COMPUTER-MEDIATED COMMUNICATION IN A SOFTWARE ENGINEERING PROJECT COURSE

by

STEVEN PAGE

B.Sc.(Computer Science), The University of New Brunswick, 1994

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

in

THE FACULTY OF GRADUATE STUDIES

(Department of Computer Science)

We accept this thesis as conforming to the required standard

THE UNIVERSITY OF BRITISH COLUMBIA

July 1997

Abstract

Independent tools and fully integrated systems are currently being applied to educational settings, delivering educational content and activities to students. Asynchronous, computer-mediated communications enable students to reflect upon what they are learning, share their thoughts, and read those of others. Such communication tools facilitate active and collaborative learning.

Two integrated systems, the Virtual University and the World Wide Web Course Tools, are examined, and compared to one another. Both are Web-based systems that deliver course content and activities to students, across a variety of hardware platforms and without geographical restriction. They are systems that integrate features for a full learning environment, combining instructor preparation facilities, on-line course content, student activities, assessment facilities, and communication tools into a single application. Comparisons are drawn between these systems based on their feature sets, and in the context of CPSC 319, an undergraduate software engineering team project course.

I have observed the course, as a teaching assistant, for two years. It appeared to be a good fit for on-line tools, particularly because teamwork is a central emphasis of the course, and the students are familiar with computers. The intensity of the group work, however, and the unique grading structure, makes the course less compatible with these tools than it first appeared. A study of the course newsgroup and other on-line conference systems showed that, as expected, most students participated in the optional on-line discussion groups. The shape of this participation changed, however, between 1995-96 and 1996-97. There was a significant increase in the number of messages posted by each participant, while fewer students actually posted messages. The students rejected the VGroups conferencing system overwhelmingly, because it was perceived to duplicate tools they already had, and failed to reach a critical mass of participants.

Table of Contents

List of Tables

List of Figures

Acknowledgments

This research has been funded in part by the Natural Sciences and Engineering Research Council, in the form of a post-graduate scholarship. Other support has come through the TeleLearning Network of Centres of Excellence program, Project 3.2.

I would like to thank my supervisor, Dr. Kellogg Booth, for his guidance, my readers, Dr. George Tsiknis, and Carolyn Wick, for their feedback, and my wife, Julie Golding Page for her support.

Chapter 1 -- Background

For centuries, the traditional classroom, consisting of a unidirectional flow of information from the teacher to the students, has been the main model for educational environments. With the advent of computer-based communication, teachers in universities and other institutions perceive the opportunity to improve the learning process. This could come through faster, more efficient delivery of educational materials, greater student participation in discussions and interaction with the material, or the increased convenience to all participants achieved by decoupling learning from a specific time and place. New approaches to education are emerging that facilitate the delivery of courses, either partly or wholly on-line, in a networked computer environment. On-line tools emphasize computer-supported collaborative learning (CSCL) through discussions and group project activities. Some of these tools enable course participants to make full use of the textual, image, and multimedia capabilities of the World Wide Web and the Internet.

The purpose of this thesis is to examine several of these software tools, created for on-line learning, and to focus in particular on two systems that are being developed in British Columbia. Both of these tools are gaining success across Canada and around the world. The Virtual University (Virtual-U), developed at Simon Fraser University, and the World Wide Web Course Tools (WebCT) application, developed at the University of British Columbia, are discussed as exemplary systems. These and other electronic tools for on-line learning and distance education deliver learning activities and materials, facilitating discussions and access to course materials.

The comparison between systems is made in the context of CPSC 319, an eight-month software engineering group project course offered by the Department of Computer

Science at the University of British Columbia. Many traditional courses work as well or better in the on-line environment [Hiltz 1995, Harasim et al. 1995]. Some types of courses, however, do not easily fit the emerging new delivery methods. Courses with fluid marking schemes can be poor fits, if on-line grading facilities do not handle unusual cases gracefully. Courses which focus primarily or completely on group projects and collaborative work may share a common pedagogy with these on-line tools, yet they offer no guarantee of fitting with current CSCL systems. Courses that employ on-line discussion areas for practical issues, such as group coordination, may strain a system geared for learning through on-line intellectual exchanges.

The course being considered here, CPSC 319, has individual and team grades, requires formal group work, and permits teams to negotiate their own marking structure. This makes it a good candidate to test how well the on-line systems can support a course of this type. My involvement with the course has been as a teaching assistant and observer for the past two academic years, 1995-96 and 1996-97. This has involved working with teams and individuals, grading assignments, lecturing on occasion, and serving as the database contact.

This document assesses existing on-line group learning tools, and offers suggestions and lessons learned in this exercise. It is divided into five chapters. Chapter 2 describes the course, CPSC 319. Its structure, history, and idiosyncrasies are discussed, establishing a basis upon which to evaluate, for a software engineering project course, the strengths and the weaknesses of CSCL tools. Chapter 3 describes existing systems for delivering part or all of the educational content of a course in an on-line environment. Special attention is paid to the Virtual-U and to WebCT; these two systems are examined in greater detail, and are compared directly to one another on the basis of their respective feature sets. Chapter 4 reviews related studies, and examines several of the issues from the literature in the context of CPSC 319, including a variety of textual annotation tools as methods for more easily tracking and evaluating the process and the learning exhibited in the interactions of the group members. Chapter 5 summarizes what we have learned from our studies, from the particular perspectives of the course instructor, the enrolled students, and researchers in the fields of computer science and education. The remainder of this introductory section establishes a common background for the rest of the thesis by defining several key terms and concepts that arise in the literature relevant to this study.

1.1 Telelearning

A central theme in this document is the delivery of course content and coordinated group learning activities supported by a system of networked computers. This is sometimes referred to as "telelearning," which is defined as "the use of multimedia learning environments based on powerful desktop computers linked by the information highway" [Curry 1996]. This definition explicitly incorporates variation in the employed medium, and places the activity in a state of dependence upon computers connected through the Internet or some other computer communication facility. One way in which this can be accomplished is through learning networks, which employ modern communication technology to broaden the pool of peers in the courses, without requiring the participants to gather in a common central location [Harasim et al. 1995]. The issues of time, place, and speed of covering the material are then placed primarily on the shoulders of the individual students, while the ease of communication encourages and explicitly emphasizes collaboration and participation. An ambiguity surfaces in the term "learning network," a problem upon which Harasim et al.

comment. The term “network” is used throughout their book in reference to the computer-based aspect, in reference to the linking together of the computers, at different physical locations, but it can also refer to the group of students that are collaborating and learning. Harasim et al. carefully trace the history of the electronic versions of learning networks, from the ARPANET to the emergence of electronic mail and the USENET, Bulletin Board Systems, and the increasing acceptance of these systems for education at all levels [Harasim et al. 1995]. They view a “networked classroom” as one in which computer technology is employed for synchronous and asynchronous discussions, and connections to the information available over the Internet and the World Wide Web can be made.

