part of students. The results are often unsatisfactory since too little analysis of the problem is undertaken. We are concerned with teaching our students the process of solving problems in the context of its impact on society [6].

The major innovations of this initiative are:

1. Application of academic principles to projects that will produce graduates who can improve productivity in industry and government.
2. Consideration of other dimensions of problem solving which impact and are impacted by software technology.
3. An infrastructure which integrates a coordinated set of formal laboratories into the curriculum.
4. Coordinated study of Productivity Project over multiple years.
5. Extensive student participation in all aspects of this project. Upper levels students will manage project requirements for lower level students.
6. Effective utilization of computer technology emphasizing the problem solving process [12].

Initial selection of computer technology includes support for collaborative work [1], both as an instructional delivery technology and for group problem solving and computer assisted requirements engineering and multimedia documentation services.

This initiative is in accord with many of the areas identified in the NSF sponsored *Workshop on Undergraduate Computer Science Education* [7]. The use of a coordinated multi-year Productivity Project with the addition of the advanced project laboratories that address systems problem solving in which computer technology can provide significant productivity gains represents a major initiative in **Curriculum Development**. The implementation of this project using formal labs for the lower level courses and lab facilities to support team problem solving for the upper level courses provides the necessary **Laboratory Infrastructure**. These labs will be supported by emerging technologies in computer collaboration support and decision based problem solving. The use of collaboration facilities and multimedia (specifically video tapes for team interaction evaluations) addresses new approaches to **Instructional Delivery Systems**. The faculty will gain a much deeper appreciation for the role that computing technology can play in solving systems problems and in increasing productivity. This in turn may generate new joint academic/industrial research ventures which have been recognized as a critical need by the Computing Science Technology Board [3] (**Faculty Enhancement**).

2 Objectives

Our overall objective of this initiative is to increase the instructional effectiveness of the undergraduate learning experience. This objective will be achieved by applying a multi-year, large scale real-life productivity project in an infrastructure of formal labs. The effect of the proposed initiative will be the education of computer scientists who have a thorough understanding of the underlying CS principles, and who can also apply this knowledge to enhance the productivity of businesses and individuals. The CPI enriches the learning experience of our students in the following ways:

• *Realistic Application of Principles Learned:* Students will have the opportunity of solving well-specified problems in regular course work and also applying these principles in a less structured environment of the productivity project. The hands-on experience in the laboratories provides both a motivation and application of the principles.

• *Systems Problem Solving:* Students will appreciate the role of problem solving in the solution of large real-life applications by defining the requirements for the overall solution (including software components) within the framework of non-computer science issues [8].

• *Programming in the Large:* In a traditional curriculum, the projects are mostly restricted to courses (or sequence of courses) such as Software Engineering, Operating Systems, etc. The CPI adds several dimensions to these projects (see also [10]). The programs and documentation produced will be used by students in other classes in addition to meeting the course requirements. Requirements analysis, verification and prototyping will involve communications with the "customer." Components of the system will be produced in various courses. The latter requires that a management plan for producing these components be generated and implemented.

• *Effective Technology Utilization:* 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 will learn to increase their own effectiveness, both individually and as teams, through the use of computer technology [11].

3 Issues and Options

The undertaking of the proposed initiative is not without risks: we recognize these risks and plan actions to eliminate and/or to control them. In the most abstract sense a solution to meeting our objectives is the selection of a non-trivial Productivity Project, development of specifications by the most experienced students (seniors) and management (again by seniors), of students in the lower level courses to obtain needed software and impact studies. The mechanism is a series of additional courses and the addition of formal laboratories to existing selected courses. This section outlines our concerns about potential difficulties in this undertaking as well as possible solutions.

• *Management of Multi-Year Projects*: Multi-year projects place a dependency between courses: a failure in a prerequisite course will cause problems in later courses. Addressing this concern will place a major emphasis on the intercommunication and coordination between lab supervisors and the Project Coordinator.

• *Quality Assurance*. We believe that this emphasis will be one of the innovations of this proposal. That is, the project does not emphasize building something but rather the process of ensuring that there will be quality in the final product. This objective can be achieved by a combination of the following: (i) making quality a requirement, (ii) being prepared to throw away the earlier versions, and allowing time for the improved version to be built, (iii) having gold versions generated by faculty, (iv) students evaluating and fixing the best of their efforts.

• *Beginning Students may not be Ready:* We propose to use the project examples in the formal laboratories at the lower levels. This introduces CPI to the lower level students. Over time, these projects will form a library of building blocks that can be incorporated into future projects.

• *Curriculum will Serve the Project:* The production of software to be used by the customer is not a project deliverable. We should limit project exposure to manageable number of courses - clearly defining the educational objectives of the project in that course. We feel by clearly defining the objectives of each of the labs involved in the project, it will be possible to give all students the advantage of a project oriented curriculum while maintaining flexibility in determining a specific set of activities that any particular class will undertake.

• *Impact on Degree Requirements:* We can reclassify some courses, e.g., by moving certain mathematics requirements to general education, we can adjust the contents and directions of existing computer science courses and laboratories. We can also use some electives and if absolutely necessary increase the number of credits. Care will be taken not to interfere with the CSAB and ABET accreditation.

• *Difficulty for Transfer Students:* Students can read documentation on the Productivity Project, talk with other students, and the Project Librarian can run orientation seminars at beginning of each semester for transfer students. It is expected that experienced students will "instruct" the new comers (this is part of the project experience).

4 Curriculum Structure

Our plan is to meet all the objectives of the proposed initiative, outlined in section 2, through a formal lab experience that spans the four year undergraduate program. The novelty of our approach is that the multi-year initiative will serve as a focal point to the curriculum: it provides a natural source of examples and motivation for a number of "hard-to-illustrate" concepts; it also gives the students experience in problem solving "in the large". The CPI will add a new dimension to our entire undergraduate curriculum: students will see how the theoretical concepts fit together in an effort to solve a credible real-life problem. Conversely, the project will stimulate the students in various proposed labs to find solutions to many problems arising in computer science as well as in non-computer science areas of human activities. In some cases, students will have to understand several ramifications of the proposed solutions, including social and societal impact.

The curriculum structure was derived from a model of the activities to be undertaken in the productivity project. Fig. 3 shows the process flow for the productivity project in terms of activity modules.

Students participating in module M1 are given a problem statement selected by the executive committee and are concerned with understanding the project and in identifying the major issues to be resolved and a set of task activities which should be undertaken to resolve these issues. It is in the performance of this module that the students will interact with the customer and domain experts brought in for consultation. The resolution of these tasks are achieved by tasking other courses within the curriculum with impact studies, prototyping and simulation and theoretical analysis or by internal analysis. Students involved in module M2 generate a software specification for a major software study to be implemented by the *Software Engineering Lab* and a detailed implementation plan. They also define software

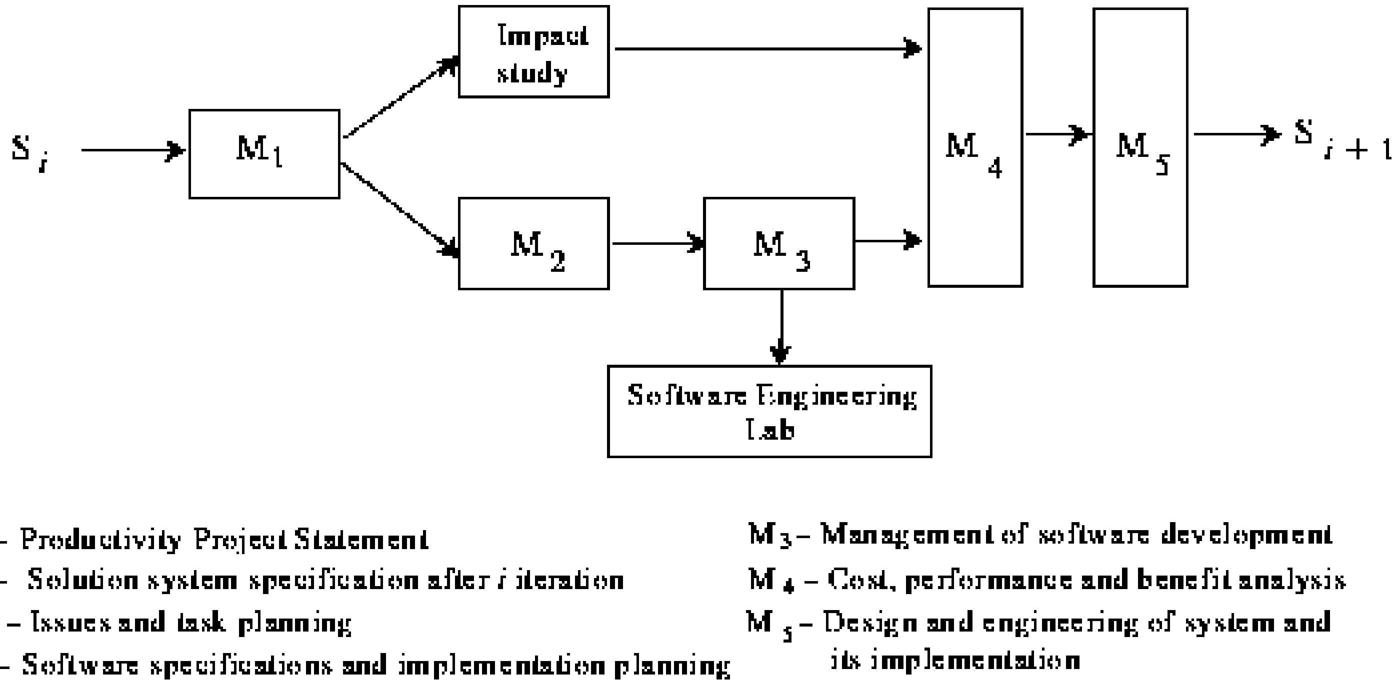

Figure 3: Process Flow for Major Course Modules

components to be generated in the *Problem Solving Lab.* Both the requirements document and the management plan will undergo an extensive review by the students themselves, the lab supervisors responsible for the affected courses, and the Project Coordinator. During module M3, students manage this implementation plan by supervising major aspects of the *Software Engineering Lab.* Students involved in module M4 study the results of all the studies and define a set of feasible solutions. The results of this analysis is presented to the customer and domain experts and the potential productivity improvements are assessed. Module M5 generates the specification of a system to address the original problem posed as input to M1. This specification addresses computer hardware and software components, non-computer hardware (such as cranes, sensors) and the interfaces among these components (this comprises the systems architecture for addressing the productivity project). In addition, M5 also produces cost/benefit and performance analysis, prototypes and simulations, operator and training manuals and a management plan for the implementation of the systems solution.

