3 PROJECT SUMMARY

Beginning in 1991 the University of Virginia renovated its undergraduate computer science curriculum. One central feature of these innovations was the development of closed laboratories of four core courses. The closed-laboratory components of these courses use a variety of organization and techniques, including prescribed exercises, collaborative learning, artifact analysis, studio-style presentation, and the use of projects based on interesting hardware devices such as computer-controlled cameras and robots. The laboratory components of these courses have been an extremely successful part of our curriculum, but we plan to redesign and reimplement the laboratory components of these courses to introduce new innovations and to address certain problems characteristic of the closed-laboratory approach.

Rapid advances in Web-based technologies and increasing ownership of networked personal computers by students makes it even less sensible to use closed laboratories to deliver exercises or tutorial activities. Many institutions are facing increasing enrollments, shortage of space, or reduced budgets for laboratory assistants that require departments to insure that resource-expensive closed-laboratory components for courses are used as wisely as possible. Our desire is to re-examine the strengths of the closed laboratory model and assess methods and technology for learning outside a closed laboratory in order to deliver better practicum experiences for students and to reduce a department's costs for developing and delivering closed-laboratory components in a computing curriculum.

The project we propose will:

- analyze the needs for a closed-laboratory component for a course and also for the set of student activities carried out during a course’s weekly laboratory sessions;
- determine what practicum-oriented student activities are best served in a closed laboratory environment and which can effectively be handled outside a closed-laboratory session;
- identify and document a set of best-practices of techniques and tools that can support effective learning activities for closed-laboratories, and also for activities that are better moved to an open-laboratory situation;
- use these set of best-practices to define a development guide for how to plan and implement laboratory components for a set of courses in a degree program, and to refine and evaluate this development guide by renovating the laboratory components of several courses in our program;
- create a number of new innovative laboratory resources that (based that our experience) require and benefit from a closed-laboratory learning environment;
- export both the development guide and the new laboratory resources to several other universities, in order to evaluate their effectiveness and refine their implementation, and to demonstrate that our efforts can be adapted and benefit a large number of universities.

Evaluation of results and dissemination of artifacts will be in cooperation with James Madison University in Virginia and Florida Atlantic University.
4 PROJECT DESCRIPTION

RESULTS FROM PRIOR NSF SUPPORT

- NSF Award 9981071: Prototyping a Software Engineering Educational Community
  PIs: Thomas B. Horton and John C. Knight

  This CCLI EMD proof-of-concept grant focuses on developing software engineering education resources that can be shared between universities. In particular, the goal is to allow students at various universities control expensive and interesting hardware devices like cameras and robots across the Internet. Work was begun when Horton was employed at Florida Atlantic University, and we are now developing relationships with other universities while continuing collaborations with FAU. Current activities focus on exercises that use a networked webcam; we plan to attempt to expand this idea to allow other universities to access the Nomad Scout robots used in Virginia’s CS340 course. Another goal, to develop an active community of software engineering educators, shows promise as we are now collaborating with the PIs of NSF award 0080502, a project with similar goals. We have not yet prepared publications from this work.

- NSF Award 0080502: Modern Development Practices in Undergraduate Software Engineering Courses
  PIs: Thomas B. Horton, Neal S. Coulter, and Robert B. France

  This ILI equipment grant provided funds for a personal computers and software for a laboratory at Florida Atlantic University to support software engineering courses. The lab was established and serves two upper-division software engineering courses (including a software project course adapted from Virginia’s course CS340). Several specific exercises related to prototyping, requirements modeling, etc. were developed. Though the laboratory made a significant impact at FAU, the work has not yet produce any publications (though a paper on studio laboratories based on the project courses at FAU and Virginia is in preparation).
PROJECT DESCRIPTION

Goals and Objectives
Reflecting upon our experience to date with closed laboratories, we have concluded that a number of enhancements in the basic technology are possible. Our goal with the proposed project is to develop the next generation of laboratories in computer science. The specific objectives we have are:

• Development Cost Reduction.
  We propose to refine the approaches that we have developed to reduce the initial effort required to develop laboratories.

• Operational Cost Reduction.
  We propose to enhance the techniques used in laboratories to reduce their recurring operational costs.

• Expansion of Laboratory Concept to Non-laboratory Courses
  We propose to expand the range of courses that employ laboratories by exploiting our experience to date and adapting techniques to courses that presently use traditional educational methods.

• Novel Laboratory Concept Development.
  We propose to develop novel new laboratory concepts based on the ideas that we have evolved—in particular we propose to enhance the studio presentation and artifact analysis laboratory styles.

  This project is largely education research. To achieve our goals it will be necessary to make significant experimental changes to laboratory styles and carefully assess performance and effect. The impact outside of the University of Virginia will be extensive and include basic results that others can exploit and courseware that others can adopt.
Detailed Project Plan

Background
In 1991, the Department of Computer Science at the University of Virginia began a complete renovation of its curriculum. The activity was undertaken because the then-current curriculum at Virginia did not prepare students well for either industry or graduate study. As part of the renovation, a new curriculum philosophy was developed [6]; a strong emphasis was placed in the new curriculum on excellence in software development and mathematics. In the initial activity, seven new core courses were developed. In a later, second phase of enhancement an additional four upper-level courses were developed. Both major phases were funded by the NSF.