1.2 Groups and Collaboration

Collaboration is a main emphasis and advantage of many telelearning environments. Written text is a key component of education: textbooks, course notes, examinations, and assignments are all text-based resources used in the traditional classroom. Through text, teachers develop the knowledge of students in the class. Language and text help to frame ideas and concepts in the minds of the students; they share their ideas and knowledge with others, largely through spoken and written words. The computer enhances simple text-based learning by making it interactive; students become participants, rather than observers. In class discussions, students help create the text from which they are learning. When this interaction is asynchronous, students are encouraged to read the messages, consider what is being said, and carefully craft a thoughtful reply [Harasim et al. 1995]. Discussions in the traditional classroom, on the other hand, are restricted to a single speaker at a time, with little time for reflection and crafting of exact phrasing. On-line communication has been found to increase both the quantity and quality of class interaction [Eisenberg and Ely 1993]. This improvement comes from decreased inhibitions; while students may feel shy in face-to-face class discussions, they are freed by the lack of social cues in the on-line environment.

Learning networks that encourage this increase in textual interaction among the students are frequently called "computer-based conferencing systems." This introduces "conference," a potentially ambiguous term. In a telelearning system, "conference" may be employed in reference to a message-based, class discussion or lecture [Wick et al. 1996, Harasim et al. 1995]. The term is not found in every such system, but two notable telelearning environments examined in this document - the Virtual-U, and the Virtual Classroom - consider a conference to be an asynchronous messaging structure, for discussion groups, question-and-answer sessions, debates, project planning, and more. The ambiguity of the term is caused in part by the previous existence of the concepts of "video conferencing," and "conference call," with the accompanying conceptual expectation of real-time group communication capabilities. The ambiguity, specifically with respect to its use in the Virtual-U, leads to people being "completely shocked to discover its asynchronous nature" [Wick et al. 1996]. Bulletin board and newsgroup message systems have many similarities with the Virtual-U conference tool, but avoid the ambiguity by using metaphors that imply asynchronous communication modes. The newsgroup calls to mind images of letters to the editor, which reply to one another after a separation of several days. The bulletin board metaphor implies making a posting and leaving it behind for others to see in the coming days.

The conferencing system within the Virtual University, discussed in Chapter 3, is called Virtual Groups, or simply VGroups. This introduces another potentially ambiguous term: "group." In the Virtual-U, a group denotes a named subset of participants within

VGroups, those who follow a given conference. At the basic level, a group is a collection of people who participate or have access to one of the on-line conferences. In the human sense of “group work” on the other hand, a group is a social unit of two or more people who exist in a sufficiently close relationship that they exert some influence over one another through their internal communication over a given period of time [Finholt and Sproull 1993].

In the university-based team software engineering environment that is under consideration in this document, a group is a set of people in the same place (the university), working together on a project, collaborating in software development. This collection may more properly be called a program team [Weinberg 1971]. Weinberg makes this distinction, seeing a “programming group” as people in the same place, working on the same machines, but not necessarily on the same projects; by his definition, a team is a subset of the larger group (although the two terms will be used interchangeably in subsequent sections). Weinberg further differentiates groups and teams from a “programming project,” which he considers a superset of people, or multiple teams of people, building either a collection of programs or a single, large, integrated system. Greater consideration will be given to groups and teams in Chapter 2, where CPSC 319 is discussed.

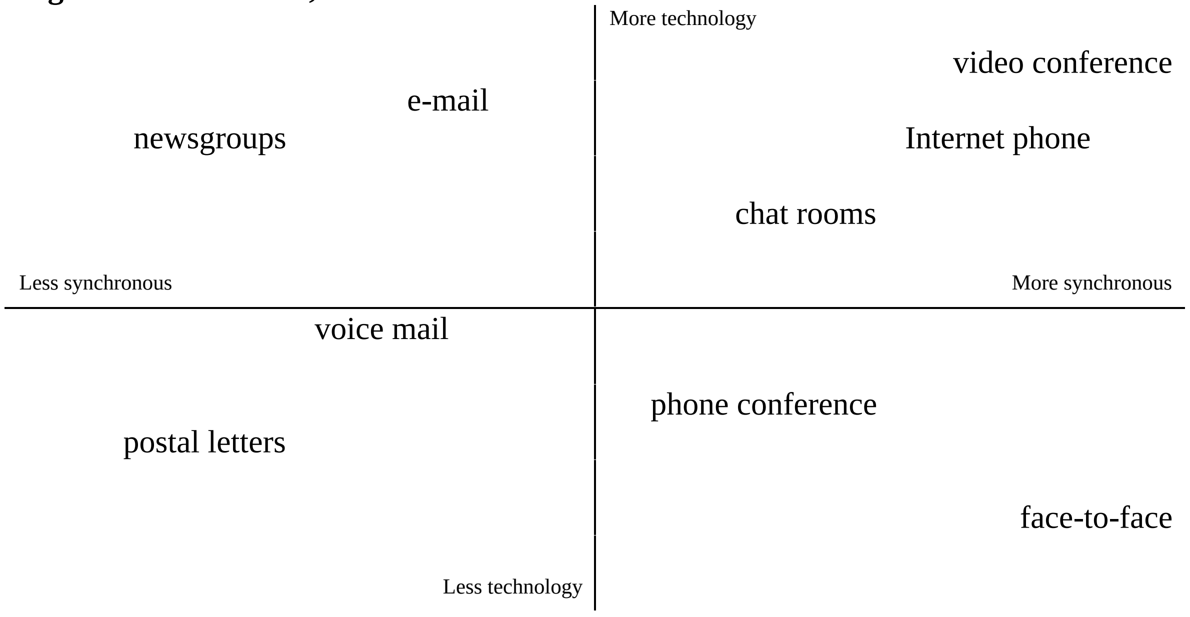
1.3 Computer-Supported Collaborative Work

In educational settings, in the business world, and in informal personal communication there are distinct intersections between the modality of the communication and the media through which the communication is accomplished. The modality encompasses the style of communication that is being engaged in; synchronous and asynchronous communications are two distinct modalities that impact what interaction is possible. These are traditional, broad categories into which the Computer-Supported Collaborative Work (CSCW) community places the various applications that support on-line collaborative work. “Synchronous” designates real-time exchanges between participants, which are described as “tightly coupled” and “fine-grained” by Edwards and Mynatt [1997]. A face-to-face conversation uses a synchronous mode of communication, as does an electronic chat room. “Asynchronous” is a more flexible term, and can encompass expected delays ranging from a few minutes to an indefinitely long period of time. The term designates, simply, communication that can occur at different times for the different participants [Edwards and Mynatt 1997]. Asynchronous electronic mail does not require that the recipients be currently logged on to their systems for communication to occur. Similarly, correspondence by traditional mail also occurs asynchronously, and carries expectations of an arbitrarily long wait for a reply.

The medium of the communication encompasses the set of devices through which interaction is accomplished. A face-to-face conversation uses nothing but a common spatial location, while electronic mail, in freeing the participants from temporal and spatial requirements, needs appropriate software and networked computers.

Figure 1 illustrates a taxonomy of communication media according to the interaction modalities. The vertical axis shows dependence on computers or other devices for communication, and the horizontal axis shows dependence on simultaneous participation. The resulting chart is divided by the axes into four distinct quadrants, similar to the two-dimensional distinctions drawn by Shneiderman [1992]. There are more dimensions which figure into the interaction of media and modality, however. A further distinction is whether the communication is coupled tightly or loosely. The notion of same or different computers adds a fourth dimension to the interaction. Cost, availability, and reliability are also factors that influence media, modalities, and their interaction. For the purposes of this thesis, the focus is primarily on the dimensions of different times (the horizontal axis) and different places (the vertical axis). Coupling will not be covered, so that the terms “synchronous” and “asynchronous” will refer to the human, not computer, perception of the communication; different computers will be subsumed into the discussion along the vertical axis.