The implementation of the curriculum consisting of four formal, closed lab, each attached to a computer science lecture course, one study, part of an existing course, along with two senior courses is described in Fig. 4.

The upper tier consists of the following three courses:

- *Project Design Lab* (CS400, a new course): This course implements activities modules M1, M2 and M3.
- *Systems Engineering Lab* (CS401, a new course): This course accomplishes activity modules M4 and M5.

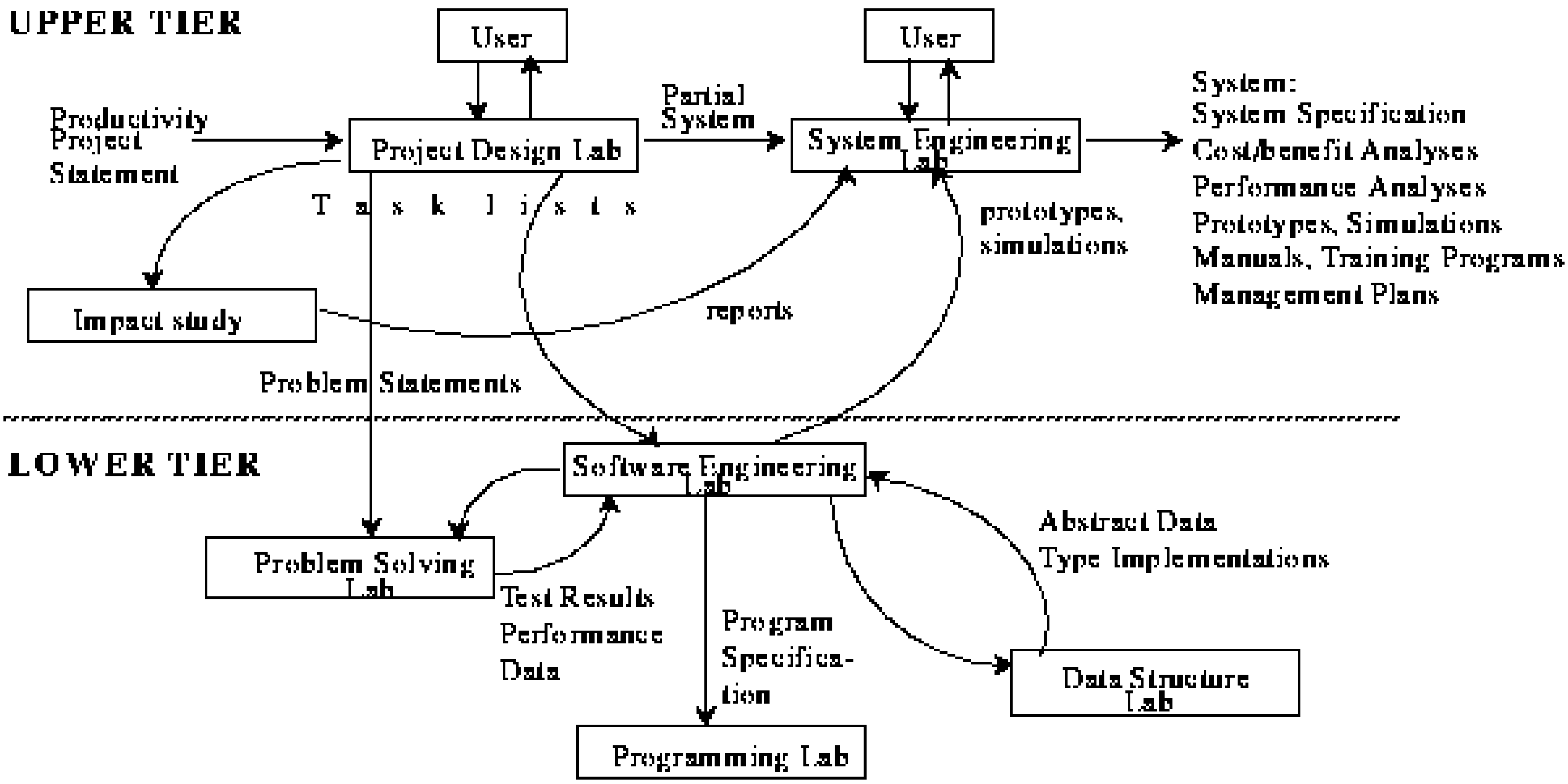

Figure 4: Productivity Project Curriculum Structure

• *Impact Study* (attached to CS300, Computers in Society): Students in this class will use the Productivity Project theme as the focal point for their reports. The specific areas of the project to be evaluated will be received as products from the Project Design Lab. The impact report is a semester long effort of detailed research into the obvious and subtle impacts of using computers in the customer's operation. In the VIT problem, for example, typical reports may be: "International Business", "Integrating Different levels of Technology", "Automation and Unions", "The Impact of the Educational Level of Users on the Computer Interface".

The Lower Tier consists of the following four labs:

• *Programming Lab* (attached to CS150, Introduction to Programming): This is an existing closed computer-supported lab with a hands-on experience in problem solving "in the small". The major objective of this lab course is to assist students in mastering the fundamental programming skills required in upper level courses. Programming assignments will be derived from the software specifications generated in *Software Engineering Lab* where such specifications coincide with the educational objectives of this course. This lab is not tasked to produce components needed by the project. The major benefit is to introduce students early to the CPI.

• *Problem Solving Lab* (attached to CS250, Problem Solving): This is an existing course with well defined educational objectives which include introduction to problem solving methodology and advanced programming techniques. Software problem specifications derived from module M2 are used to meet these objectives and thus also serve to introduce students to the CPI. Again the resulting programs are not considered essential to the project. In addition students in this course will run experiments with parts of the project developed in the *Software Engineering Lab*. These experiments will be used to collect testing and performance data.

• *Software Engineering Lab* (attached to CS350, Introduction to Software Engineering): This lab will implement, using teams of students, software subsystems of the Productivity Project, develop prototypes, and use simulations for other subsystems. All of the problems are supplied by seniors who manage the software engineering teams.

• *Data Structures Lab* (attached to CS360, Data Structures): The main purpose of this supervised lab is to provide an in-depth hands-on experience in the design of generic modules encapsulating abstract data types. The modules to be implemented will be useful not only in the Productivity Project, but also in subsequent upper level undergraduate courses such as algorithms, operating systems, compliers, etc. For the project modules, the design will be performed in the classes in appropriate places; the designed data structures will then be implemented in the subsequent lab session.

While the rest of the courses in the CS and CE curriculum are not formally in this proposal, we believe that individual course projects already in place may draw inspiration from the Productivity Project, using it both as the background for examples and a fertile ground for experimentation with principles learned.

Details of objectives, resources and examples of individual laboratories are given in Appendix A.

5 Technology Support

We plan to exploit emerging technologies in software process modeling, collaboration research, and multi-media presentation to increase the productivity. Although we see the injection of maturing research concepts and hardware/software innovations as a continuing commitment, for the initial effort we plan to integrate three state-of-the-art technologies with our curriculum: XTV for collaboration and instructional delivery, Decision Based Software Development (as implemented in the CASE tool D-HyperCase) for requirements analysis and system analysis, and multimedia technology for documentation and laboratory teaching.

5.1 XTV: Computer-Supported Collaborative Work System

X Teleconferencing and Viewing (XTV) [2] is a distributed system that allows the sharing of tools and applications in real-time by a group of users linked by a network using any type of workstation running X. XTV [2] is an ongoing joint research project between ODU and UNC-Chapel Hill and the first release is now available in the public domain through MIT X11R5 distribution of the X Window System.

XTV is an educational enhancing tool that will be used in the Problem Solving and the Data Structures Laboratories as an instructional tool, since in both labs the students will be using workstations running X. The instructor will use XTV to run and control unix tools (e.g. editing, formatting, previewing, figure drawing, compiling and debugging tools) and allow students to take turns in controlling and manipulating the tools from their own workstations. XTV allows late students to join an ongoing class and review previous or missed material at their own pace. A student may use XTV alone outside the regular class and start working on a programming project. When confronted with a problem that the student cannot solve, the student can request an instructor join in the current setting and together solve the problem. The instructor can witness the student's problem even though the instructor is not at the student's workplace.

XTV provides extensive facilities for brainstorming and free discussions among the participants. By using the whiteboard-like facility, any participant may freely handwrite or sketch figures. The contents of the board can be imported by anyone in whatever form the contents are in. In addition anyone can cut pieces or sections from any window and past it on the whiteboard for others to see. XTV provides a sophisticated communication facility (chat) that allows the participants to talk to each other as a whole or as subgroups and send messages (including voice), including anonymous comments to freely express themselves. These aspects of XTV will be used in the senior level courses.