A central feature of the curriculum that has been developed is the incorporation of a variety of different closed laboratories; initially in four of the core courses (in part with the support of a NSF DUE USE 91-56112 grant) and subsequently with several other courses. By a closed laboratory we mean an exercise activity that meets at an assigned time and place, and with a set agenda to be performed by the students[2, 5]. The simplest analogy for the concept is the familiar type of student laboratory exercise in physics and chemistry where students undertake some form of appropriate activity at a fixed time each week (usually) in a specially-equipped, dedicated facility.

At the University of Virginia we have a specially-equipped, dedicated facility for laboratory exercises in Computer Science. The room is organized into a set of small work areas each of which has several networked workstations. The workstations have a variety of specialized software installed as well as all the usual software services and tools. To support certain laboratory activities, some specialized hardware is installed in the facility (see below).

Four different styles of closed laboratory are used at the University of Virginia:

- Prescribed exercises.

Some laboratories, especially those in the introductory course, follow an agenda that is quite common, namely students work through a prescribed sequence of small interrelated steps that are checked off by a staff member as each step is completed. Help is immediately available from the staff.
• **Artifact analysis.**

Understanding artifacts (software, documents, etc.) prepared by others is a crucial skill in industrial development, and it is difficult to teach. Artifact analysis involves the examination of certain assigned and carefully selected artifacts with a view to developing suitable comprehension and analysis skills.

• **Studio presentation.**

In other design disciplines, such as architecture, students benefit from studio settings for exhibiting their designs to and being critiqued by their peers. We have developed a variant of this approach for use in computer science in which students present their solutions to a given problem and discuss them with their peers and instructors.

• **Case studies.**

Many innovative teaching approaches in other disciplines make extensive use of case studies—the systematic study of existing artifacts. This approach is utilized in our curriculum by providing the students with, for example, the design documents and source code for a system that has special or desirable characteristics and engaging in an analysis of it.

Through these four different laboratory styles are woven three important pedagogical concepts that make the laboratory activities substantially more effective. These concepts are:

• **Specialized Hardware.**

Specialized hardware items in the laboratory exercises enhance the laboratory greatly because they present diverse challenges to the student and add immense interest.

Two examples of the specialized hardware that is used are shown in Figure 1. The two devices shown are: (1) a TV camera mounted on a pan/tilt head that is controllable from the network; and (2) a mobile robot equipped with a TV camera on a pan/tilt head, a TV transmitter, and a radio-Ethernet connection. These devices are used in two courses on software development (not robotics or image processing), and they have proven very significant pedagogically. The use of such devices is only possible using the closed laboratory concept because the labora-
Group Activities.

As is common in Computer Science programs, extensive use is made of groups in our curriculum. However, the interaction between the use of groups and the use of closed laboratories is quite extensive. Group activities involve multiple students in a particular laboratory exercise and this provides an experience that is much closer to modern industrial practice because of the laboratory timing and location. Group activities are also used to link sets of weekly laboratories into sequences—some sequences are just two weeks long and some are a complete semester long.

Undergraduate Teaching Assistants.

Many undergraduate students are hired as teaching assistants for the closed laboratory portion of our courses; they are responsible for interacting with the students in the course in the same manner as the instructor and graduate teaching assistant. They meet with the teaching staff, prepare for the laboratory activities and aid in the continuous updating and improvement of the course.
The third year course in our curriculum entitled “Advanced Software Development Methods” (CS340) is an example of the interplay between the different laboratory styles and these three concepts†. The goal of this course is to introduce a number of concepts in software development including extensive documentation, project management, and software processes. The course involves a semester-long project to build a control system for a mobile robot of the type shown in Figure 1. The students are organized into groups of four and they stay in the same groups for the entire semester. The groups are paired into teams with each group in a team developing part of the system.

The course includes 12 laboratories of which nine are studio presentations to the entire class and three are prescribed exercises. The studio presentations are not graded and involve topics such as specifications of the robot system. During a typical studio laboratory in this class, one group of students presents their solutions to a specific technical challenge, such as developing a specification, and the solution is examined and discussed by the instructor and the rest of the class. This is then repeated thereby allowing all students to see several solution efforts. In a large class there is not sufficient time for all of the groups in the class to present their work so a set of groups is selected at random for presentation. All groups have to be prepared to present. Our experience has been that this approach works extremely well. In a class with 30 groups of size four, for example, presentations by five or six groups during a laboratory usually reveals all the major problems and allows several candidate solutions to be reviewed. Following the laboratory, a solution to the problem has to be developed that is graded, but this can be done with the extensive experience gained during the laboratory.

The use of the specialized robot systems in this course introduces many challenges that arise typically with real hardware including scarcity of equipment, equipment breakage, communications delays, finite acceleration times for electric motors, erroneous documentation, and so on. It would be virtually impossible to create these challenges artificially.

†. For further details of this course, see the Web site http://www.cs.virginia.edu/cs340
Laboratory Experience
After a decade of experience with the use of laboratories in computer science courses, we are convinced that they offer tremendous benefits in undergraduate education. By expanding beyond the most common form of laboratory (the prescribed-exercise style) we have realized a number of additional benefits.

One of our most important observations is that success with undergraduate laboratories arises from the fact that teaching computer science is often best achieved using a master/apprentice model. Much of computer science has to be learned by experience because there is no simple distinction between good and bad. In design, for example, there are often many good designs for a particular application and choosing one is a matter of judgement. Similarly, the reason why a design is good is often very subtle and cannot be learned easily from isolated study or even conventional lectures.