Figure 1: Same-Time, Same-Place Chart



The upper right quadrant of Figure 1 requires a variety of electronic devices, and the temporal presence of all participants. All of these are synchronous in mode, but require increasing amounts of hardware media. Communication methods within this sector include Internet chat rooms, the Internet phone, and video conferencing. A Chat tool requires textual input devices, software, and networked computers, and permits simultaneous, text-based interaction. With the addition of auditory input and output devices, the computers can deliver simultaneous voice-based interaction through the Internet phone. The cost of adding visual presence to the communication, through video conferencing, is that of attaching additional, usually more costly, input and output devices such as cameras and video processing boards.

The lower right quadrant has less dependence on sophisticated technology. This quadrant still requires common temporal commitment of the participants, because the modality is still synchronous, but the computer requirements are lower than in the first sector. Communication methods in this quadrant include voice-based telephone conferencing, which requires little more than a telephone, and face-to-face interaction, which requires no electronic media, just physical presence.

The third, lower left quadrant in Figure 1, does not require sophisticated technology or simultaneous participation. On the left side of the vertical axis, the modality switches to some degree of asynchronous communication, and the lower of these quadrants utilizes less electronic media. Communication through voice mail is the asynchronous version of phone-based interaction; little more is required than a telephone with an answering machine. Written correspondence, which is text-based communication, requires pen, paper and a postal system. While there are no lower or upper bounds on the temporal gaps between the initial message and the reply in these different media, there is a difference in expectation. Voice mail systems bring anticipation that a reply will be made shortly after the message is received. It relies on the metaphor of face-to-face communication, or phone-based interaction. Written communication anticipates a longer delay in receiving a reply, time enough for the postal system to deliver the message, and to have a reply written and delivered. The modality of the delivery of written communication carries an implicit message regarding expected speed of reply; a faster reaction is expected when the original message arrives by special courier.

The fourth quadrant, in the upper left of Figure 1, is also asynchronous in modality, but again requires a networked system of computers. Electronic mail (e-mail) requires networked computers and support software, but is akin to traditional written correspondence, but faster. The speed of e-mail relates to the time that elapses between a message being sent, and its arrival in the mailbox of the recipient. Because of this speed, e-mail anticipates a correspondingly quicker reply than written correspondence. Many have described the universality of certain features of e-mail systems, including speed, asynchronous modality, "text-based" interaction (that is, consisting only of the written word), and the ease with which messages can be sent to a single person or to a long list of recipients [Sproull 1993, Finholt and Sproull 1993, Harasim et al. 1995]. Sproull adds two more important attributes to e-mail: external memory (defined as the ability to store and later retrieve messages), and the ability for this external memory to be processed by computers (defined as the ability to search, edit, rearrange, and relocate messages). E-mail compares very favourably to other communication media and modalities, Sproull concludes, as it is the only one that encompasses all of the above attributes [Sproull 1993].

If messages are saved, e-mail can provide teams with organizational memory. This creates an archive of discussions and of decisions made, at little additional cost for the extra storage required. The low cost trades off against the low value of this style of organizational memory, however, because the individualized storage structures that are used tend to be poorly organized [Conklin 1993], and are subject to a variety of individual styles [MacKay 1988]. E-mail is similar to verbal and postal communication in that once a message has been sent, the sender has no way of revoking the action.

Electronic mail is joined in the upper left quadrant of Figure 1 by newsgroups, messaging systems for communication among peers who are potentially spread across a vast geographical area. This permits the exchange of information in a structured and selective way. Newsgroups are occasionally called bulletin board systems, although the terms are not entirely interchangeable. While they offer a structure of message organization similar to newsgroups, bulletin boards are generally located within a single telephone calling area, and provide more services than simply communication [Harasim et al. 1995].

The implementations of these two messaging systems have much in common. They both consist of messages, creating asynchronous discussions, that can be threaded by author, by date and time of posting, or by subject of the messages. A generic version of such a system is appropriate for discussions and learning among reasonably small groups, but as the group size increases, it may be necessary to place multiple discussion areas under the umbrella of a hierarchical ordering system.

Newsgroups are currently open to public participation; some are moderated, with posting controlled by a central authority, but generally anyone can access the newsgroup through the appropriate computer hardware and software, and can then read and post messages. Postings can be read, or marked as read, on the basis of an individual message or a full thread of messages. Articles can be saved to a local disk, or printed. Because the medium is asynchronous, and archiving is centralized at a small number of computers (news services), an author who wishes to remove a redundant, erroneous, or embarrassing message before it is seen by more than a small set of newsgroup readers usually has this capability.

1.4 Computer-Supported Collaborative Learning

Computer-Supported Collaborative Learning (CSCL) is a specific sub-field within Computer-Supported Collaborative Work that focuses explicitly on education. Both fields are encompassed by the field of Human-Computer Interaction (HCI), which strives to understand the contexts in which a computer system will be used. McGrenere [1996] surveys key HCI design considerations for CSCL.

The concepts behind machine-based tools for education have existed for decades. In the 1950's, some researchers pioneered a "teaching machine," based on immediate, positive reinforcement of successes. The simple, mechanical machine contained textual "programs" to take students through a series of drills. This was an early realization of the concepts of student control over the pace of learning, and of optional links to more detailed "sub-programs" [Price 1963]. These two concepts remain key features in modern, computer-based education applications, although the emphasis has shifted away from rote content drills, into delivering the content itself, and providing facilities for interaction with other students and instructors.

Chapter 3 discusses applications that deliver educational content in an on-line environment, showing that they include several communication media and modalities that are discussed above. They generally provid asynchronous interaction, usually a newsgroup-like facility, or an e-mail facility. Some applications include synchronous communication, such as chat systems, or even video-conferencing tools. By providing these communication facilities, on-line systems encourage knowledge creation through collaborative learning.

The term "collaborative learning" applies to activities and environments in which a group of students and instructors cooperate in order to create an improvement in new or existing skills, explore subjects, and synthesize their findings in order to create meaning; their rewards are based on the accomplishments of the entire group [Harasim et al. 1995, Slavin 1980]. To be successful, the social construction of new knowledge is accomplished by the active interaction of all participants [Hiltz 1995], through cooperation and evaluation. Such collaboration is held to be a necessary part of problem-based learning situations, which present the students with situations for which they must seek a solution, rather than with a section of material they must master [Hmelo et al. 1995].

Such a learning environment is clearly not restricted to an on-line, computer-based situation. To create the proper collaborative atmosphere in the classroom, without any requirements for computer support, Hymel, Zinck and Ditner have suggested five necessary elements [Hymel et al. 1993]. First, there must be positive interdependence within the groups, where individual success is only possible if there is also group success. Second, there must be the concept of individual accountability, insisting on a fair division of labour within the group, and all members of the group learning the necessary material. Third, there must be a level of social skills training, to facilitate cooperation within the group. The fourth element is for members to evaluate the group as a whole, to identify what needs to be improved internally. Finally, the opportunity for face-to-face interaction is one of the strongest means of forming the positive interdependence referred to above.