5.2 D-HyperCase: Decision-Based Engineering Case Tool

A problem with most current software development methodologies is that they are tied to specific software products (modular design, data flow, etc). Because of this most approaches to software development do not adequately support the creative and sometimes chaotic behavior characteristic of requirements capture and analysis. Over the past several years, we have developed the Decision Based Software Development (DBSD) Paradigm [13] which we believe better supports the problem solving process inherent in engineering projects. In DBSD the process is documented by articulating problems, identifying alternate solutions, justifying the set of solutions in the context of the total system and finally implementing the solution. By documenting the decision structure, DBSD provides support for the engineering process long before any product structure becomes apparent. However once the final product is implemented, the decision structure provides an alternate grouping of the solution, from the point of view of the decisions made, which is critical in supporting subsequent change activities.

Because problem solving and decision making spans the engineering life cycle, DBSD can support the generation of entire set of documentation produced throughout the project's lifetime. For this proposal, we will apply DBSD to generation of the requirements document itself. This decision is based primarily on the lack of computerized support for this activity. If experience shows this effort to be successful, we plan to utilize DBSD in the other stages of the project as well. DBSD is currently supported by a software engineering tool called D-HyperCase which runs under X windows. It is important to note that the DBSD paradigm in general and D-Hypercase in particular can be applied to any problem solving activity. This is important because our requirements document contains not only specifications for software modules but also interface specifications to other components of the total system.

5.3 Multi-Media

Both XTV and D-HyperCase support multimedia access. We will use the existing audio linking in D-HyperCase and in the future add video access as well. With video access, it would be possible to explain a problem through the use of video tapes taken at the customer's site. Since access to real customer sites will be limited, by video-taping customer site visits, students can "revisit" using these video tapes to clarify problems. These tapes will also be used to inform the new students about the Productivity Project. Where appropriate, portions of these tapes can be tied to individual problems in the decision structure of DBSD and accessed through D-HyperCase.

We also expect that video-tapes of group interactions (such as a code review or decision review) will be invaluable for training students on the subtle points of group interaction. Video tapes of invited guest lectures will also be available and can be tied to project documentation where appropriate. Video-conferencing and collaboration by means of workstations together with the needs expressed above all point towards a general multi-media library facility with much of the material available in digital form for presentation and use on high performance workstations.

6 Management Plan

The project will be managed by an executive committee with the PI as the chair; its staff consists of a systems engineer, a project librarian, a project coordinator, and a part-time secretary. The endowment fund raising committee and all the lab supervisors report directly to the executive committee which consults with the advisory board and the external academic review panel. Teaching assistants work for the lab supervisors. Figure 5 shows the interrelationships among these entities.

Their interactions and responsibilities are as follows.

- *Entire Staff:* In the first semester and the three subsequent summer semesters the main effort is towards the planning and the preparation for the academic year. There will be bi-weekly meetings of all participating faculty to discuss matters of concern to the entire

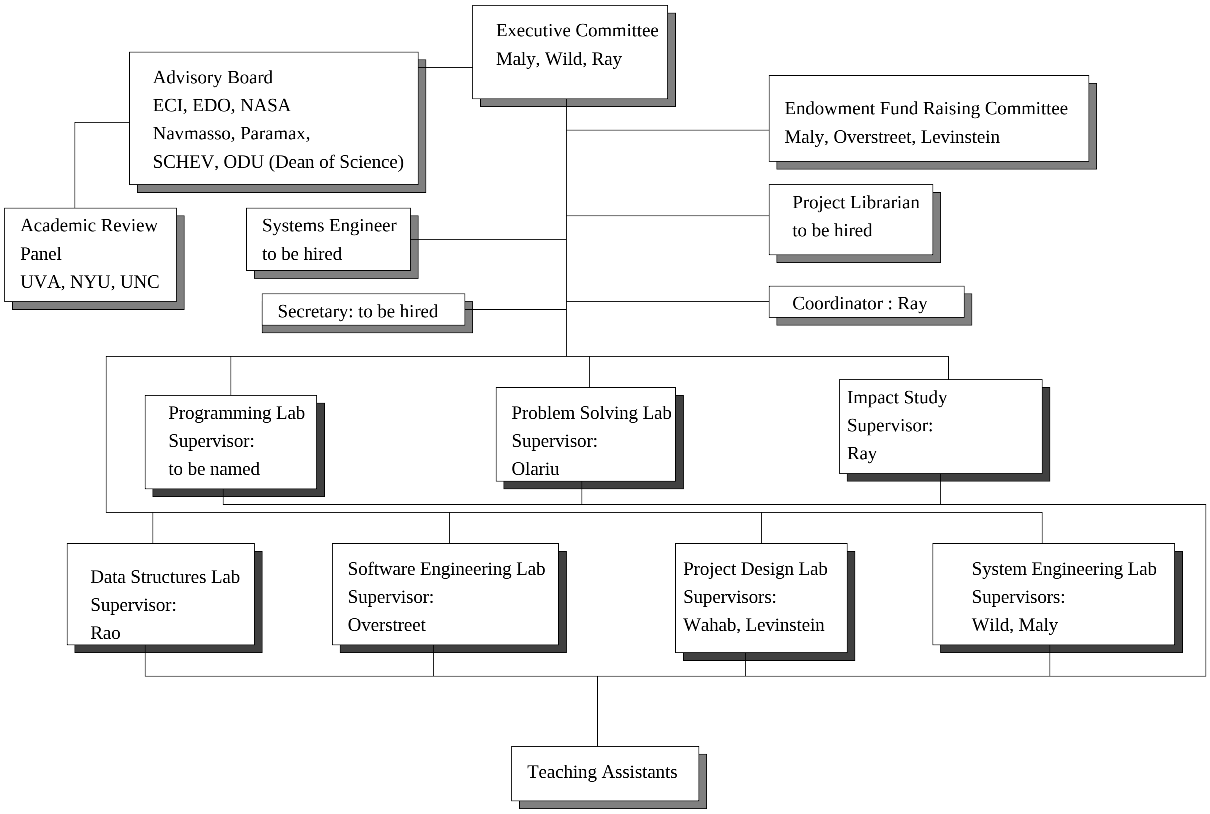

Figure 5: Management Plan

project such as problem selection, solution approaches, deliverables to be produced by each lab, progress reports, and coordination plans. Individual lab supervisors will be responsible for developing the syllabi and all necessary material for their labs. In steady state, monthly meetings will be held with all participating faculty to monitor progress and address unanticipated problems.

• *Advisory Board and the Academic Review Panel:* The advisory board meets at least twice a year and is charged with four major responsibilities. First and foremost is the monitoring of progress, giving strategic directions and advising in the Productivity Project selection process. Secondly, the board serves as the evaluation agent together with the external academic review panel to provide feedback to the Productivity Project. Thirdly, the board is a key resource to the fund raising committee and finally, the board will help insert the concept into the academic world if the Productivity Project proves to be successful.

• *Executive Committee:* The executive committee has overall responsibilities to manage the project, including all hiring, allocation of resources, and is responsible to NSF for funds being spent according to the proposal. It is the final decision body on all academic and financial matters.

• *Endowment Fund Raising Committee:* The project assumes that in steady state, after the NSF funded period, the project will have funds available to continue the practice of inviting outside speakers to delineate the scope of a chosen project problem. The committee is charged to establish a permanent endowment by raising funds from local and national companies as well as individuals agreeing with the aim of this project.

• *Project Librarian:* The responsibilities of the librarian are twofold, one, to provide a resource library and two, to maintain the project problem status in an electronic documentary. The resource library will consist of such items as journals, proceedings, video lectures, case tools, software environments, software manuals, and books. The problem library will contain the various documents relating to the project problems such as requirements analysis, management plans, source code, user interviews, expert lectures, and reports. The librarian will make the library available to students at reasonable hours and to all staff on an unlimited basis.

• *Systems Engineer:* The systems engineer is responsible to maintain all equipment, handle all purchases, and where appropriate obtain service contracts.

• *Lab Supervisors*: Supervisors have to prepare all materials, teach (or supervise the assistants to teach) the labs, and coordinate with the faculty members (if not themselves) teaching the corresponding lecture courses.

• *Project Coordinator:* The project coordinator is responsible for supervising the day-to-day operation of the overall project and implements the directives of the executive committee.

The summary of the responsibilities is as follows:

Faculty Member	Responsibilities
Kurt J. Maly	Executive Committee, Endowment Fund Raising Committee Supervisor for Systems Engineering Lab
J. Christian Wild	Executive Committee, Supervisor for Systems Engineering Lab
Dennis E. Ray	Executive Committee, Project Coordinator, Supervisor for Impact Studies
Hussein Abdel-Wahab	Supervisor for Project Design Lab
Irvin Levinstein	Endowment Fund Raising Committee Supervisor for Project Design Lab
Steve Olariu	Supervisor for Problem Solving Lab
Mike Overstreet	Endowment Fund Raising Committee Supervisor for Software Engineering Lab
Nageswara Rao	Supervisor for Data Structures Lab
To be hired	Systems Programmer,Secretary
To be hired	Project Librarian, Supervisor for Programming Lab

7 Implementation and Evaluation Plan

• *The Upper Tier Cycle:* The Computer Productivity Initiative described in Section 4 has two focal points. One, the development of specifications of systems, the justifications for these specifications, and a plan to produce the system. We will use systems which are intended to make people or processes in some application area more productive. As presented, this activity is mostly performed in the upper tier. A second focus is the involvement of students across four years in the management and production of software components which are used as justifications to engineer the specifications of the system. These foci prove to have requirements difficult to reconcile with a four year curriculum when we attempt to map the structure described in Figures 3 and 4 into a sequence of courses. This is particularly difficult when we have to account for the readiness of students for the tasks of a particular course. The most natural way is to map modules 1 and 2 of Fig. 2 into a one semester course, followed by a course for module 3 which is run concurrent with the *Software Engineering Lab* and the Impact Study. We can then conclude with a semester course containing modules 4 and 5. However, that implies a three semester sequence in the senior year. Instead, we propose a pipelined course structure where modules 1, 2 and 3 comprise the *Project Design Lab* and 4 and 5 the *Systems Engineering Lab.* In the *Project Design Lab* one group of students will work on the issues and prepare task lists and management plans for the Impact Study and Software Engineering Lab to be performed the following year. At the same time another group will implement the management plan produced the year before with the students of the Software Engineering Lab. The Impact Study group will perform the tasks requested last year. The Systems Engineering Lab will take the output of these two labs and produce the specification for this subsystem. That is, in any given year the Impact Study, the Software Engineering Lab and the Systems Engineering Lab will work on completing the specifications for S_i while at the same time the Project Design Lab works on the task lists for S_{i+1}. We use subscripts only to indicate flow of time. There is no dependency implied between subsystems. By definition subsystems as we use them are portions of the total system which are more or less independent of each other. One advantage pipelining has over the sequential presentation of courses is that students who are developing specifications and management plans for the new subsystems can learn from their work in implementing the specifications and plans from the previous subsystems.