The apprenticeship paradigm appears in all four of our laboratory styles. In a prescribed-exercises laboratory, the instructor selects the exercises to make a point and this documents the instructor’s experience for the student. Then it is interaction with staff during the laboratory that really completes the educational experience. In a studio presentation, the discussion with other students and the instructor is very much in the apprenticeship model. The other two styles of laboratory that we use are obviously related closely to apprenticeship.

The fundamental master/apprentice paradigm as it appears in closed laboratories is illustrated in Figure 2. The role of the master in this paradigm is to create artifacts that present canonical issues or examples for the apprentice, to respond to the apprentice’s questions, and to examine the work of the apprentice. The role of the apprentice is to access the artifacts and perform whatever assignment has been set, submit questions to the master to ensure understanding, and deposit modified artifacts for examination by the master. The thing that distinguished a closed laboratory from other activities is the fact that the roles are carried out largely during a fixed period of time at a fixed location (concurrent/collocated). For many exercises presently carried out in a closed lab-
oratory, concurrency and collocation are essential—in some they are not.

The fundamental approach that we propose in this project is to start from first principles and examine the four different closed-laboratory styles using the diagram in Figure 2 as a model. Using the model we will determine those aspects of the various exercises that must be both concurrent and collocated. If either one (or both) is not essential then it suggests that the specific exercise might be a candidate for removal from the laboratory or at least for redesign. This analysis will indicate steps that can be taken to both enhance effectiveness and reduce costs.

A second important observation that we have made is the value of including specialized equipment. The use of this equipment actually drives the development of laboratory exercises in a very powerful way. Exercises quickly become different from those seen typically in undergraduate programs once a scenario based on specialized equipment is developed. In our second year course entitled "Software Development methods", for example, students have to treat the laboratory room as a warehouse and develop a security camera system for the warehouse. Although simple to state, it creates an atmosphere of considerable interest and brings many realistic problems to
the exercise.

Our cumulative experience with specific laboratory styles can best be summarized as follows:

- **Prescribed exercises.**
  These are an effective way to convey elementary and detailed material. We have used this style of laboratory extensively in our introductory course (CS101) and our second-year course in software development methods (CS201). In CS101, we have handled annual enrollments of more than 500 successfully. They are very resource-intensive in both preparation and operation.

- **Artifact analysis.**
  Our experience with these laboratories has been quite limited because acquisition of suitable artifacts has proved to be very difficult. The essential qualities of artifacts are that they be realistic (preferably real), that they do not use techniques beyond the training of the students, that they offer characteristics that permit the desired pedagogical result, and they can be both executed and analyzed with available equipment and resources. We have found very few artifacts that meet these requirements.

- **Studio presentation.**
  This laboratory style has been a resounding success. The success far exceeded our expectations. They are an excellent substitute for a pure master/apprentice model and they are inexpensive to operate requiring only a classroom and one or more instructors.

- **Case studies.**
  As with artifact-analysis laboratories, our experience with case studies has been very limited and for many of the same reasons. An additional problem with case-study laboratories is the preparation of the case. This is both difficult and resource intensive.

**Proposed Activities**
We describe project activities that will take place during the lifetime of the project by organizing them into three broad, high-level tasks: (1) refining methods and technologies; (2) processes for
developing new laboratory courses; and (3) new innovations for closed laboratories. Each of these three high-level tasks plays a role in meeting our goals, and each will include activities that test the results at other institutions, that evaluate the results at all institutions, and that package the results in a form suitable for dissemination.

- **Refining Methods and Technologies—High Level Task 1**

  The first step in this task is to carry out an analysis of our current laboratory-oriented student exercises based on our master/apprentice model. The results of this analysis will include details of which exercises require the concurrency and collocation properties of closed-laboratory implementation and which exercises can employ an open-laboratory implementation. We will then use these results to identify best-practices and tools to support all laboratory activities, identifying and constructing new tools or support mechanisms if needed.

  We will also revisit our analysis of the *teaching goals* defined for our core courses. This will be based on well-known models of teaching goals such as the Teaching Goal Inventory (TGI) developed by Angelo and Cross [18] (which is similar to that found on the Web site of the Field-tested Learning Assessment Guide [16]). We will also use Bloom’s taxonomy for levels of learning expertise. This detailed analysis will allow us to better see what goals laboratory-based activities should target and at what level of expertise. We will attempt to generalize this study of our core courses to apply to models for first-year courses, such as those described in Computing Curriculum 2001.

  Combining this goal-oriented analysis with our master/apprentice concept will drive our efforts toward defining “first principles” for using closed laboratories in computing courses. All of these analytic results will be accumulated and will constitute a significant part of a document that we propose to prepare, the *Laboratory Curriculum Component Development Guide* (LCCDG).

  The second step in this task is to examine the results of the analysis completed in the first step and to develop techniques for cost reduction. The first set of operational-cost-reduction techniques that we plan to assess and implement if they are worthwhile is:
- **Separation of closed and open laboratory elements.**

  Using the analysis that we will have completed in the first step of this task, we will separate the elements of current laboratories into those that need the closed laboratory context and those that do not. With this separation in place, we will reorganize and shorten the closed laboratory part of the exercise so that only those activities that require the concurrency and collocation of the closed laboratory are scheduled for closed-laboratory time.

- **Optimized resource utilization.**

  Anecdotal evidence suggests that students can and do learn from each other in closed laboratory exercises. We will investigate this issue and evaluate it by using a larger ratio of students to workstations than is usual (currently it is one) to see whether a better teaching paradigm can be developed if there is required, explicit cooperation between laboratory partners.