The first four components mentioned above apply as well to on-line environments as to the traditional classroom setting, which was the focus of Hymel et al. [1993]. There is nothing in the interdependence, accountability, opportunity for training, and opportunity for self and team evaluation, that precludes the same principles from being applied to computer-based learning environments. The fifth element, however, is lacking from the collaborative setting when the entire course occurs through computer-based communications. When there are no face-to-face meetings, they obviously cannot play a key role in the formation of the group. An intriguing question that shall not be addressed in this thesis (refer instead to the CSCL and CSCW literature) is what the impact on collaboration would be when this fifth element is absent.

Regardless of any emphasis on collaborative or individual learning, one of the best metaphors for on-line course-work is the 24-hour classroom [Stoloway and Guzdial 1995]. By freeing students from the constraints of being in the same place at the same time, such an environment empowers them to read the material or participate in discussions at their leisure and convenience, and to reflect on what is being read before replying or asking questions.

There are three forms that a course delivered on a “learning network” can take: adjunct mode, mixed mode, and totally on-line [Harasim et al. 1995]. In adjunct mode, which is currently the most common of the three, computer-based materials and communications are optional. This often takes the form of an optional course newsgroup, in which students can discuss relevant issues and ask questions. By responding to questions in a publicly accessible place, the instructor reduces the need for repetition, as several students with the same question can all read the answer in one place. Mixed mode, on the other hand, has the on-line component of the course constitute at least a portion of the graded or required activities. This may take the form of a required discussion group, or self- or group evaluations. The remainder of the course, however, is held in a traditional face-to-face setting. On-line mode, lastly, places the activities primarily within a computer-mediated environment; the course may have no face-to-face interaction. Some would caution, however, that a virtual environment, a computer-based setting for course work, functions best as an extension to physical interaction [Guzdial and Weingarten 1995].

Harasim et al. [1995] identify five properties that are attributable to current computer-based discussion areas, many of which are similar to the e-mail attributes described previously. They are independence of time, independence of place, group-based communication, mediation of the messages by computer, and the textual nature of the messages. (It has been noted, however, that while voice-based interaction can be an order of magnitude more wordy, it is twice as fast as text-based interaction [Chapanis 1975]). A survey conducted by Harasim and Yung [1993] identified several positive ways in which computer-mediated communication differs from the traditional classroom. These include:

- The teacher becomes a mentor, assuming the role of learner on occasion, because the students teach and learn from one another as well.
- Student participation increases, because discussions involve more students who delve more deeply into the material, with an overall increase in communication.
- Students become more independent, and self-paced. There is greater equality, both in learning opportunities and in access to the teacher and other resources.
- There is more time for reflection, with participants encouraged to read messages or material, ponder what they have read, and exchange replies and other ideas.
- Active learning through participation, reflection, and discussion in asynchronous settings avoids the limitation of one-speaker-at-a-time that exists in traditional classrooms.

The Harasim and Yung study did not stop with only the positive aspects, but highlighted several negative ways an on-line environment affects the learning process. Some of them are almost direct opposites of some of the positive aspects, indicating the nature of the trade-offs; obtaining an advantage or a new feature often comes at the cost of losing something else. The negatives identified by Harasim and Yung include:

- Greater time commitment to prepare for future topics, and to remain current in the on-going discussions by reading, reflecting, and sharing. This applies to both teachers and students.
- Participation must be active; a message must be sent to establish one's presence, reducing the possibility for passive observation.
- Anxiety may increase, as a result of an overload of information, difficulty in navigating through the on-line system or the messages, loss of visual cues in communication, and discomfort from the delay between message and reply in an asynchronous environment.

An on-line individual learning environment can be patterned after several conceptual models that guide the structure of the course, as the instructor designs it. For all the above benefits of student participation in the learning processes, some facts do need to be related by the instructor. These might be delivered on-line, in the form of an electronic lecture. In computer-based systems, the educational material may include sounds and images, or be plain or hyperlinked text, the latter of which will allow students to follow their own path through the textual material. All of the following models are discussed in greater detail by Harasim et al. [1995]:

- "Ask an Expert" gives students first-hand, current information about subjects.
- Mentorship enables students to gradually master a given subject or task under the apprenticeship of a more knowledgeable professional.
- Tutor support enhances instruction coming from another source.

- Three models follow a student-centered pedagogy: student access of on-line resources of their own initiative, interaction with other students over a common interest, and group activities as part of the curriculum.

An on-line learning environment that encourages group and collaborative learning can also be patterned after several models. The models they follow reflect many found in non-CSCL situations. On-line seminars, small group discussions, role playing, and debates function in much the same way as their off-line counterparts. While greater participation is possible, the activity must stretch over a somewhat longer period of time if applied in an asynchronous modality. With more time to reflect on the issues, and to frame their replies, student debates and role playing can be more detailed, as students have time to seek support for their side or their choices, enabling a better final synthesis of the information. Peer learning groups, of two or more students, can be established. These groups can be used to discuss important concepts, and study or work together on a specific project. A "Café" model provides encouragement for students to interact socially, as well as intellectually. These models and more are discussed in detail by Harasim et al. [1995].

To be successful as an on-line environment for group work and collaborative learning, a system should take into consideration the dimensions of group problem solving and of human communication. Groups experience stages of recognition, exploration, and idea formulation about the problems they are to solve. They break a problem down and evaluate each component of it, before determining a solution. Groups achieve these various stages through communication, which involves some aspect of cooperation, intensity of interaction, leadership dominance, or lack thereof, and degree of formality [Turoff 1993].

To succeed as a collaborative communication tool, a system must enable all of these aspects; the most significant problems faced by groups are sociological, rather than technological. The success or the failure of the efforts of a group rest, more than anything else, on the quality of their interaction [DeMarco and Lister 1987]. However, to succeed as a collaborative learning tool, beyond merely facilitating interaction, the tool itself plays less of a role than the individual participants. Collaborative learning is most successful for those students who exhibit the motivation to learn, have the self-discipline to participate frequently, and have sufficient access to the system [Harasim et al. 1995].

The focus in later chapters is on a Software Engineering course in the Department of Computer Science. McConnell [1996] has applied basic active learning techniques, such as small discussion groups, role playing, and debates, to a computer science lecture course. Baram and Mandviwalla [1996] also offer several examples of collaborative learning in Computer Science, as do Yerion and Rinehart [1995]. They identify time constraints, clarity of expectations, group dynamics, and student resistance as key problems to be addressed.

1.5 Moving On-Line

The remainder of this section provides a brief discussion of moving a course into an on-line environment, as synthesized from several sources in the literature. Our specific experience with CPSC 319 will be discussed in greater detail later; Chapter 2 describes the course, while Chapter 5 relates the lessons we learned.