• *The Lower Tier:* Labs at this tier are much easier to realize. Each receives specifications relating to some software component and is tasked to produce them. The Software Engineering Lab is the lab responsible for providing the tasks for the other labs in the lower tier. Since these labs have objectives to meet other than the ones relating to this initiative there is sufficient time to interweave the tasks from the Software Engineering Lab with the regular tasks of these labs. Some of these labs are offered more often than the senior level courses in which case students will work on the same problems of the last offering or from a pool of problem specifications which accumulates over the years. The library of software components is updated if the newer solutions to old problems prove better than existing library components.

• *The Overall Cycle:* A Productivity Project begins with the selection of a project and with the preparation of the statement (S_0) and the output of modules 1 and 2 for the first sub system. Next all courses but the Systems Engineering Lab start with material to be used in the lab following in the next semester. We iterate for as many years as incomplete subsystems are identified. Before the last iteration of the Productivity Project is started a new project is selected and the overall cycle begins again.

• *Implementation of the Computer Productivity Initiative:* In Table 1 we have summarized how we envision to implement an Productivity Project over the three years of this initiative and how the cycle will continue after the NSF funding ends. The first year is essentially a preparatory phase to enable us to start in Fall '93 with the first iteration , subsystem 1, in the Project Design Lab. The first major milestone, end of Summer '93, will have all facilities ready and the task lists and management plans for the selected software components available. The second major milestone, end of Summer '94, will have incorporated the experiences of the first iteration into modifications of the plans for the second iteration. It also will mark the selection of the Productivity Project to start in Fall '95 in order to start the tasking for the new Productivity Project as well as the second iteration of the first project. In the table we have marked by † course offerings which replicate previous assignments and by ‡ those which represent a dry run to gain experience.

• *Evaluation Plan:* The evaluation of our success will be in answers to such questions as:

1. Did we have to eliminate material from existing courses to accommodate teaching of the new technologies and solving the Productivity Project?
2. Are we turning students (and faculty) off because our courses are too time consuming?
3. Are students actually producing an acceptable solution to the Productivity Project?
4. Is the benefit (better educated students) worth the cost?
5. Is the flow of information between the tiers meaningful, i.e., do students of the lower tier benefit from the examples and do they produce products usable for the upper tier?
6. Is the pipelining (students do not supervise their own plans but those produced a year earlier) a significant negative factor?
7. Is having two foci (engineering large, non-CS, productivity systems, and managing multi-year software developments) too much for the initiative?
8. Should we expand the Problem Solving Lab to include higher level theory courses covering specific problem solving techniques in depth? If we do so, this lab could then actually feed information back to module 1 in the Project Design Lab.

We plan to answer these questions by establishing a set of baseline metrics and measuring during the entire period such items as: time spent by students, faculty and staff, number of students participating in various courses, material covered, satisfaction (student surveys). Qualitative issues such as the pipelining, multiple foci, information flow, benefits, and problem solving issues will be evaluated by analyzing the products of the students. A major help in this task will be the members of the academic review panel and the advisory board.

8 Contribution to Institutional Plan

Old Dominion University has a stated plan to move into the twenty-first century in a spirit of creative experimentation. The location of the university in one of the world's major seaports, places it at the center of a complex of seven major cities in the Tidewater Virginia regional area. This area is the center for Eastern Virginia banking and is a major area for research and development for extensive scientific and technological activities in many areas. These areas include marine science, ship design and construction, health specialities, aerospace, advanced electronics, and nuclear physics. The large industrial technology base places a great demand on the university for educational training and support. The university meets this demand with programs that place particular emphasis on education with a significant commitment in science, engineering and technology. The Computer Productivity Initiative is a prime example of one of these programs. It will expand the educational effectiveness of the Computer Science Department and will provide an increased capability for students who graduate to make the transition into the commercial, industrial, business, or government in the twenty-first century. The curriculum developed in this initiative fully supports the university's commitment to provide for and expand the scientific knowledge base in support of the technological requirements of the region.

Table 1. Course Offerings and Project Milestones

Fall '92	**Spring '93**	**Summer '93**
Order Hardware Refurbish Lab Space Hire staff Advisory Board Meeting Select First Project Develop Task Lists	Programming Lab‡ Problem Solving Lab‡ Impact Study Report‡ Data Structures Lab‡ Software Engineering Lab‡ Develop Functional Specifications	Programming Lab† Problem Solving Lab† Advisory Board Meeting Refine Task Lists Readiness Report
Fall '93	**Spring '94**	**Summer '94**
Project Design Lab Software Engineering Lab Programming Lab Problem Solving Lab Impact Study Report Data Structures Lab Advisory Board Meeting	Systems Engineering Lab Programming Lab† Problem Solving Lab† Impact Study Report† Data Structures Lab †	Programming Lab† Problem Solving Lab† Advisory Board Meeting Select 2nd Project First Iteration Report
Fall '94	**Spring '95**	**Summer '95**
Project Design Lab Software Engineering Lab Programming Lab Problem Solving Lab Impact Study Report Data Structures Lab Advisory Board Meeting	Systems Engineering Lab Programming Lab† Problem Solving Lab† Impact Study Report† Data Structures Lab†	Programming Lab Problem Solving Lab† Advisory Board Meeting Evaluation Report

9 Potential National Impact

Productivity is recognized as a key factor to remaining competitive in a global marketplace. In the Information Age, computing technology plays a critical and extensive role in the productivity of most businesses. Industry continues to increase its use of sophisticated technology and computer software technology in particular. Today's businesses expect reliable, easy to use technology and will reward those who provide solutions which are practical, effective and consistent with current or planned business practices. As Denning has recently pointed out, the most important criteria for the acceptance of technology depends not so much on correctness or algorithm complexity as customer satisfaction [4]. When this initiative is fully integrated into the computer science curriculum, it will serve as a model for an advanced computer science curriculum for the twenty-first century. The model includes the incorporation of an extensive laboratory infrastructure that effectively uses advanced education delivery technology to provide an increased use of faculty creativity in the delivery of the undergraduate curriculum. This model allows future students to be productive in the commercial business upon initial entry which is consistent with the required productivity edge for the information age.

10 Plan of Transfer

This initiative will provide major benefits to our students, local industry and government agencies. A measure of success would be the adoption of this initiative by other universities. The Department of Computer Science at Old Dominion University was the leader in the state of Virginia and the country in introducing formal closed laboratories into the undergraduate curriculum (1987). Many other universities have subsequently introduced similar formal labs into their curricula. The Department introduced the first undergraduate computer science minor in High Performance Computing (HPC - 1989). Several graduate programs now have a similar minor or track but few have a HPC undergraduate minor. The Department's commitment to undergraduate education earned it a six year accreditation on its first try.

The complete curriculum model (traditional ACM curriculum plus the Computer Productivity Initiative) can be transported into any functional CS undergraduate program with a suitable environment. The model's flexible structure allows for additional educational creativity to be applied and for the addition of future educational technologies. The transfer of the ideas of this initiative to universities will begin with the other universities within Virginia. One of the members of the CPI Advisory Board is a member of the State Council on Higher Education in Virginia (SCHEV). SCHEV is committed to transferring the successful ideas of this initiative within the state of Virginia. We also plan to disseminate the results of our experience at major education conferences (such as those run by the Software Engineering Institute (SEI)), SIGCSE sponsored conferences and through publication in technical and educational journals. A joint SCHEV/ODU-CS workshop is planned during the Summer of 1994 for all researchers in the advanced CS education, use of high technology instructional equipment, or large laboratory infrastructure computer science programs. We will present a tutorial at this workshop on the results and experiences of this initiative.

References

[1] H. Abdel-Wahab. Multiuser tool architecture for group collaboration in computer networks. *J. of Computer Communications*, 13(3):165–169, April 1990.

[2] H. Abdel-Wahab. Xtv: A framework for sharing x window clients in remote synchronous col laboration. *Proceedings of the IEEE TriComm '91*, pages 159–167, April 1991.

[3] CSTB. Scaling up: A research agenda for software engineering. *CACM*, 33(3):281–295, March 1990.

[4] P. Denning. What is software quality. *CACM*, 35(1):13–15, Jan. 1992.

[5] M. Dertouzos, R. Lester, R. Solow, and T. M. C. on Industrial Productivity, 1989.

[6] C. for Virginia Commission on The University of The 21st Century. The case for change.

[7] N. S. Foundation. Workshop on undergraduate computer science education, 1988.

[8] W. Scacchi. Managing software engineering projects: A social analysis. *IEEE Trans. S.E.*, SE-10(1), Jan. 1984.

[9] B. Simmons, D. Frailey, A. J. Turner, S. Zweben, and P. Denning. The scope and directions for computer science. *CACM*, 34(10):121–131, Oct. 1991.

[10] A. Tucker and B. Barnes. Flexible design: A summary of computing curricula 1991. *IEEE Computer*, pages 56–65, Nov 1991.