- **Electronic laboratory support.**

  We will investigate the use of electronic rather than direct communication between laboratory students and laboratory staff using a concept similar to Microsoft’s NetMeeting. This will be prototyped and tested for effectiveness and adequacy in our current facility basing the overall approach on the “language laboratory” model from foreign-language education. In such a facility, students’ progress is monitored by an instructor from a central facility. We will create a monitoring site and associated software in our present facility for evaluation but staff will remain collocated with the students. If local use is satisfactory, we will attempt to expand it to remote use. Remote use will facilitate concurrent but not collocated activities, and this will permit non-specialized facilities such as walk-in computer facilities and personal computers to be used.

  We will make use of the results of these three concepts by refining the existing laboratories in
two courses at Virginia, CS101 and CS201. This experience will allow us to better package, export and evaluate these methods and technologies for use at our test-site universities.

- **Processes for Developing New Laboratory Courses—High Level Task 2**

  Activities in this high-level task will produce two outcomes that can be exported to and evaluated at other institutions:

  - **The Laboratory Curriculum Component Development Guide (LCCDG)**
    
    The first outcome will be a development guide for laboratories. We will use the same set of best practices and tools described in the previous item as part of a process document (a “how to” guide) for creating a laboratory component for courses that do not have a closed laboratory. We will evaluate this by using it as the basis for adding a laboratory component to a course at the University of Virginia, and then export and evaluate its use at the test sites.

    The guide will include extensive process information, software, detailed examples, and templates from which others can develop necessary laboratory artifacts.

  - **The Studio Laboratory Development Guide (SLDG)**

    The second outcome we will produce will be a similar document to the first but will treat a particular style of laboratory, the studio laboratory, in considerable depth. This style has been singled out for special treatment because we have extensive experience and have had very considerable success with studio laboratories but they are not commonly used. The guide will also include extensive process information, detailed examples, templates and so on. We will evaluate this guide by employing it to develop new studio laboratories for specimen courses at Virginia and the test-site institutions.

- **New Innovations for Closed Laboratories—High Level Task 3**

  In this high-level project task, we will focus on two innovative approaches:

  - **Exploitation of inexpensive specialized equipment.**

    In the first approach, we will build on our extensive experience with specialized equip-
ment in laboratories by creating closed-laboratory exercises for our CS1 course (CS101) using the CMU Palm Pilot Robot. The Palm Pilot Robot designed by Carnegie Mellon University is a small autonomous mobile robot that uses a Palm Pilot handheld computer as its computing and display component. These robots are available commercially in kit form and are supported by a wealth of software, both development tools and applications. These devices are simple yet powerful, and they provide an attractive means to extend our use of specialized hardware to lower-level courses. We plan to introduce them systematically over the three-year period of performance to expand the educational opportunities that we have found valuable using the more complex devices illustrated in Figure 1. We will also be able to make them available on-loan to our evaluation partners during the latter part of the proposed project thereby simplifying the evaluation process.

- **Acquisition and preparation of artifacts.**

In the second approach, we will focus on developing closed-laboratory exercises that employ case studies and artifact analysis for courses at several levels of our program (our CS1 and CS2 courses, as well as the software-engineering oriented course CS340). Our limited success with artifact-analysis and case-study laboratories has come about because of our inability to locate, acquire, and prepare representative teaching materials that possess the necessary qualities of (1) realism, (2) being at the right technical level, (3) teaching suitability, and (4) suitable for available equipment. We plan to deal with these difficulties in three ways: (1) by soliciting assistance from colleagues in industry who have artifacts of interest; (2) by exploiting the now large collection of software artifacts that are in the public domain (such as open-source software); and (3) by including a technician in the planning for this project so that essential activities related to artifact preparation can be completed in a comprehensive and timely fashion.

As with our work using specialized equipment, after introducing these innovations
into our own laboratories, we will evaluate their effectiveness, and we will export them and evaluate their use at test sites after making any needed changes.

Activities that make up these high-level project tasks will take place in parallel. Figure 3, gives a high-level overview of our project plan and shows the three tracks that we are planning.

**Outline of Work**

In documenting our outline of work, we define activities for each of the three high-level tasks described earlier. We assign each activity a symbolic identifier (e.g. “RMI”) for easier reference. The set of activities for the high-level tasks are documented first. For each activity we give the
semester(s) in which that activity will take place. Finally, we provide a table showing the same information in chronological order by listing all activity ID by semester.

**RM—Refining Methods and Technologies (High-level Task 1)**

RM1: Analyze the need for laboratory components in a curriculum in terms of our master/apprentice model and the needs for collaborative learning. (*Semester 1*)

RM2: Review teaching goals (e.g. Angelo and Cross’ TGI) for CS1 and CS2 courses in general and for the University of Virginia’s CS101 and CS201 in particular. (*Semester 2*)

RM3: Review current closed laboratory contents for CS101 and CS201 in light of results from RM1 and RM2. (*Semester 2*)

RM4: Assess alternative educational delivery mechanisms that do not require closed-laboratory environment. Assess technologies that support effective delivery outside of a closed laboratory. (*Semesters 1 & 2*)

RM5: Determine development and operational cost for new laboratory techniques. (*Semester 3*)

RM6: Develop, implement, and assess explicit cooperation between laboratory partners. (*Semesters 3-6*)

RM7: Develop detailed design for electronic laboratory support (the “language laboratory” approach to laboratory operation). (*Semester 1*)

RM8: Develop, implement and assess electronic laboratory support. (*Semesters 2-9*)

Note: All results for these activities in the high-level task “Refining Models and Techniques” feed into the early activities for the second high-level task (described next). Thus related implementation and dissemination activities are described as activities in that section.