The focus of the preceding discussion has been primarily on collaborative education, in a cooperative learning structure where the success of the individual student is tied to the success of the group as a whole in achieving its goals. By providing faster and asynchronous communication tools, on-line educational applications enhance the potential for collaborative learning, as we have seen. There are also two additional learning structures in the regular classroom, for consideration for an on-line environment [Hymel et al. 1993]. The opposite of the collaborative pedagogy is an individualistic learning structure, where students work alone, and are responsible for achieving their own goals. In such an environment, each student works, and succeeds or fails, independently of others in the class. There is also a competitive learning structure, where individual (or even group) success is weighed relative to the other students (or groups) in the class. Success is no longer completely independent, but rather some aspect of course achievement is a zero-sum game with open competition against one another. There is room, in a telelearning environment, for all three of these learning structures. We have seen the communication benefits that such a system provides to a cooperative situation. A class that focuses on individual work and achievement is also possible. For example, the course content can be delivered through Web pages and multimedia presentations, coupled with individualized quizzes, assignments, and examinations. If communication facilities are used, they might support class-wide discussions, or question-and-answer sessions with the instructor, but not be employed in any team work context. An example of a competitive structure is a computer-based marketing simulation, with teams learning about business strategies by competing for market share (a metaphor for course marks) of a mature sector of industry. The market share is a zero-sum environment, where the gains of one simulated company come at the expense of another. The expectation is that direct competition might heighten the interest in that particular unit of the course.

Learner-centered courses prove to be the most effective candidates for succeeding in an electronic environment [Harasim et al. 1995]. Such courses already emphasize interaction between students, encouraging them to share ideas and insights, and to work together to build knowledge. They are therefore best poised to capitalize on many of the advantages offered by computer-mediated communication. The flexibility of communication is one key advantage; it is available twenty-four hours a day, it supports individual and group work on several projects at a time, and it can support a variety of potentially shifting human roles. Good communication systems are tailorable to the needs of specific individuals and groups, and they can be structured to reduce information overload. The educational content, presented primarily in text form, facilitates the delivery of a large amount of material electronically, making a telelearning environment ideal for courses that cover topics in either great depth or great breadth. Supplemental, non-textual material may need to be supplied, but the text and any related discussions are easily moved into an on-line environment.

Situations in which computer-based information and resources are important also convert well to an on-line setting [Johansen and Bullen 1984]. With extra material and effort, it is possible to hold courses on-line that one might not expect to find there, such as second-language courses or an art appreciation class [Harasim et al. 1995].

There are some cautions to be raised as well. When a course moves into a telelearning environment, it becomes largely text-based. Media that emphasize textual information are most successful when all participants in the class are comfortable with the computer keyboard, and when the assigned tasks can easily be adapted to a text-only communication format. While text-based communication is helpful in group situations in which participants have difficulty finding common meeting times, there are significant differences in the ways groups plan, sub-divide, and perform their tasks in an on-line setting, versus a face-to-face setting [Eveland and Bickson 1988]. Electronic groups experience delays at the start of the task by the additional requirement of learning the communication tools, although they hope ultimately to reap benefits from ease of coordination and increased team participation. Several authors caution against simply automating existing tasks and communications [Finholt and Sproull 1993, Turoff 1993]. With respect to communication in particular, greater benefits will be realized if we move beyond merely gaining more efficient communication, and into the creation of group organization, structure, and processes that take full advantage of asynchronous, text-based communication.

Of equal importance to the logistical concerns are the course-specific steps to take. The components of the course that are to be delivered over a network must be identified, and computer-based resources for these units must be found or created. The curriculum may need to be restructured, and the resources must be arranged accordingly. The instructor takes on a variety of roles in an on-line setting, many of which are similar to those in cooperative learning environments. Rather than being the source of information and grading, the instructor becomes a guide and a mentor, introducing the students to the specific set of expectations and guidelines for a particular course, on a particular system. The instructor monitors levels of participation in the class, encouraging discussion and offering suggestions or answers to questions that arise. It may be necessary for the instructor to establish what minimum level of participation is acceptable for the course. If there is group work in the course, instructors may find it will proceed more smoothly, and produce results more quickly, if groups and roles are assigned, rather than worked out asynchronously by the students themselves. Evaluation of the success of an on-line course, and of the enrolled students, can be accomplished through assessment of the participation of the individuals, including the content of their messages and of any spontaneous comments that are offered, and of the quality of their assigned work. Usage data such as the number of messages sent by one student, or the number of pages of content accessed by another, provide insights into the amount of participation, the degree of equality of that participation, which units were most difficult or most interesting, and which require alteration or clarification [Harasim et al. 1995].

While there are no guarantees for creating a successful on-line course by following these or any other guidelines, there are four aspects that increase the likelihood of success. These are identified as vital elements of the Virtual Classroom [Hiltz 1995] (discussed in Chapter 3), but are applicable in general. The first of these is richness and variety of media, with skillfully written text, and activities that encourage participation and collaboration. Second, a clear organization, which establishes expectations and a regular schedule, allows the students to see the course as a whole, rather than independent units, making it possible to schedule their own interaction with the material ahead of time. A third element is the level of interaction among the students, and between students and the instructor. Encouraging this participation is important to achieving the full benefits of collaborative learning. Finally, the instructor must interact with the students in their discussions, and answer questions on a regular basis. This may require a commitment on the part of the instructor of up to an hour a day, every day, but frequent, timely replies will avert disillusionment with the on-line setting.

Ultimately, the success of the computer-based class, whether cooperative or individualistic, whether entirely on-line or mixed with traditional classroom instruction to some extent, hinges on the motivation of the students. The traditional classroom style has been found to be most effective with students who are self-motivated to learn the material [Guzdial and Weingarten 1995]. Correspondingly, on-line education is most effective among students with high motivation and ability [Hiltz 1995], since motivation is one of the key factors to success in the networked classroom [Harasim et al. 1995]. The benefit of the on-line system, then, is the increase in the motivation to participate and learn that occurs among a greater number of students [Eisenberg and Ely 1993]. The instructor is able to influence this motivation somewhat, by allaying fears of the system, presenting course content in an interesting way, and encouraging broad class participation. The communication tools, both synchronous and asynchronous, offer opportunities to increase the sharing of information and ideas, enabling students to create knowledge and synthesize conclusions that go beyond simply memorized information, but are conclusions that are truly the students' own.

Chapter 2 -- CPSC 319 - Software Engineering Project

The Department of Computer Science at the University of British Columbia offers a third-year group project course in software engineering, "Software Engineering Project (CPSC 319)." Students are organized into teams, to create a large software system through the phases of design, implementation, testing and maintenance. I will refer to the course as CPSC 319.

2.1 Thumbnail sketch of CPSC 319

The course is delivered in a university setting, targeting third- and fourth-year undergraduate students. It is a required course within the Computer Science curriculum. The class has approximately 150 students enrolled each year. Students are organized into eighteen to twenty teams, ranging from six to ten students in each team.

To ensure that a proper amount of time is available for each of the phases of software development upon which the course touches, the class lasts for about seven months, from September to early April. This is intended to provide the teams of students with enough time to get to know one another, discover individual and group strengths and weaknesses, identify the system requirements, propose a design of a solution, and finally create, document, and test their system. The course may also include a maintenance and enhancement phase, with students building changes into their own or another team's system, but this has not been included every year that the course has been offered.

Recently, the project has involved coding an airline reservation system (ARS), which recognizes multiple levels of user access. The "travel agent" access level is able to query the current status of flights and passengers, and to book and cancel customer flight reservations.