[11] D. Updegrove. Computing intensive campuses: Strategies, plans implications. *EDUCOM Bulletin*, 21(1), 1986.

[12] C. Wild, K. Maly, J. Dong, and G. Hu. A process model for decision based software development. *Proceedings of Conference on Software Maintenance*, pages –, Oct. 1991.

[13] C. Wild, K. Maly, and L. Liu. Decision-based software development. *Journal of Software Maintenance*, 3(1):17–43, March 1991.

III. Educational Facilities Support

1. Current Computing Facilities

The facilities which now support the undergraduate curriculum include three laboratories of Sun workstations managed by the computer science Department and a campus-wide network of IBM PC clones managed by the Computer Services Department. The Computer Services PCs support the Programming Laboratory and an occasional course in a specific programming language offered by the department. All other Computer Science courses are supported by the department's facilities.

The department laboratories consist of Sun Sparc workstations, 17 in the Advanced Projects Lab, 12 in the Problem Solving Lab, and 12 in the High Performance Computing Lab. The latter also includes two mini-super computers (a DAP and an NCUBE). Three laser printers (including one PostScript printer) are available to students.

All faculty and supported graduate CS students have Sparc workstations on their desks. Thus a total of about 100 Suns are interconnected via the department's network of fileservers. The department network is connected to the university backbone that provides access to university-wide computer resources including the university mainframe and various workstations managed by other university departments. Access to the university backbone provides the connection to the Internet and external resources for all CS users. Dial-up lines are freely available both through the department and through the dial-up facilities of Computer Services. See the network figure for details.

Current facilities are heavily used since they support the following courses:

- CS 251 Closed lab for CS 250 Problem Solving and Programming
- CS 301 - 308 Half semester courses in various programming languages.
- CS 350 Introduction to Software Engineering: major project.
- CS 355 Principles of Programming Languages: several small projects.
- CS 361 Data Structures: several medium sized projects.
- CS 450/550 Introduction to Database: several small projects or one medium project.
- CS 460/560 Computer Graphics: major project.
- CS 471 Operating Systems: major project.
- CS 476/576 Systems Programming: major project.
- CS 475/575 Introduction to Computer Simulation: several small and one large projects.
- CS 480/580 Introduction to Artificial Intelligence: several small projects.
- CS 484/584 High Performance Scalar Computing: several small projects.
- CS 485/585 Vector Computing: several small projects.

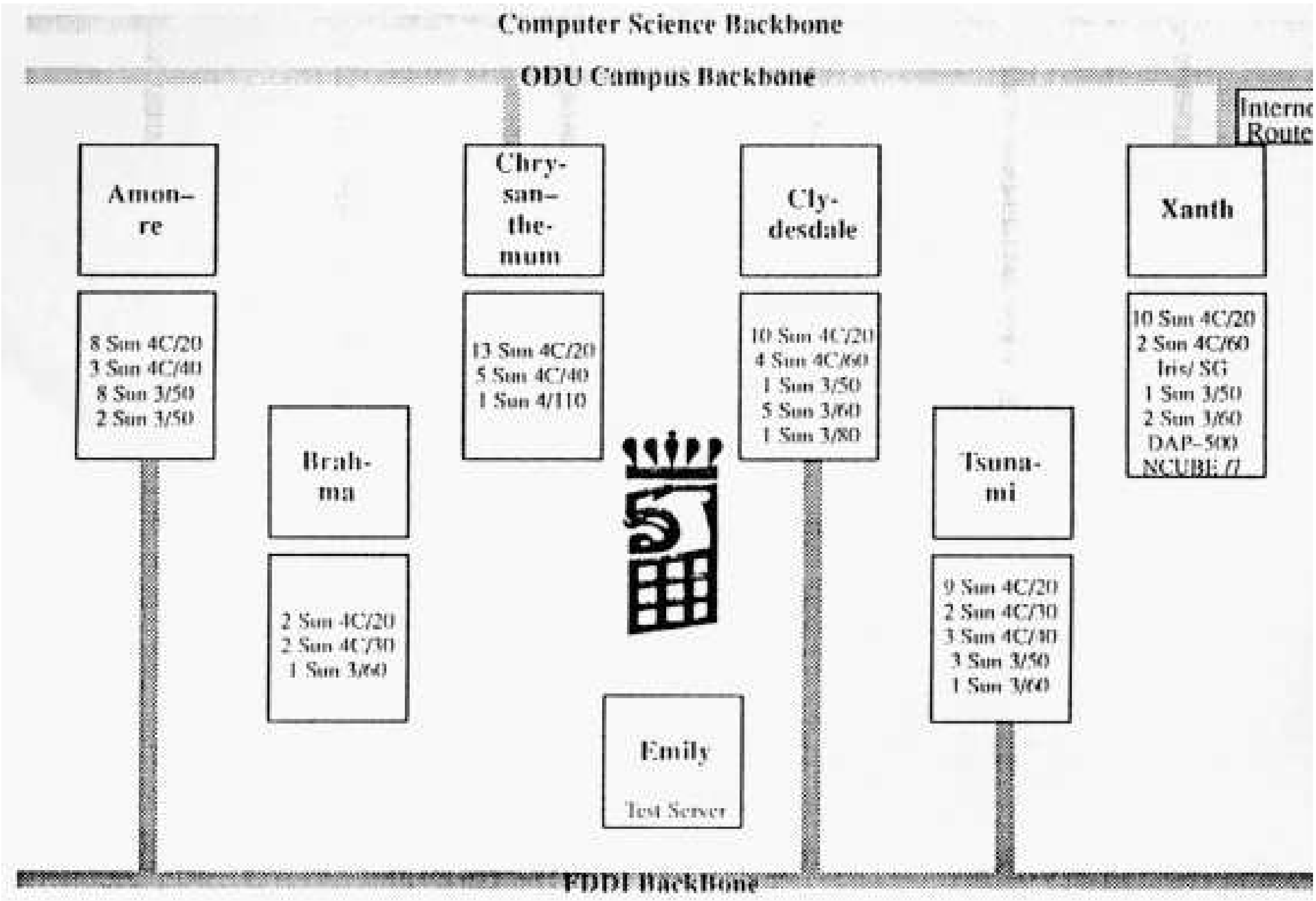

Figure 6: Existing Facilities of the Department of Computer Science

- CS 486/586 Introduction to Parallel Computing: several small projects.
- CS 488/588 Principles of Compiler Construction: major project.

In addition, the existing laboratories support documentation preparation for non-project courses as well as access to email, news, and other network and internet facilities for all computer science students.

Other Facilities: The Computer Science Department shares a building with the studios and mediated classrooms of Academic Television Services. These classrooms provide teleconferencing support via satellite or land line. They have been used for many national and world-wide teleconferences either produced by or participated in by ODU. These facilities can be used to connect our students with distant experts at relatively little expense, (with automatic video taping of the interview) and can be used by our students to edit video tapes they make of customer sites and interviews.

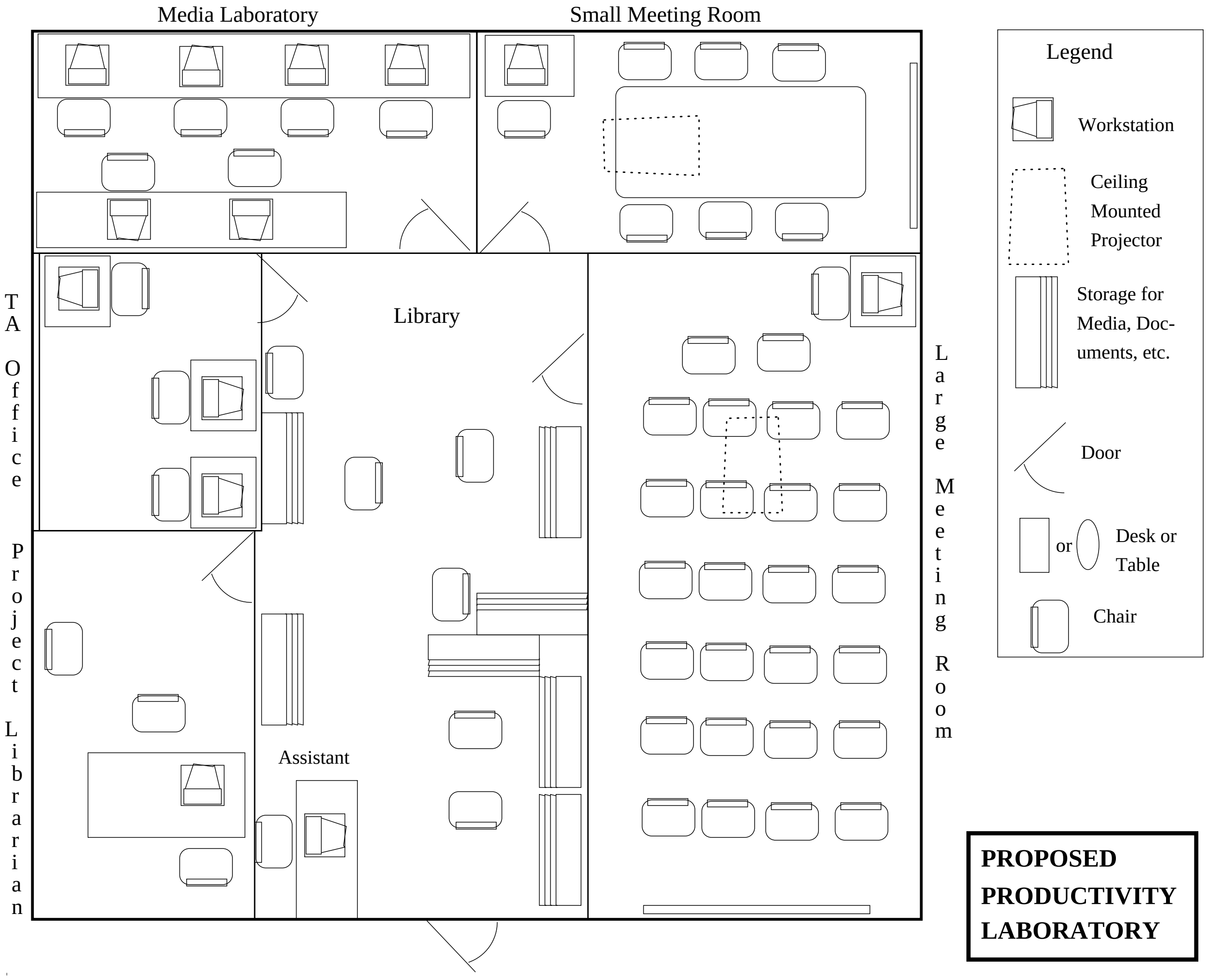

Figure 7: Proposed Productivity Laboratory

2. Required Facilities.

The CPI requires additional support in three main areas:

- Productive group interaction.
- Video technology.
- Access to computing.