**PR—Processes for Developing New Laboratory Courses (High-level Task 2)**

PR1: Document results of activities RM1 - RM5 as a *Laboratory Curriculum Component Development Guide* (LCCDG) that provides guidance on how best to organize and deliver practicum-oriented learning activities inside or outside of a closed-laboratory. (*Semester 3*)

The LCCDG will be revised as experience is gained with new laboratory concepts
throughout this project.

PR2: Use the LCCDG to evaluate the laboratory component of CS101. (Semester 4)

PR3: Revise the CS101 closed laboratories and assess the revisions. (Semester 5)

PR4: Use the LCCDG to evaluate the laboratory component of CS201. (Semester 4)

PR5: Revise the CS201 closed laboratories and assess the revisions. (Semester 5)

PR6: Develop a new laboratory component for a course at Virginia. (Semester 4)

This course might be an existing course (e.g., CS333, Computer Architecture), or a new upper-division course.

PR7: Deliver and assess the newly developed laboratory component. (Semester 5)

PR8: Distribute the LCCDG to the test-sites and possibly other universities and solicit evaluations. (Semester 4)

PR9: Work with test-site universities to exploit the LCCDG in their curricula. (Semesters 5-7)

PR10: Prepare final form of LCCDG and publish on the Web and possibly via a traditional publisher. (Semesters 8 & 9)

PR11: Document the experience with the studio presentation model for a closed-laboratory course component that we have gained. Create a second guide document, the Studio Laboratory Development Guide (SLDG), for introducing studio presentation laboratories into computing courses. (Semester 3)

The SLDG will be revised as experience is gained with new laboratory concepts throughout this project.

PR12: Distribute the SLDG to the test-sites and possibly other universities and solicit evaluations. (Semester 3)

PR13: Work with test-site universities to exploit the SLDG in their curricula. (Semesters 4 & 5)

PR14: Prepare final form of SLDC and publish on the Web. (Semesters 8 & 9)

NI—New Innovations for Closed Laboratories (High-level Task 3)

NI1: Acquire and assess Palm Pilot robots. (Semester 1)

NI2: Using results of TGs and SLOs analysis of CS101, identify where introduction of robots
would be most effective, and develop closed-laboratory exercises. \textit{(Semester 2 & 3)}

NI3: Test and evaluate robot-based exercises and revise as necessary. \textit{(Semester 3)}

NI4: Introduce robot-based exercises into CS101 and evaluate effectiveness. \textit{(Semester 4)}

NI5: Review results of exercises with other universities and revise. \textit{(Semester 3 & 4)}

NI6: Incrementally expand role of robots in curriculum. \textit{(Semesters 4 - 9)}

NI7: Document use of robots and robot-based exercises for dissemination. \textit{(Semester 4)}

NI8: Loan several robots to test-sites for evaluation in their curricula. \textit{(Semester 5 & 6)}

NI9: Assist test sites to introduce robot-based exercises into their curricula. \textit{(Semester 7 & 8)}

NI10: Develop monograph about the use of robots in introductory computing courses and seek a publisher (minimally the Web). \textit{(Semester 8 & 9)}

NI11: Determine educational goals throughout the curriculum for artifact analysis and for case studies. \textit{(Semester 1 & 2)}

NI12: Determine approach to assessment of educational effectiveness of artifacts in both open and closed laboratory settings. \textit{(Semester 3)}

NI13: Acquire artifact candidates from industry partners & public-domain sources. \textit{(Semester 2)}

NI14: Develop artifact-analysis and case-study exercise(s) for third-year software development course, CS340. \textit{(Semester 3 & 4)}

NI15: Deliver CS340 artifact case-study exercise(s) and assess student response to its use. \textit{(Semester 5)}

NI16: Develop artifact case-study exercise(s) for the laboratory component of our second-year computer science course, CS201. \textit{(Semester 3 & 4)}

NI17: Deliver CS201 artifact case-study exercise(s) in Spring ‘03 and assess student response to its use. \textit{(Semester 5)}

NI18: Develop artifact case-study exercise(s) for the laboratory component of our first-year computer science course, CS101. \textit{(Semester 3)}

NI19: Deliver CS101 artifact case-study exercise(s) in Fall ‘02 and assess student response to its use. \textit{(Semester 4)}
NI20: Review results from all three deliveries for artifact case-study exercises. Revise in conjunction with input of representatives from test-site universities. Document so they can be easily adopted for use at test-site universities. (*Semester 6*)

NI21: Encourage personnel at test-sites to introduce artifact case-based studies into their courses as part of a closed-laboratory or as an open-laboratory assignment. University of Virginia personnel will assist in deployment and assessment of educational effectiveness. (*Semesters 6-8*)

NI22: Repeat delivery of artifact case-study exercises at Virginia. (*Semester 7 & 8*)

NI23: Seek more broad distribution of these educational resources including the Computer Science Teaching Center (http://www.cstc.org). (*Semester 9*)

Table 1 shows the proposed schedule for this project. The various activities are identified using the letter/number designation used in the paragraphs above and the time is shown by semester number with summers being counted as semesters.