The “Supervisor” access level has authority to cancel reservations by other agents. The “System Administrator” level of access is capable of entering or removing from the system information about cities, aircraft, flights, and users. Each team must create documents for the requirements analysis, based on descriptions of the functionality that are provided by the instructor, and through interaction with the course staff to gain further information about the problem. From their analyses, teams draw up a design plan and a project schedule, outlining the effort required in terms of number of hours and allotment of team members to tasks. The final, implemented system consists of a graphical user interface, serving as the front end to the airline reservation system, and a back end that interacts with a relational database system to store and retrieve the data.

There is only a single hour of lecture time scheduled each week for CPSC 319; these are the only times the class is physically together as a whole. The students are expected to enter the course having a basic grounding in software engineering methods and terminology. To achieve this grounding, the Department’s course on the principles, techniques and methodologies of software engineering (CPSC 310), or its equivalent, is required as either a pre-requisite or co-requisite course, taken no later than during the first half of CPSC 319. By leaving the majority of the instruction that is specific to software engineering to the pre-requisite CPSC 310, CPSC 319 is freed from the constraint of teaching detailed software engineering theory. The CPSC 319 lectures, then, are less frequent, and cover only material related to the project, such as specific software tools, group dynamics, running effective meetings, and other concepts useful for the completion of the project. There is enough slack in the schedule to allow freedom for the lectures to respond to specific concerns that arise in the various phases of the projects.

In addition to the weekly class lectures, each of the project teams meets separately for a minimum of one additional hour each week. This time is reserved for the team members to share their ideas, discuss advantages and problems, ask questions, relate difficulties or progress, and make decisions. This time is also available for meeting with the teaching assistant, who may provide comments or guidance, clarify course administrative issues, offer advice, and observe the current status and progress of the team. The composition of the project teams is assigned by the course staff, based upon student self-assessment in key skill areas. To this end, the students complete an experience survey during the first week of the course, to identify the level of skill and experience of each student in programming, graphical user interface design and implementation, relational database technology, Web-based publishing, and team leadership. This questionnaire is completed in the first week of class, and is included in Appendix A. A simple heuristic algorithm uses the results of the surveys, to ensure that each team has a relatively comparable collection of talent, so that no team is placed at a disadvantage in any of these areas.

When registering for the course, students are presented with several different tutorial times. Their selection of tutorial time further affects the formation of project teams, in that teams are formed only from people who have registered for the same tutorial time. This administrative-logistical constraint guarantees that there will be a common time during the week when all students on a team are available for a team meeting.

The project, a large, non-trivial software system, is completed in multiple milestones or deliverables. These assignments constitute a set of project artifacts, upon which are based assessments of the design, analysis, implementation, testing, documentation, project management, and meeting effectiveness of the teams. The deliverables are generally on a bi-weekly schedule, which encourages the project to continue advancing toward completion, and provides the TAs with sufficient time to deliver helpful comments on previous deliverables. As each milestone is reached, students are requested to provide the course staff with self-evaluation reports. These include indications of the work scheduled for each student, for both estimated and actual effort required. Ideally, reports also include justifications for any discrepancies between estimation and reality, and their assessment of the work and effort of other team members. These reports serve as extra information for the teaching assistants, in tracking the progress of each team, enabling them to detect such problems as slippage in the schedule, or brewing conflicts.

The main role of the course instructor is to provide guidance, information, administration and coordination. The pre-requisite software engineering course teaches technical concepts, principles, and methodologies. In CPSC 319, students apply the knowledge garnered in the pre-requisite course to a team setting and a non-trivial project. The instructor must guide the class, as the teams learn to work together to achieve their common goals, as issues and difficulties arise, and as plans are laid for the completion of the project. As the primary lecturer, the teacher must instruct the students about forming and working well as a team, resolving internal conflicts, and running efficient and effective meetings. Software engineering methods and principles, and some aspects of coding and testing practices, which are expected to have been learned already, are reviewed. The course staff provides access to a sufficient range and number of tools for the teams to complete their projects. While it is reasonable to expect tools such as compilers and programming languages to be learned prior to CPSC 319, the instructor passes along an adequate amount of information regarding GUI toolkits and database systems to the teams for them to succeed.

Certain difficulties arise repeatedly within the course structure, such as schedule slippage, and implementation issues dealing with networking or security. The standard project, an Airline Reservation System, raises issues of client-server distributions (running a single database back end with multiple terminals simultaneously running a graphical front end), multiple simultaneous connections (for a single user), levels of access to system functionality (agent, supervisor, and system administrator), and chains of command (only the agent and any direct superiors can undo agent reservations). When these issues arise, the instructor serves as a primary source of assistance and supplemental information for the teams. The instructor is also cast in the role of chief administrator and coordinator. This involves coordinating the activities of the TAs, ensuring they are kept sufficiently busy but are not overwhelmed by marking, and ensuring they are sufficiently informed about the course structure and topics to be able to assist the teams in the weekly meetings. The TAs are responsible for booking rooms for the team meetings, but coordination of team presentations, and access to the tools are the responsibility of the instructor.

The teaching assistants for CPSC 319 are responsible for three to five project teams. They are to become familiar with the teams and their members, ideally remaining with the same set of teams for the duration of the course. They are responsible for grading the completed deliverables, monitoring team progress, providing advice, comments, and guidance, and answering course questions. They represent the main contacts the project teams have with their “clients,” offering suggestions about functionality, usability, and other issues.

The unique properties of CPSC 319 served as the underlying parameters motivating this study. Computer-based systems for on-line educational content delivery often stress the benefits of collaboration in learning and understanding [Hiltz 1995, Harasim et al. 1995, Hmelo et al. 1995, Guzdial et al. 1995]. In this respect, CPSC 319 appears to match the goals of these systems very well, because it is predominantly collaborative group work. However, as will be seen later, the very nature of the course presents problems and considerations for any on-line package which attempts to include all learning situations in one system; CPSC 319 emphasizes a set of requirements which sets it apart from courses based more upon a traditional classroom or lecture setting.

2.2 History

CPSC 319 has been offered for seven years, and has evolved over that period, in scope, tool set, and emphasis. The course was initially modeled loosely on the "Software Hut" concept [Horning and Wortman 1977, Wortman 1987]. In its evolution, changes have been introduced in response to a changing industry, student suggestions, departmental requests, and new ideas. Through all the changes, the main goals have been to give the students some experience, often with industrial-strength tools, in applying software engineering methods to a large project in a team setting.

Three years ago, in response to increasing demand for exposure to tools currently being used in industrial settings, the Oracle suite of products was introduced as the database management system with which the projects would interact. Prior to the introduction of Oracle, the project teams were required to carefully plan and construct a detailed internal data structure, in which to store the cities, flights, reservations, and other data items within the system. This necessitated designing the structure, and coding the procedures that would successfully insert new data, accurately extract it, and traverse the data set, seeking information in reply to a given query. With the introduction of Oracle, data storage moved into the Oracle database management storage system. Responsibility for data integrity, writing, retrieving and searching, was taken out of the hands of the students and given to Oracle. The tradeoff for this reduced workload was the new requirement of learning enough of the SQL programming language to be able to tell Oracle what to do and when.