Group Interaction: We propose to give the students the opportunity for productive group work and to train them in this area using three technologies.

- Electronic whiteboards. These are used as regular whiteboards but have the capability to save what was written on them for later use. This is an excellent medium for a brainstorming session or for exercises in group design. These systems will enhance the instruction through the effective use of this technology. One is required in each of the meeting rooms.

- Computer projectors. This technology allows the display of a workstation or a window thereon to be projected on a large screen for group viewing. This is an excellent tool for small or medium size groups for document reviews, code walk throughs, and presentations. When combined with video technology, the projector will allow the presentation of video sequences and hypermedia documents which incorporate video. This combination of equipment will provide the instructor with an effective instructional technique to describe actual screen or tv images as they occur. One is required in each of the meeting rooms.
- Video recorders and camcorders. These allow the review of sessions of group interaction by the instructor and the groups themselves as a learning aid.

Video Technology: The CPI makes heavy use of video technology to connect the "real world" and the campus. Experts lectures and customers' interviews are taped, archived, viewed, and incorporated into hypermedia documents using decision structure of DBSC and D-HyderCase. Students are taped in group activities and their performance critiqued. Presentations are composed on workstations and transferred to videotape for remote presentation.
Access to computing: The CPI will exceed the capacity of the existing computer laboratories. In order to adequately support the CPI, we will require 8 additional workstations for undergraduate access. These will all incorporate VideoPics cards to allow video tape to be displayed on a monitor. One of them will include a RasterOps card which allows presentations composed on workstations to be transferred to video.
Added Space: The university has pledged to add 1600 square feet of space in support of the CPI. The space will be used as follows.
Library: This room will also contain video transcripts, audio recordings, and paper documents in support of the Productivity Projects.
Media Lab: Six workstations equipped with VideoPics cards allow access to the video documents as well as other network facilities.
Small Meeting Room: Workstation, Projector and Electronic Whiteboard. Available to small groups (6-8 members) working on Productivity Projects.
Large Meeting Room: This room has similar facilities as the Small Meeting Room but has a capacity for larger groups (25 members).
TA office: Space for the TAs who support the Productivity Projects.
Project Librarian's Office: Space for the project librarian.

Equipment Worksheet

The rationale for the equipment listed in the Table III.1 (next page) is provided below:
Workstations: These are to equip the Media laboratory and the two meeting rooms. All will be equipped with VideoPics cards which will enable the viewing of video tape on screen. Taped segments can be incorporated into the hypertext requirements documents through the development of DHC.
Projector: These projectors are mounted in the ceiling and are able to project images under workstation control on to a large screen.

Video Equipment: The VideoPics cards are described above. The RasterOps card allows us to transfer images from workstation to videotape. Thus presentations can be prepared for customers to be viewed off campus.
Software: This software will complement the capabilities of DHC.
Network: Productivity Projects will require additional storage space, especially since video and audio files are to be created. Documentation will be heavily emphasized and so another laser printer will be required. The color printer will allow the preparation of more effective documents and presentations.
Electronic Whiteboard: These whiteboards capture what is written on them electronically for later use. This is an effective productivity tool for brainstorming and other meeting activities.

Table III.1. Productivity Laboratory Equipment

Equipment	total		year 1		year 2		year 3	
	Qty		Qty		Qty		Qty	
Workstations								
Sparc IPX	6	51,000	4	34,000	2	17,000		
SparcStation 2	2	24,000			2	24,000		
Memory expansion	8	8,000			8	8,000		
Projector	2	36,000	1	17,500			1	18,500
Video Equipment								
VideoPics Card	8	2,500	8	2,500				
RasterOps Card	1	6,000	1	6,000				
VCRs	3	1,000	3	1,000				
CamCorders	2	2,000	2	2,000				
Software								
Hypermedia	1	8,000	1	8,000				
CASE tools	1	12,000	1	12,000				
Network								
Router	1	2,000			1	2,000		
Gigabytes	4	9,000	2	4,500	2	4,500		
Tape Backup	1	4,500	1	4,500				
2nd Ether/SCCI card	4	1,000					4	1,000
Laser PS printer	1	2,700	1	2,700				
Color printer	1	10,000					1	10,000
CD ROM jukebox	1	20,000					1	20,000
Other								
Electronic whiteboard	2	20,000			2	20,000		
Total		219,700		94,700		75,500		49,500

Appendix

A New Course and Laboratory Descriptions

The format for the description of individual courses is as follows:

Summary: Brief description of the course. This description is given in terms of the input and outputs expected in each course emphasizing those aspects of the course which deal with the CPI.

Objectives: The educational objectives of this course as they relate to the Productivity Project.

Resources: The resources required to implement this course including laboratory space, equipment, human resources, Productivity Project resources (documents, speakers, customer site visits, etc).

Example: An example of the kind of activities to be undertaken in this course using the VIT project described in B as an example.

A.1 CS151: Introduction to Programming Laboratory

Summary: The Programming Laboratory is a closed computer supported laboratory that immerses students in a hands-on guided tour of the experimental aspects of solving small problems using a computer. Students receive coded programming examples, laboratory exercises, and programming assignments that are designed to provide them with the capabilities of solving a wide range of small problems using computer programs. For a Productivity Project they will receive carefully selected small programming problems that when completed add to the library of coded algorithms that support the requirements of the Productivity Project. The specific requirements and explanation of the assignments are products received from the upper tier laboratories.
Objectives: This laboratory course introduces students to not only basic programming concepts but also the intrinsic scientific observation, evaluation, and modification aspects of experimental computer science. The course is designed to assist students in mastering the fundamental programming skills required for completion of upper level courses. Specific course objectives include:

- The translation of program specifications into functional computer programs
- The development of the skills used in testing of small programs
- The development of the use of programming tools
- The development of the skills used with numerical calculations
- The development of fundamental problem solving skills
- The introduction to the top down problem solution design methodology

- The introduction to the bottom up programming methodology

The students who successfully complete this laboratory will have gained a working knowledge of the Pascal programming language that is sufficient to solve a wide range of small problems. Many of the small problems that are solved during the semester will be directly linked to requirements of the curriculum project.
Requirements: This laboratory will require the use of the available video tapes or other Productivity Project information that will assist students in understanding the programming environment for the small programming assignments given to them. The availability of this resource virtually at any time will provide students quick answers to questions and provide a greater level of productivity in their programming efforts.
Example: A typical VIT requirement is the development of a list of containers to be loaded onto a ship. This list is composed of multiple orderings, by destination, by weight, by size. This is a typical programming task that is required by the project and is a suitable programming assignment for the Programming Laboratory. The task contains the elements of multiple sorting of array structures, a fundamental programming problem for the Programming Laboratory.

A.2 CS251: Problem Solving Laboratory

Summary: The Problem Solving Laboratory receives a brief description of a broad problem statement relating to a subsystem from the Project Design Laboratory, to motivate its problem solving component. It also takes test modules, testing plans, and design documents from the Software Engineering Laboratory for black box testing. The assignments are supportive of furthering the students' programming skills. The major output of this laboratory are test results and performance measurements that will be returned to the Software Engineering Laboratory.
Objectives: The course aims at both problem solving and at a further development of programming skills of the students. With this in mind, the course objectives include:

- Training in problem solving methodology in general
- Training in analytical skills
- Training in generating candidate solutions
- Training in evaluating proposed solutions
- Training in software testing
- The furthering of their programming skills
- Training in performance measurement and evaluation

Requirements: The course requires a type of meeting facility on campus that would be supportive of communication among all the members of the class. This requires facilities for group viewing of documents, designs, videotapes, etc., and for recording the product of group interaction: notes, sketches, and the like on a whiteboard. This is Large Meeting

Room. Within this course we plan to use XTV as an instructional tool. In addition, a SUN laboratory is needed to perform testing and the performance measurement segment of the objectives of this course.

Example: Assume the VIT project has been selected. The following is an example of what might happen in the problem solving segment of this lab. A problem such as "Improve the container loading operations at a marine terminal" is given to the students. The students need to understand specific activities that pertain to loading including the kinds of people involved, the specific activities: cranes are used to load containers, people communicate via a combination of hard copy, radio and hand signals.

With an understanding of these preliminaries firmly in hand, the students analyze the given problem and pinpoint areas for possible improvement: for example the communications system of the terminal, automation of the entire loading operation etc. They then select one problem to work on and define it precisely, say improving loading productivity by improving communications among the people involved in the process.

The next stage involves generating candidate solutions, typically, by brainstorming. One such solution might involve providing the crane operators with additional radio equipment, for example. This candidate solution will be analyzed for feasibility/implementation, advantages and disadvantages. It is possible that no suitable solution can be found. In that case either more candidate solutions must be sought or the problem must be analyzed again and a new problem must be defined.

If the proposed solution is acceptable, the solution must be presented to the class using multi-media forms.