<table>
<thead>
<tr>
<th>Semester:</th>
<th>Activity Identifiers:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Semester 1, Fall 2001</td>
<td>RM1, RM4, RM7, NI1, NI11.</td>
</tr>
<tr>
<td>Semester 2, Spring 2002</td>
<td>RM2, RM3, RM4, RM8, NI2, NI11, NI13</td>
</tr>
<tr>
<td>Semester 3, Summer 2002</td>
<td>RM5, RM6, RM8, PR1, PR11, PR12. NI2, NI3, NI5, NI12, NI14, NI16, NI18.</td>
</tr>
<tr>
<td>Semester 4, Fall 2002</td>
<td>RM6, RM8, PR2, PR4, PR6, PR8, PR13, NI4, NI5, NI6, NI17, NI14, NI16, NI19.</td>
</tr>
<tr>
<td>Semester 5, Spring 2003</td>
<td>RM6, RM8, PR3, PR5, PR7, PR9, PR13, NI6, NI8, NI15, NI16, NI17.</td>
</tr>
<tr>
<td>Semester 6, Summer 2003</td>
<td>RM6, RM8, PR9, PR14, NI6, NI20, NI21.</td>
</tr>
<tr>
<td>Semester 7, Fall 2003</td>
<td>RM8, PR9, NI6, NI9, NI21, NI22, NI23.</td>
</tr>
<tr>
<td>Semester 8, Spring 2004</td>
<td>RM8, PR10, PR14, NI6, NI9, NI10, NI21, NI22.</td>
</tr>
<tr>
<td>Semester 9, Summer 2004</td>
<td>RM8, PR10, PR14, NI6, NI10, NI23.</td>
</tr>
</tbody>
</table>

Table 1: Project schedule.
Experience and Capabilities of Principal Investigators

The principal investigators have extensive experience in curriculum development and closed laboratories. Knight is the chair of the curriculum committee in the Department of Computer Science at the University of Virginia and has held that position for several years. He was the developer of two core laboratory courses at Virginia—Software Development Methods (CS201) and Advanced Software Development Methods (CS340). He pioneered the use of studio laboratories in CS340 and refined the techniques considerably. He also led the introduction of specialize hardware into the core courses at Virginia including both the remotely-controlled TV camera and the mobile robots.

Knight has been involved in the curriculum development activities at Virginia since they were started almost a decade ago. He participated in all the planning activities that resulted in the first and all subsequent versions of the current curriculum and made numerous contributions to the curriculum philosophy and content.

Until recently, Horton was on the faculty of Florida Atlantic University. He spent a sabbatical leave at Virginia in 1999 and then joined the Computer Science Department at Virginia in 2001. During his sabbatical leave he co-taught CS340 with Knight and enhanced the techniques in the course considerably. After returning to Florida Atlantic University, he took the CS340 course model and laboratory manual and developed a version more suitable for the curriculum at Florida Atlantic University. He taught that version of the course in Summer 1999 and Spring 2000.

Horton has extensive experience in both curriculum development and curriculum assessment. He designed and implemented a complete revision to FAU’s introductory programming course (making it laboratory-based) and also designed and implemented the closed-laboratory component for the follow-up course for computing majors. This laboratory was assessed for several years after its introduction. Horton is nationally active in software engineering education issues, particularly with the Working Group on SW Engineering Education and Training (http://www.sei.cmu.edu/topics/collaborating/ed/workgroup-ed.html). He is a CSAC/CSAB program reviewer and has been very involved in ABET and CSAB accreditation for his departments.
Evaluation Plan
The evaluation plan for this proposal will address both formative and summative evaluation aspects. We will measure both the effectiveness of the educational resources we develop at the University of Virginia and at the test-site institutions. The Computer Science Departments at James Madison University in Virginia and Florida Atlantic University have agreed to support both the evaluation of and the dissemination of the results of this project (see attached letters of support). We will also evaluate our goals pertaining to making it easier and less costly to create laboratory course components and activities as well as operating them with fewer resources.

One of the project PIs, Horton, will have lead responsibility for project evaluation. We will form a project evaluation committee (PEC) that will hold dedicated evaluation-focused meetings at regular intervals (and will review document and reports more frequently using e-mail). The PEC will have some members who are not direct participants in the project work in order to gain an “outside” viewpoint and direction. Two members of the University of Virginia’s Computer Science department who have experience with the evaluation of projects like those funded by the NSF, Jane Prey and James Cohoon, have agreed to serve in this role. (Cohoon is a co-author of the textbook for our course CS101, which features prominently in our project plan.) In addition, we will ask representatives of our test-site institutions to participate. We will recruit at least one other person from our university who is familiar with evaluation and laboratory innovation—there are several candidates in our engineering school and in our physical science departments. Finally, we will request participation from our Teaching Resource Center (http://www.virginia.edu/~trc), an organization on our campus that fosters and actively supports education innovation among Virginia faculty.

Planning Evaluation
Early in the project, the PEC will meet to help us plan evaluation. The major goals of this meeting will be to review this evaluation plan, strengthen it, create detailed plans for subsequent evaluation activities, etc. We will re-examine the overall goals of the project in order to ensure that our proposed evaluation activities will really measure those goals in an effective and practical man-
ner. We will also agree upon a schedule of evaluation meetings and review for the three years of project activity. The plan presented here is the PIs’ initial vision of such a schedule, but we expect the PEC to improve upon it.

Results of this planning evaluation will be presented to the industrial advisory board of the Computer Science Department at the University of Virginia at the earliest opportunity in order to get input on our activities from this group of stakeholders. This board meets approximately once a year, and includes representatives from many companies that regularly hire our graduates.