There was a significant alteration in the focus, structure, and tool set of the course in the past year. Prior to this, the course was structured so that all teams completed the same project, on roughly the same schedule, with the same milestones and due dates. This structure allowed the staff to control the directions and workloads of the groups. One day a week, a bulletin would be released, summarizing the current scenario and situation, offering some suggestions for discussion topics to add to the weekly meeting agendas, and alerting the students to any “surprise” changes they were required to deal with. The sequence of the course was as follows:

- Milestone 1: The students were given a set of bug reports, corresponding to a text-based airline reservation system having the same basic functionality they would eventually be expected to provide. They were also given a header file (referred to as code.h) that defined the expected functionality and how the interface communicated with the data storage back end. Their task was to verify each bug report, and either show it to be true and identify the nature and probable cause of the bug, or show it to be incorrect, through black box testing methods.
- Milestone 2: The teams developed a design for a graphical user interface (GUI) for the airline reservation system. The design was to include state transition diagrams and paper mockups of each screen in the GUI.

- Milestone 3: Teams created an alpha version of the GUI. The course provided the Simple User Interface Toolkit (SUIT) for creating a GUI, because it was a system that could be learned quickly.
- Milestone 4: Each team continued to refine its GUI in SUIT. Teams were also required to test and critique the GUI of another team, as specified by the course staff. Ideally, a cycle would be created, one that could iterate two or three times: test the GUI, write a critique, provide feedback to the other team, receive their updated GUI, and repeat the cycle.
- Milestone 5: By the mid-point of the course, each team was to have a full, working GUI as a front end to the system.
- Milestones 6-8: Various components of the design documents were delivered, for the full and integrated system - GUI and back end. These were to include module specification guides, P-specs, C-specs, data dictionary elements, module implementation guides, data flow diagrams, and more. Examples of some of the actual work submitted by the teams are included in Appendix B.
- Milestone 9: Teams were to deliver a working version of the full system, with the database back end working with the GUI. This was an alpha version of the final system, prior to full system testing.
- Milestone 10: Each team continued to refine their system. Teams were also expected to test and critique the systems of one or two other teams, as specified by the course staff. As in Milestone 4, the staff hoped to create a cycle that would iterate a couple times: test the system, provide feedback, receive the updated system, and repeat the cycle.
- Milestone 11: The final milestone was to complete the integrated, tested, and fully documented airline reservation system.

During this sequence of milestones, teams were asked to adapt to wrinkles and surprises as they were introduced. Minor functionality enhancements, such as restrictions on the minimum time between arriving on one flight and departing on another, would be announced, and teams would be expected to include it within the final system. At the mid-point of the courses, project teams were shuffled, with the new teams again assigned by the course staff. Rearrangements were kept within a TA's set of teams, so that the knowledge of individual students gained by the TA in the first half would remain applicable in the second half of the course. Shuffling was arranged so that each person was paired with at least one other member of their past team in the new setups. The ideal situation was for a TA to have four teams of eight. Teams would then give two members to each of the other three teams, and keep two for themselves. The results were to again have four teams of eight students, with no more and no less than two students from any single previous team. The reality was a little more messy, as teams ranged from six to ten members, and occasionally a TA did not have four teams.

Another situation requiring student adaptation was the introduction of "industry standardization." Initially, teams were free to modify the module interface header file (code.h), between the GUI and the database back end. At one point, all teams associated with a given TA were requested to send two representatives to standardization meetings, in order to identify a common module interfacing standard. With this standard implemented, a TA could potentially compile the GUI of one team with the database system of a second team into a working ARS. Several weeks later, teams were asked to create a class-wide standard. Certain minimum requirements were insisted upon within the standards. The basic, course-defined functionality had to remain, such as the system feature set, multiple GUI connections and multiple user connections. Other minor functionality changes were introduced, changing from year to year, but the above represent the most significant ones.

For the 1996-97 academic year, an attempt was made to reduce the workload of the course, by rearranging the course structure, and changing the emphasis of the material. Teams were given more freedom over their workload and schedule of deliverables. The course began with the students examining an old, faulty, text-based ARS, as in previous years. But this time they were asked to use their observations, and the code.h interface header file, to perform design recovery of how each required function operated. For each operation, such as creating a new reservation, they applied techniques taught in the co-requisite CPSC 310, and reviewed in lecture, to identify what information was needed, and what the proper responses were under all possible conditions. These were written at a high conceptual level, to be independent of any implementation. Following that assignment, each team spent the remainder of the first half of the course preparing a project proposal, and learning about group dynamics and how to run effective meetings. Lectures provided the theory, team-building exercises with the TAs provided practical experience and guidance, while a component of the final grade provided motivation. One lecture discussed personality types, explaining how different personalities might create conflict within the groups, and how to deal with the conflicts. Another reviewed six keys to facilitating effective and productive meetings.

The proposal document was presented orally to the course staff prior to its written due date. It contained an evaluation of the skill set of each team member, a detailed description of each intermediate product the team would deliver, a schedule of deliverable dates, and a list of team members involved in each intermediate product. Also included were estimates of the number of hours required for each team member on each project phase, the relative worth of each deliverable toward the project mark (expressed in dollars), and justification of the choice of development strategies (rapid development, object-oriented, waterfall, or other). Unlike the previous year, each team had a different schedule of deliverable due dates, and different values assigned to the deliverables.

One aspect of the freedom for the students to propose their own project work load was that they could choose from a broader selection of implementation tools. Course materials and support were provided for SUIT and TCL/TK as GUI toolkits, C and C++ as primary programming languages, and Oracle as the database system, but teams were free to reach beyond those tools, if desired. Teams were also not required to complete the entire system, but could choose to only implement a GUI or only the back end (Reservation Management System). Teams implementing only half the system were required to show more thorough designing and testing. The networking requirement was emphasized less, so that most teams aimed to implement a standalone system, with only one possible user at a time. The concept of multiple connections did surface at one point, however, which will be discussed in Chapter 5.

One notable difference between last year, and preceding years, is in the grading schemes. In 1995-96, grades were issued for each student at each milestone. Each student received three grades: a team grade that was common to all members, an individual effort grade based on participation and effort, and an individual achievement grade based on the quality of the work of that student within the deliverable. Each grade counted equally over the eleven milestones, to give a straightforward final percentage. There were some subjective judgments on the part of the TAs, particularly in assigning individual effort marks, but ultimately the grades were divided as one-third team and two-thirds individual. This year, the team grade was increased to approximately 50%, and the individual component of the grade was correspondingly lowered. Subjectivity was reduced within the individual grade, through the introduction of individual quizzes and assignments, although a component still remained at the discretion of the course staff, based on student participation, effort, and teamwork. Half of the team component of the grade was accounted for by the actual project, including design, implementation, and testing. The other half was awarded for the team's project management skills, as demonstrated over the course of the year. To assess the latter component, TAs periodically performed meeting evaluations, and graded bi-weekly Web-based deliverables, wherein the teams were required to keep a record of their meeting agendas and minutes, rosters, task assignments, and up-to-date time charts, such as GANTT charts, showing their progress relative to their proposed or revised schedules.