A.3 CS300: Computers in Society - Impact Study

Summary: Students in this class will use the productivity project theme as the focal point of their impact study report. The specific areas of the project to be evaluated will be received as tasks from the Project Design Laboratory. The impact study report is a semester long effort of detailed research into the obvious and subtle impacts of using computers in the customer's operation.

Objectives: The objectives for this course are to expose the students to the dimensions other than technical computer science issues which need to be studied when designing system and to have students apply knowledge from one of these dimensions to the cash project at hand. Specific course objectives include:

- Learn about a broad set of issues (ethical, legal, economic, social, man-machine operation interface) raised in the use of computer technology (other than technical issues) in various application areas (business, government, education, manufacturing).

- Develop system scenarios describing different ways to integrate computers, machines, operating procedures, humans to achieve the same external functionality.

- Develop a set of adverse problems and expected benefits for same issue in the chosen task.

Requirements: This course has a large number of non computer science students. The tasks from the Project Design Lab will be given only to the subject in order that the TV class room can be used for interviews with domain experts. Speakers who will be invited to address the above mentioned issues will give their talks to the entire class which implies a large auditorium for regular use. Many of the research questions raised will be worked on in groups of ten to twenty (the expected number of students working on a given impact study). The large meeting room of the CPI has to be available at least once a week for up to five groups.

Example: In the VIT project the key issue is the relation between the union and the management on the question of automation. The Project Design Lab requests a study on "Automation" and "Union" in general and wants details on the interaction of longshoremen, crane operators, stevedores and terminal management at VIT and how it should be implemented by automating computer loading and unloading.

A.4 CS350: Software Engineering Laboratory

Summary: CS 350, Introduction to Software Engineering, involves students in a group project for a relatively large project and presents issues and solutions related to developing and managing the development of large software systems. Much of the material presented in lectures is of immediate utility to the student and is applied in the laboratory using the term project. The flavor of the class is very much that of "best known practices;" for example, while the student should come away with an understanding of the problems inherent in the effectiveness of testing, the student should also know "best known practices" for testing code. Each student has several responsibilities derived from the group project, both related to group support activities and project development responsibilities.

Objectives:

1. Expose students to a set of "best known practices" for specification, analysis, verification and validation to reduce life-cycle costs.

2. Have student experience dynamics of working in a small group. Each group delivers a complete project.

3. Detailed exposure of student to at least one reasonable technique for dealing most steps of software life cycle. The purpose is less breadth (e.g., presenting several alternative techniques), but more depth in a few carefully chosen techniques.

4. As currently presented, heavy emphasis is placed on the on review process (e.g., requirements, design, test plan) and on unit and integration testing.

5. Each student develops at least one detailed design for a module. The student presents and defends one of his/her detailed designs to the project team.

6. Each student develops at least one test plan for a module. Components include: A test driver for the module under test, stubbed versions of other modules with which this module interacts, a set of test data (in the form of input/output pairs) with documentation on each test case justifying the test case and the choice of data used in the test case. The student presents and defends one of his/her test plans to the project team.

Resources:

1. The requirements and implementation schedule for the course project is generated by the *Project Design Lab* and must be approved in the semester prior to the offering of the Software Engineering Course by the Supervisors and Instructors for the Project Design Lab, Systems Engineering Lab and Software Engineering Lab. Project deadlines must be realistic and on a weekly basis. The course instructor will act as an experienced project manager who can effectively understand the difficulty of each assignment made to each student. Since mistakes (in requirements, in high level design, and in effort estimates) are inevitable in a project of any size, the project manager must also be available to make modifications quickly.

2. Access to D-HyperCase specifications prepared by the Project Design Lab.

3. Facilities for small group presentations; materials and tools which students can use to prepare for presentations (e.g., overheads) for group presentations. Videotaping facilities so that each student can review his/her presentation.

4. Rooms for conducting team meetings with computing facilities, printers, work tables, marker boards and small lockable storage areas for each team. Teams can reserve a room for a block of time.

5. Project management and module design software for Sun workstations.

Example: The design and implementation of a prototype for the graphical user's interface for the crane operator is an example from the VIT project of the kind of subsystem which could be implemented in this course. This prototype would then be used to illustrate project plans to the customer (VIT). A second example is the placement of the ship's container cargo to reduce handling during various ports of call.

A.5 CS360: Data Structures Laboratory

Summary: The Data Structures Laboratory takes as input from the Software Engineering Laboratory design specifications for various abstract data type modules, along with performance profiles for the key operations involved in these modules. The Data Structures Laboratory produces working implementations of the data types, together with complexity analyses of the atomic operations on the abstract data types.
Objectives: The course aims to provide the student with the experience of development of abstract data type modules and with the analysis of the complexity of the operations implemented. Therefore the course objectives include:

- To provide hands-on and supervised laboratory experience in implementing various generic and applied data structures; this closed mode of operation has to be contrasted from the present mode where the data structures are discussed (in an abstract manner) in the class room and the students are left alone to perform the implementation in the form assignments.

- To give the students a broader perspective of applying the classroom work towards a solution of a large practical problem.
- To give the students some exposure to decisions involving the choice of a data structure most suitable for the problem at hand.
- To implement abstract data type modules that will be useful not only in the project proper, but also in subsequent courses such as algorithms operating systems, compliers, etc.
- To analyze the complexity characteristics of an implemented operation.

Requirements: The students in this course will be given a written description of the intended use of each abstract data type module to be designed and implemented. The students will then devise the specific way in which the modules will work, while documenting the interface. The innards of each module should be transparent to the outside world. The course requires two types of meeting facilities. One which enables small working groups to be productive in the development of the abstract data types, another which facilitates communication among all the members of the class. Both require facilities for group viewing of documents, designs, videotapes, etc., and for recording the product of group interaction: notes, sketches, and the like on a whiteboard. These are the Small Meeting Room and the Large Meeting Room. For brainstorming needed in the design stage we plan to use the XTV collaboration facility.
Example: Assume the VIT project has been selected. A typical set of abstract data types needed in such a large project have to do with the study of stacks and queues, lists and applications, trees and their uses, minimum spanning tree modules, graph exploration techniques and algorithms.

More specifically, assume that the class has been tasked to produce an abstract Dictionary data type to be used in tracking loading and unloading of vessels from all over the world. The principle decision for the students in this class is to make the right choice of the underlying data structure supportive of the intended use of the dictionary in actual operation. Students will have to trade ease of implementation for speed or vice versa. They will also be concerned with the amount of search that is to be performed in the dictionary, with how dynamic the dictionary turns out to be.

Once these decisions have been made, assume that the students decide to use a 2-3 tree to implement the Dictionary data type. They will have to design all the operations that for the outsider as atomic. The interface needed for a correct use and parameter passing have to be fully described and documented. Furthermore, the students have to provide analyses of the average (or worst case) running times of the key operations on the data structure.

A.6 CS400: Project Design Laboratory

Summary: The input to the Project Design Laboratory is a partial system description (at the beginning of the cycle this will simply be the description of the Productivity Problem, later on it is the current solution as produced by the previous Systems Engineering Lab). Students analyze the remaining, yet unspecified system and search for alternate solutions

which can make the system users more productive. They will develop a set of issues and plans on how to resolve them. Specifically, they produce a task list for the Impact Study and define software components for which they want prototypes or simulations. Once the components are identified, students will develop management plans and specifications for the production of these components in the Software Engineering Lab. At the same time another group of students take previously produced software tasks (specifications and implementation plans) and supervises the production of these plans in the Software Engineering Lab.
Objectives. The course aims to provide the student with the experience of development of project requirements in a state-of-the-art setting. Therefore the course objectives include:

- Training in effective group collaboration.
- Training in interviewing techniques.
- The development of issues and plans for resolving them.
- For software components:
  - The preparation of requirements definitions and specifications.
  - The development of architectural designs and detailed specifications.
  - The development of plans for implementation, verification and validation.
- Experience with CASE tools and IPSEs including D-HyperCase.
- Experience with computerized tools for collaborative work (such as XTV).

Requirements: The course requires two types of meeting facilities on campus, one which enables small working groups to be productive in the development of designs and specifications and another which facilitates communication among all the members of the class. Both require facilities for group viewing of documents, designs, videotapes, etc., and for recording the product of group interaction: notes, sketches, and the like on a whiteboard. These facilities are the Small Meeting Room and the Large Meeting Room.

In addition, facilities are required for viewing videotapes, editing them into hypermedia documents, and viewing the latter. This is the Media Lab.

Finally, printed and taped documentation will be stored in the library. In addition, the students will need access to a variety of research materials. We expect that these will be available on CD-ROM and that a CD-ROM jukebox will make them available anywhere on the network.
Example: Assume the VIT project has been selected. For this cycle, the students in this class are assigned the planning subsystem. They must understand the work of the stevedores in planning as well as what actually occurs during the loading and unloading of the vessels. Project management has obtained the cooperation of VIT personnel. Several teams of students go to the site and tape interviews with stevedores; current stevedoring planning practices; loading and unloading of vessels; interviews with VIT management discussing their plans; views of the storage yard for containers; explanations of yard inventory control.

The tapes are edited in the Academic Television Services studios and kept in the project library. The problem is studied by class teams who prepare requirements documents using

DBHC and other software. These documents are reviewed by the teams in the small meeting room and by the whole class in the large meeting room. A videotaped version of the requirements documents is prepared for viewing by VIT customers at ODU or VIT.

Once this section of the project has an agreed on document, the class develops a management plan for the next semester's work, detailing the subsections to be designed, establishing deadlines, and determining an appropriate management structure.

The requirements document, other documentation, and management plan are preservered in the library for next semester's use.