**Formative Evaluation: Implementation**
The PEC will hold an implementation evaluation meeting at the beginning of each summer. We have chosen this time because we can review work carried out during the academic year, and can address issues before the PIs begin their work during the summer. The goal of this meeting will be to review goals and activities for the previous time periods and determine whether the scheduled activities were carried out or if changes in the project plan must be made. We will also re-examine the details of the project plan for the upcoming year and address any concerns. Results of this implementation evaluation will be documented and distributed to senior project personnel as well as representatives at the test-site universities.

**Formative Evaluation: Progress**
The senior personnel will be responsible for carrying out formative assessment activities during this project that are specifically designed to assess progress towards project goals. The nature of this proposal and the detailed project plan make it very feasible to gain such insights well before the project is completed. As noted in the detailed project plan presented earlier, all of the project deliverables are developed, then applied in courses at Virginia, then made available for use at the test-site universities. (Major deliverables are: the LCCDG and the SLDG; the laboratory activities based on the Palm Pilot robots; the laboratory activities based on case studies and artifacts.) Personnel at the University of Virginia are experienced in evaluating the application of an educational resource, and our plan explicitly includes planning for assessing the effect on student learning for each new laboratory innovation or modification we introduce. The PI in charge of
evaluation and the PEC will remind PIs of their responsibilities in this regard, and assist the group in meeting this responsibility effectively.

Two examples will illustrate our concern for formative evaluation and how our plan allows for its effective use. Suppose that we develop a laboratory exercise for Virginia’s CS201 course that uses an artifact from industry to create a case study that teaches students about class design in large software systems. As noted in the detailed project plan, this exercise will be planned and developed during Semesters 3 and 4 (activity NI16) and then used during Semester 5 (activity NI17). While developing the exercise, we will explicitly consider how to assess it. We might employ a classroom learning technique (CAT) to do this, or use a survey or some other mechanism. We will also define an assessment method that can be employed the semester before it is implemented (i.e. Semester 4 in this case) to see how well our students understand class design in large systems without the new case-study exercise. These results will be compared to our assessment results the next term when we do introduce the new activity.

For our second example, activity PR2 in our detailed plan is about planning a major revision for the closed-laboratory component of CS101, and activity PR3 is the introduction of this new version of the closed-laboratory component of the course. The goals we would seek to measure relate to the effectiveness of the new version in terms of providing a successful learning experience, and also in terms of moving certain activities outside the closed laboratory, focusing the closed-laboratory meetings on activities that require that environment, and finally about the cost of operating the new version of the laboratory component (in terms of TA time, space resources, etc.) Again, during earlier semesters we will gather information about operation costs under the old system. We will go beyond our usual methods of assessing student satisfaction and the education effectiveness of the laboratory components before the semester we introduce the new version, in order to obtain baseline results that can be compared to measurements made after we introduce the new version. In addition to applying these same assessment measures for the new version, we will use a method that we have found successful in evaluating previous NSF-funded curriculum innovations: our weekly TA meetings have proved to be a rich source of immediate, formative
feedback on how well new laboratory activities are working. The TAs funded under this project will have specific responsibilities for attending laboratory meetings and monitoring these aspects.

One of the goals of this project is to develop guides that can be employed by instructors to create new laboratory components (as opposed to individual laboratory exercises or activities). At the end of the project our summative evaluation will measure if adopters at test-sites found that these met our goals. Before then, project personnel at the University of Virginia will have used these guides to revise existing closed-laboratory courses (for CS101 and CS201) and create new laboratory components for other courses. Personnel carrying out these activities will record how they used the guide, problems they encountered, etc. while using the guides, and create an evaluation report. These materials will be used to revise the guides, but will also be available to the adopters at the test-sites when they use the guides in the third year. In this way we believe we will have interim formative evaluation information on how well we are succeeding assisting instructors in their laboratory development needs.

Faculty at the University of Virginia make use of a Web-based survey system developed here: SurveySuite (http://intercom.virginia.edu/SurveySuite). This is a mature and robust tool that is freely-available to anyone and easy to use. In using this tool and developing other assessment mechanism, we will look ahead to the need for adopters at test-sites to carry out the same assessments, and thus will develop and document our methods so they can be easily applied later in the project at the test sites.

The annual meeting of the PEC will review results of these formative evaluations that attempt to obtain measures of our success in achieving project goals.

**Summative Evaluation**

At the conclusion of our project our summative evaluation activities will address how well we met our original goals. We will carry out these activities in conjunction with representatives from the test-sites that adopt our deliverables, and this activity will receive careful attention from the PEC before it is carried out. This evaluation will address all stakeholders. We will review logs kept by project personnel at the University of Virginia and at the test sites as they used the devel-
opment guides or introduced new innovations. We will use questionnaires and surveys (using SurveySuite) to capture student reactions and outcomes. We will review records of development cost and operation cost information to determine if we have met our cost-reduction goals for the departments participating in this project.

We anticipate using the design matrix technique outlined in the NSF’s *User-friendly Handbook for Project Evaluation*. The PIs will develop a complete plan in the early part of the project, and the PEC will review this at its first meeting. But the following describes our initial ideas about appropriate evaluation questions, information gathering techniques, and collection process.

Our first set of questions relate to the general evaluation question:

- **Did project deliverables help create closed-laboratory components that are both effective and less resource intensive?**

More specific subquestions for this goal might include the following:

- Did revised version of closed laboratory components reduce operation costs? (Data on this would be collected from records kept by instructors teaching or supervising courses with a closed laboratory component. Date would be collected immediately after each course was taught.)

- Did the revised versions of the closed laboratories provide effective learning experiences for students? (Data would be based on various assessment methods, as discussed above in the section on formative evaluation, during each semester the revised laboratory was offered.)

