describes the support and pedagogic mechanisms used, and presents preliminary observations from the first year.

1.5 Further Aims with the Runestone Project

The Runestone project aims to:

- Give students international contacts and experience with teamwork with people from a foreign culture.
- Give students experience of collaboration with a group having a different educational back-ground.
- Encourage learning through peer-teaching.
- Give students experience with the use of Information Technology in problem solving.
- Prepare students for the possibility of working in a foreign culture.
- Use the foreign experience to aid students in producing a superior product locally.
- Benefit staff by close collaboration with other universities, giving insights to other departments and ideas for new teaching methods.
- Gain experience with use of new techniques in the running of a course.

Another goal is to create a well-organized setting with courses that, after the initially higher start-up costs, run at normal or lower costs. One example of cutting costs without compromising quality is the use of student peer-teaching, which can reduce the demand for staff hours. Another example of cost cutting is that the costs for renewing the course can be distributed across the departments involved.

In carrying out the Runestone project, we will establish results that address the issue of transferability to other departments and institutions. For this reason, the evaluation will aim to distinguish between domain-specific and general lessons, particularly with respect to the impact of international collaboration on group interaction and personal development, the extent of peer-teaching, and the costs of using this form of education. For example, the project shall examine questions such as how much time is spent on becoming acquainted with new techniques for communication and in what ways (if any) using non-native language impairs learning.

1.6 Peer-teaching

Based on anecdotal evidence from our own experience as teachers, we believe that having students explain concepts and solutions to one another is a powerful learning technique. Our conjecture is that there will be plenty of occasions for the students involved with the Runestone project to help each other with activities such as explanations, clarification, sharing knowledge or rehearsal of ideas. Occasions for peer-teaching can be formal or informal. Formal occasions arise when students at site X present

information for the students at site Y. Informal occasions include questions that arise during day-to-day e-mail or simple study sessions.

The Runestone project will systematically examine peer-teaching by considering which settings tend to encourage or discourage peer-teaching as well as factors that contribute to the effectiveness of peer-teaching in these situations. One of our hypotheses is that the rather different educational backgrounds of the two sets of students involved in the project will encourage peer-teaching. The differences in backgrounds should motivate the students to articulate their reasoning, rather than assuming that there is mutual tacit understanding between them and their foreign counterparts.

1.7 The pilot study

From early January through late March 1998, the Runestone project ran a pilot study, which involved a group of eight students: four in Uppsala and four in Michigan. These students were all in their third or fourth year of university studies. For the Swedish students, the group project was part of a course that started in September, whereas for the Americans it was the major part of a course that started in early January. The problem that was specified for the group project was fairly advanced, involving study areas such as real-time systems, networking, and distributed systems. A major goal of the Runestone project is to examine the influence of the group project and factors in how the project is set up upon what the students learn and how they learn it.

2. THE GROUP PROJECT

2.1 The Task

The actual project was to navigate a steel ball through a maze by tilting the maze in two dimensions with stepper motors. The user submits a navigation algorithm, defines a path for the ball to follow, requests the server to execute the algorithm, then waits for access to the game. When the user gains access, the game server resets the ball in the maze, executes the user's navigation algorithm, then provides feedback to the user on the result of the run. Feedback includes information on how the navigation code executed, and a graphical display of the path which the ball traced through the maze. The input to the navigation algorithm is the position of the ball. The output is the rotational positions of the motors as a function of time. Video images of the maze and ball are available from a black and white digital video camera.

This is how the group project [the Brio project] was presented to the students:

Goal

Navigate a steel ball through a maze by tilting the maze in two dimensions with motors. Use a web interface to submit a navigation algorithm, select a path for the ball to follow, request the server to execute your algorithm, wait for your turn, and watch the results of a run via video images. The input to the navigation algorithm is the position of the ball. The output is the rotational positions of the motors as a function of time.

Design

It is up to each team to decide on the architecture and design of their project. What you will be given is a requirement specification describing what your project must do, a set of hardware components (described below), and some useful software components (described below).

As the time frame for the project is quite short (approximately 8 weeks) we have scheduled a regular weekly meeting time for you to report progress, ask questions, advise of problems, etc.

We will specify a set of deliverable documents and their deadlines for the duration of the project. The deadlines on the documents will be to help you stay on track and make regular progress towards the completion of the project.

Hardware components

The following components are available for your projects. The unique components will live in rum 1219 and students will have keys to this room.

- Connectix camera [inputs, outputs & ports]
- Brio game [stepper motors, controlling electronics & inputs]
- Desktop computer [attached mounted camera & Linux and Win95 installed]
- Laptop computer [serial port & Ethernet]
- Client computer [net connection, web browser & Java compiler]

Software components

These components will be available locally and will be properly configured, tested, and if appropriate, installed.

- Linux device driver for Connectix camera
- Apache HTTP server for Linux
- Linux JDK 1.1
- image processing example code

Software technology

The following technologies or concepts will be important for the project. It would be a good idea to review these ideas, secure references (books and the web) for them, and ponder their uses over the Christmas break.

- dynamic loading of compiled code
- client/server programming technologies
- graphical user interface web client applications
- dynamic web programming technologies

- serial port programming
- interfaces and their code representation
- display of image data
- manipulation and transmission of image data
- image processing algorithms
- use of ioctl to control parallel and serial ports

2.2 The Hardware Set-up

The hardware components available for the projects were located in Uppsala. The central piece consisted of a desktop computer, with a black and white digital video camera attached to its parallel port and an Ethernet connection to a laptop computer (the Swedish students each had access to a personal laptop). One of the laptops was used to communicate with the two rotational stepping motors via its serial port. The camera was permanently mounted over the Brio maze game as was a light source. A second laptop, connected via Ethernet, was used to run as a client computer with a web browser for playing the game.

The software components included a C library of code to read video signals from the parallel port, control camera settings, Motif app (Ximprov) for viewing camera data, experimenting with camera settings, an example C program using camera data (Ximprov), an Apache HTTP server for Linux, and Linux JDK 1.1 with RMI support

2.3 The Website

The starting point for playing the game (running the maze) was a website. This website had to:
- Display the currently installed maze board.
- Allow user to define a path for the ball to follow.
- Accept user's navigation algorithm to execute.
- Give feedback to user during the run of the maze.

Optional extras were to:
- Provide information to a user on developing navigation algorithms.
- Notify a user about game queue, estimated waiting time, etc..
- Use RMI technology appropriately.
- Display graphical representation of a run, superimposed on selected path.
- Display a full video image of the maze.

2.4 The Game Server

The game server needed to be a concurrent system, either multiple processes or multiple threads, and had to:

- Maintain a queue of users who wish to play, insuring mutually exclusive game semantics.
- Accept navigation algorithms and selected paths from clients.
- Provide feedback to client on the success/failure of navigation code.
- Provide data to the client on the ball's movement during a run of the maze.
- Provide a framework in which navigation code executes predictably and safely.
- Be able to reliably reset the ball to the documented starting position.
- Drive the stepper motors via a serial port interface.
- Use priority to schedule the concurrent entities properly.
- Fetch ball position information from the video server and make available to navigation code.

Optional extras were to:

- Provide information to clients about game queue, estimated waiting time, etc..
- Provide a documented framework in which navigation code executes.

2.5 The Video Server

The video server had to consist of one or more processes which must:

- Read video frames from the camera as fast as possible.
- Reduce video data to an x,y location of the ball on the maze.
- Make ball position information available via a network connection.

Optional extras were to:

- Provide a grayscale video image of the maze via a network connection.
- Provide as many positions updates per second as possible.