A.7 CS401: Systems Engineering Laboratory

Summary: The main task of the Systems Engineering Lab is the generation of the systems specification. This specification includes not only functional system requirements for hardware and software but also functional requirements for non-computer hardware (such as cranes, sensors, radios) as well as users manual, cost/benefit analysis, performance analysis, impact studies, prototype evaluation and a project management plan. In order to generate this set of documentation, students must analyze the studies commissioned during the previous semester by the Project Design Lab. The "customer" should be kept abreast of the results of this analysis through periodic briefings and experience with the software prototype generated in the previous semester.

Objectives: The major objective of this course is to analyze and integrate the results of previous studies and to generate a coherent and comprehensive systems design document. To reach this objective, the following must be met:

- Systems Problem Solving involving the ability to tradeoff concerns along several diverse dimensions.
- Progress report to the customer which summarize previous studies.
- Ability to communicate with the customer in order to resolve critical decisions.
- Development of a Project Implementation Plan which would form the basis for eventual commercialization of the solution.
- Practice in technical writing.
- Delivery of Project Documentation including the User's Manual and other High Level Project Documentation.
- Experience in group decision making using computer assisted decision support tools.

Resources: The class will be divided into teams, each tasked with a specific responsibility (such as the generation of the user's manual, analysis of the cost estimates). This requires Large Meeting Room for periodic progress reports to the entire systems engineering group (approximately once every two weeks). The individual teams will meet frequently (twice each week) in the Small Meeting Room. Students will need access on demand to the Library and Media Room and DHC. Several visits with the customer during this class are also required.

Invited speakers and domain experts will help to resolve critical decisions. Video tapes will be used as training aids to improve oral presentation skills. Video tapes will also be used to document the prototype for the customer.

Sample Activities: Demonstration to VIT of a prototype graphical interface for the crane operator. Report on the cost and benefit of computerized placement of cargo. Development of a plan for the phased implementation of technology for ship loading/unloading. This plan will divide the implementation into a number of phases, analyzing for each phase the costs and benefits, including capitol expenditures, training costs, etc.

B Sample Project: Automating Ship Terminal Operations

B.1 Introduction

The success and growth of container shipping in Hampton Roads is due in large part to the ability of the port to provide efficient unloading and loading facilities and operations. Despite the inherent standardization of containers and crane operations, most aspects of container cargo handling in the United States in general and Hampton Roads in particular continue to be handled without the benefit of modern computing technology.

The port is managed for the State of Virginia by Virginia International Terminals (VIT). This private company maintains the port facilities which consist of piers, cranes for transferring containers, trucks for moving containers, a yard for storing containers, and a rail terminal. VIT operates the yard and maintains the other facilities. Its income is derived from leasing space in the yard and the use of its other facilities to shipping companies.

It is important to note that VIT is not in a position of authority with regard to the shipping companies who lease its facilities, the stevedores who plan and manage the shipping companies' operations, or the unions who provide the labor. The shipping companies are VIT's customers; the stevedores work for the companies, not VIT; and the union works under a negotiated contract. Hence VIT must implement changes using persuasion and negotiation, rather than management fiat.

Our customer is VIT, which would like to automate the port in order to reduce costs. We are asked to consider the cargo transfer operations, for which VIT envisions a distant future somewhat as follows:

B.2 Hypothetical fully automated system.

A shipping company would notify the terminal in Hampton Roads by computer of the approximate arrival time of a container ship, including a list of all containers, their locations, and destinations for the terminal database. The local branch of the shipping company would enter information concerning the containers to be loaded from Hampton Roads, their destinations, and their times of arrival at the port or their positions in the yard.

An optimizing program would formulate a plan for the simultaneous loading and unloading of cargo at the port, reducing the cargo transfer time and distributing the containers to be loaded so as to reduce unloading time at the ship's later ports of call. A faster transfer means that the same dock and terminal facilities can handle more ships and more containers and that the shipping company can get more use out of its ships if they choose an automated facility over a manual one.

In this ideal scenario, the cranes are robots which carry out the optimized cargo transfer plan which they retrieve from the database. Every container movement is recorded in the database in case an error is discovered. Such errors include containers which have been misidentified or misplaced during loading at prior ports or by clients of the local branch of the shipping company. Should they be discovered, the cargo transfer plan must be revised on the fly to compensate.

As the container ship departs, the new configuration information is supplied to the shipping company and to the next port of call.

B.3 The Current System.

The current, unautomated method of cargo handling has neither robots nor central database nor computer-based planning nor computer-directed cargo handling, although the container storage yard (the "stacks") are managed by computer and some stevedores use computers in some stages of their planning. A typical scenario begins when containers bound for Hampton Roads leave, say, New York, on a container ship which also carries containers destined for other ports. A distribution of containers to be removed in Hampton Roads is then FAXed by the shipping company some 10 to 12 hours prior to the ship's arrival. A similar listing of containers to be loaded is not finalized until about an hour prior to the ship's arrival. During this time the stevedore prepares plans for the efficient unloading and loading of containers, processes which may utilize several cranes. Finally, these plans are communicated to crane operators and other dock personnel via a combination of hard copy, radio and hand signals.

B.4 Intermediate System.

The customer, VIT, is in no position to impose a fully automated system or even negotiate one with unions and stevedores who fear a loss of jobs and responsibilities. Instead they wish to develop an intermediate system which will proceed incrementally. Their initial aim is to provide computerized assistance to all of their clients who can benefit from it. After accustoming the clientel to such assistance, VIT plans to make the assistance smarter and smarter, thereby increasing the productivity of all concerned. The customer suggests beginning with the following four projects:

- provide a graphical display to the crane operator of his work
- provide computer assisted planning for the stevedore
- develop capability to locate and read container numbers automatically from containers on ship or on chassis using a video camera
- develop EDI (electronic data interchange) capability for VIT mainframe

These goals are interconnected since the stevedore's plan will be available electronically to the program which displays the work to the crane operator; the automatic recognition of container numbers can be built into the crane operator program for the sake of error detection; and the crane and stevedore programs will need information from the mainframe and will have information to provide to it. The first two items are developed below.

B.4.1 Dockside Communication System Requirements.

The computer-aided communication of cargo transfer plans faces a number of severe requirements since the system will provide assistance to personnel who operate the crane, who coordinate the crane's movements with the arrival of the chassis which carry the containers to be loaded and discharged (dock foremen, checkers), and who supervise the entire operation which requires the use of several cranes simultaneously (stevedores). The system should also interface with the computer system which manages the "stacks" where containers are stored awaiting shipment.

The hardware in the crane operator's cabin must be engineered to withstand the rigors of vibration and acceleration. It must provide its information clearly in all daytime and nighttime lighting conditions. The hardware on the dock must be fully portable, perhaps using RF communications, and capable of operation in the moist, salty atmosphere of the docks and designed to withstand some hard knocks.

The software must communicate to the crane operator the general plan for the discharge and loadback of the bay he is working as well as indicating the locations of the next container for discharge or of the destination position of the next box to be loaded. It must identify "shifts" (containers not destined for this port but which must be moved to access those which are) and indicate their new locations on ship or temporarily on the pier. Finally it must display the temporary location of the hatchcovers which must be moved to access below deck containers.

Generally, this module communicates with the crane operator and the supervisory personnel by showing the next crane movement, updating displays when the movement is sensed, and communicating with the stacks database. When some problem arises, the supervisors must be able to smoothly revise the cargo transfer plans on the fly. In order to do so, the software must make available information concerning the containers so they can make informed decisions in replanning the transfers. This information includes container identification number, shipping line, destination, hazardous materials code, etc.

In general the dock foremen and the crane operators have no experience of using computers in any way and have perhaps no desire to do so since in their minds they have done perfectly well without them until now. In addition these personnel may present problems to the designers for a number of reasons: they may have minimal levels of literacy, color blindness, or even hostility to management. The stevedores may have some computer experience. Indeed several already use them. Others may feel there is no need to automate anything.

B.4.2 Stevedore's Planning Assistant.

The planning assistant software is to be an aid to the stevedore in planning the cargo transfer operations. In its simplest form it would allow the stevedore to identify the containers to be discharged and to assign locations to those which are to be loaded. It must incorporate a variety of ways of entering data since some data is transmitted electronically (but in no standard format), other data resides in the "stack" inventory computer, while still other data must be entered by hand.

B.4.3 Complications for the programs.

The programs must take into account the many different configurations of the ships, the fact that containers come in a variety of sizes, the fact that some are shifts, the variety of data formats, the variety of personnel slated to use the software, interfacing with the crane (which generates useful electronic signals), interfacing with the "stacks" computer, the use of several cranes at once to transfer the cargo of one ship, the changing of plans on the fly, and a host of other factors. In addition, the system must provide printed documentation for use in case the computer system goes down.

B.5 Automating The Current System.

The change of operation from the current method to a more automated approach can only occur gradually as technologies are developed and proven to the various constituencies of the system shipping companies, stevedores, longshoremen, crane operators. Each increment in the change-over must be worthwhile to them as well as a step in the direction of increased productivity. The intermediate system described above has benefits for all constituencies:

- Communication with crane operator. The graphical display in the cab unambiguously pinpoints the next operation to be performed and avoids time lost due to the current combination of loudspeaker and hand signals. The resulting improvement in speed of operation is desirable to the operators (who receive a productivity bonus), stevedores, shipping companies and VIT.

- Automated planning. The planning assistant is obviously a benefit to the stevedore. But it is also a benefit to VIT to replace manual methods with electronic data transfer and a central database. The acceptance of the planning assistant opens the door to future development of more efficient cargo planning in the future.

- Flexibility. Since planning (and replanning) time is reduced by use of the planning assistant, the port can be more responsive to users' needs.

- Container tracking. Currently a database tracks the location of containers awaiting sea or land transfer in the "stacks". A container transferred from ship to the stacks is entered into the stacks database only after it crosses a physical entry point where it is manually entered on a paper form. Integration of the planning process with the stacks database would allow direct electronic entry of the information into the stacks database. This would also allow quicker tracking of the containers.