These are examples. Other subquestions in this category have been omitted for space reasons.

Our second set of questions relate to this general evaluation goal:

- **Did the project produce educational resources that help instructors develop laboratory course components?**

More specific subquestions for this goal might include the following:

- Was the Laboratory Curriculum Component Development Guide (LCCDG) used at
the test-sites? (Data would be collected using questionnaires for the instructors at the test sites.)

- Was the LCCDG useful to faculty in revising laboratories? (Questionnaires would be given to Virginia instructors who use the guide to revise CS101 and CS201, and to instructors at test sites.)

- Was the LCDDG useful to faculty in developing new laboratories? (Questionnaires would be given to the Virginia instructor developing a new laboratory for an upper-division course, and to instructors who develop new laboratory components at the test sites.)

Other subquestions in this category about the LCDDG and the SLDG have been omitted for space reasons.

Our third set of questions relate to this general evaluation goal:

- *Did the project produce new innovative activities for laboratory courses?*

More specific subquestions for this goal might include the following:

- Were the robot activities developed for early computing courses effective learning activities? (Assessments as described above in the formative evaluation section will be applied each semester these exercises are used.)

- Were the robot exercises easily adopted at test-site universities? (Questionnaires will be given to instructors who introduce these exercises at the test sites.)

Similar subquestions in this category regarding the other deliverables have been omitted for space reasons.

In addition to these evaluations, we plan on surveying graduates of the University of Virginia and the test sites who have experienced the laboratory activities improved by project activities. If any of these students are working in industry at the end of the project, we will attempt to contact them to gather responses about their educational experiences and their impact on getting hired and performing their job duties.
Dissemination of Results
Since our project plan is centered on providing test-site universities with the ability to adopt and evaluate our work products during the latter half of the project, dissemination is a critical focus of our work. As noted earlier in the detailed project plan, we will get input from test-site personnel when planning work products (like the LDCCG or a robot-based exercise) that will help us succeed in creating readily adaptable resources. Also, as discussed in the evaluation plan, we will work with the test-site personnel to evaluate how easily and effectively resources can be adopted. These experiences will help as we increase our efforts to disseminate our results more broadly in latest year of the project.

Two of our work products are guides for creating laboratories and resources. We intend to hold a faculty development workshop during the second summer of the project (Semester 6), if supplemental funding can be obtained from the NSF or from industrial sponsors. We feel a workshop is the best mechanism for bringing together interested faculty from many universities to explain the principles and processes these guides will address.

As noted earlier, we feel the these two guides are best suited for publication on the Web rather than commercial publication. We hope to present papers on how we have used them in revising and developing new laboratory course components at SIGCSE and FIE conferences before the workshop, thus generating interest in both the workshop and the documents themselves.

We believe that commercial publication is a viable option for two of our work products. First, the laboratory activities for early programming courses based on small robots could be turned into a short monograph that could be used to supplement a textbook in CS1 or CS0 course. Such a book might be similar in nature to the books on Karel the Robot. We will explore this idea with a publisher. Second, as we revise the closed laboratory component of our CS101 course, we will create a new edition of the commercially published “laboratory manual” for the textbook we use, *C++ Program Design: An Introduction to Programming and Object-oriented Design*, by our colleagues at Virginia, James Cohoon and Jack Davidson. Cohoon has given his approval of this idea. The text has been a very successful publication for McGraw-Hill, and we feel that this pro-
vides an effective means of exporting new ideas for closed laboratory use.

Individual exercises, whether they are robot-oriented or based on case-studies and artifacts, will be developed so they can easily be submitted to the Computer Science Teaching Center (http://www.cstc.org), a repository for laboratory exercises and activities.

We will seek publication of results at conference like the ACM SIGCSE Technical Symposium on Computer Science Education, the Innovation and Technology in Computer Science Education conference, and the ASEE/IEEE Frontiers in Education conference, as well as in journals such as the ACM Journal on Educational Resources in Computing (JERIC), etc. Applications of case studies and artifact exercises would be appropriate for a paper (or even a tutorial) at IEEE-sponsored Conference on Software Engineering Education and Training.

Evaluating the effectiveness of dissemination can be challenging. Our experiences with the test-site universities will give us feedback on what we are doing well and how we can improve, and we will use this to guide dissemination efforts nearer the end of the project. Evaluation of this activity will be a topic discussed and monitored by the Project Evaluation Committee.
REFERENCES


ries in the computer science curriculum.” Proceedings of the 86th annual meeting of the Illi-
nois State Academy of Science. 47.


19. Beheshti, Mohsen and Bill Waller, “Instructional Software for Closed Laboratories in CS1


100-105.

science curriculum.” Proceedings of the 26th Small College Computing Symposium, 78-86.

23. Mutchler, David, Claude Anderson, Andrew Kinley, Cary Laxer, Dale Oexmann, Rimli Sen-
gupta, Frank H. Young, “CS1 Closed Laboratories = Multimedia Materials + Human Interac-
tion,” Rose-Hulman Institute of Technology, USA ITiCSE’99 Cracow, June 27 - July 1, 1999.

24. Robbins, Kay A., Catherine Sauls, Key, Keith Dickinson, and John Montgomery, “Solving the
CS1/CS2 lab dilemma: students as presenters in CS1/CS2 laboratories,” Proceedings of the
32nd SIGCSE Technical Symposium on Computer Science Education. Pages 164 - 168.


lege Of New Jersey Proceedings of the 32nd SIGCSE Technical Symposium on Computer