2.3 Groups

CPSC 319 is entirely team-based; while students do receive a portion of their grades from individual performance, the primary thrust of the course is the teamwork. The concepts of collaborative work, and group projects, are not new in an educational setting. There is a substantial body of literature, in the fields of Education, and CSCL, regarding the benefits of collaborative and cooperative learning models, structures, and activities. (As a small sample, see [Hymel et al. 1993, Harasim et al. 1995, Hiltz 1995, Baram and Mandviwalla 1996, McConnell 1996].) It must be noted, however, that collaboration does not equate precisely with the work of a group. Collaboration involves individual agents contributing their share of the effort on a common project. In a collaborative learning environment, students share their insights, thoughts and opinions, their questions and answers, with one another, and learn in the process. This collaboration can take the form of small group discussions, partnerships, group projects, team presentations, debates, and other forms of social interaction, which are possible both in an on-line and a face-to-face environment. A group goes beyond simple collaboration, although there are similarities. A group is formed around a set of goals or objectives that are common to all members; when the group achieves such a consensus, the sum of the members is greater than the individual parts. Collaboration is people working and learning together, while a group is a collection of people bound together more tightly than is necessary for mere collaboration.

Groups benefit by blending the strengths and weaknesses of each member, incorporating them into a whole that, true to the standard cliché, is greater than the sum of its parts. Like a good handbell choir, team members support and build upon one another, making better music than any one of them could create alone. To form a team that surpasses the potential of any one individual of the group, the members must develop solid teamwork habits. Participation, dedication, accountability, constructive and open discussions, and flexible leadership are all key factors to successful group work [Ferraro, Rogers, and Geisler 1995]. The members of the group complete their work, individually and collaboratively, while coordinating their efforts through team meetings and other activities. In the traditional setting, this involves face-to-face meetings, while in a networked environment, several modalities exist that can be employed [Ferraro, Rogers, and Geisler 1995]. Groups are more effective over a longer term if they can develop a corporate memory, one that provides a record of what decisions were made, and why particular choices were selected over others [Conklin 1993]. The ability to capture the answers to these questions in a collection of group artifacts becomes more important as the temporal distance between the current issue and the past decision increases. A simple solution, which can be learned and implemented quickly, is discussed in Chapter 5.

The core essence of a project team, the group communication and culture, is influenced by four attributes: task, setting, criteria, and characteristics. These attributes affect the internal behaviour and communication of the group through either direct or indirect influences [Finholt and Sproull 1993], and help to shape the internal processes within the group [McGrath 1993].

The task to be accomplished impacts the essence of the group, by affecting the division of labour, by defining the challenges to be met, and by laying out the parameters for potential conflict. The nature of the task helps to shape the internal structures of the team, for example influencing the formation of sub-teams and their composition. The size of the task impacts the size of the project team and the duration of the effort, placing workload and commitment requirements on the members, and requiring increased coordination among the separate work units. The complexity of the task also shapes the size of the project team, and directly impacts the communication requirements, to ensure the project's full scope is properly accounted for.

A common team location, with everyone sharing physical proximity, helps to form an effective team. DeMarco and Lister [1987] phrase this in the opposite direction: one of the items in their list of seven ways to commit "teamicide" is physical separation. Barring proximity, members may have other common links, all within a common larger organization, for example, that provide shared goals and a corporate culture. On-line communication, however, can alter this attribute by removing the physical commonalties, and to some extent the common temporal setting as well. The real benefits of distributed, asynchronous communication, are in freeing teams from the constraints of time and space.

McGrath [1993B] has noted, with respect to setting, that groups generally have common elements to their pasts, perhaps previous familiarity with the other team members, or a common organizational culture upon which to draw, and in which each group member is embedded. They also expect to spend the future together in some respect, again whether as an on-going group or within a larger corporate context. Their membership may vary, and their attention is divided among several specific tasks, as the members are engaged in other projects that exist concurrently and compete for attention. These group traits, he notes, differ from many academic settings of research into group dynamics. He distinguishes four modes into which all group activities fall: inception and acceptance, finding technical solutions, resolving conflict within the group, and executing the tasks required to complete the project. The processes follow several complex temporal patterns, including structuring the flow of the effort through scheduling and synchronizing, coordinating time and effort with tasks to be completed, and entrainment of effort and components for internal and external coordination.

The multiple criteria for group membership filter the set of all people down to a smaller subset - such as third-year computer science students - which affects the character of the group. This excludes other potential group members and resources that, if present, would have created a different subdivision of responsibilities or perhaps a different deadline schedule.

Finally, the individual character traits of the team members impact the essence of the corporate group. The internal composition of a project team shapes and influences the culturally accepted behaviour within the team, the internal processes, and the fruit of the team's labour, the end product. Something as seemingly predictable as the internal structure of the implemented program can be influenced by such factors as the number of people in the programming team, and their areas of relative strengths and weaknesses [Weinberg 1971]. Internal group processes that are affected include interaction and communication, influence attempts, and maintenance of the group's unique identity [Finholt and Sproull 1993].

The nature of the team, and of internal interactions, is influenced by the psychological and sociological makeup of the individuals within the team, and of the team as a corporate entity. Through the interaction of team members, the group forms its own unique set of accepted and expected relationships. The team may quickly settle into a pattern consistent with a previously defined or externally imposed hierarchy. Such a structure can be disproportionately influenced by a strong-willed, out-spoken or authoritarian figure.

Alternatively, one or a small number of visionaries may emerge, with a firm conceptual grasp of the problem, a distinct vision of the path to follow to completion, and a well-defined expectation of the end result. This group composition is akin to the one suggested by several authors, including Brooks [1982], who describes distinct roles for approximately ten people within a development team as follows: a surgeon, copilot, editor, clerk, tester, administrator, toolsmith, lawyer, and secretaries. Such a structure tends towards a more democratic culture than the authoritarian option mentioned above, but is still given its strongest guidance from the small core (the surgeon and co-pilot), at the centre of the larger group, with the clearest vision of the project.

A third group structure exists, more democratic in nature than either of the previous two, as described, for example, by Weinberg [1971]. This structure recognizes that members have distinct responsibilities, but also have distinct areas of knowledge expertise. Leadership in such a context shifts onto the person who is best qualified to lead the team through any given task or problem. In such a group, it is unlikely that the leadership will be distributed in an even fashion throughout the team, but rather that the knowledgeable members will assume the added responsibility when needed. In such a model, the members with the greater motivation, determination, energy, and self-confidence are more likely to become the leaders of the group, in function if not in title. The quantity and quality of their participation naturally affect the perceptions of the other members with respect to their competence, interest, and contributions to the common goals of the group [Hollander and Julian 1978]. Democratic group leadership is one component of the Distributed Functions theory espoused by Johnson and Johnson [1975], who see leadership as being a learnable set of skills, and being specific to given situations. They view the autocratic styles described above as being instances of "designated leadership," in which one or a small number of people are in charge of the group, people who are given authority either by the group itself or by an external agent.

2.4 Groups in CPSC 319

With almost 150 students enrolled in the course each year, we divide the students into eighteen to twenty project teams, averaging about eight to ten students in each team. In this setting, they are expected to behave as a project team, moving through all the "real" phases of a software engineering project. Similarly, they are expected to traverse all the phases of group building, moving from a collection of individuals, all independent students, through the formative stages as the team defines its group culture and the roles of all the participants within the group, into finally becoming a *de facto* working social unit. The group work is a primary course emphasis, including team building exercises, lectures on personality, proper procedures for conducting effective group meetings, and more. The majority of students had participated in some form of group work in previous courses, with group brainstorming (71% of students) and class presentations (63% of students) being among the most common.

The teams formed in CPSC 319 differ from those in a corporate setting in that all team members enter the group on a relatively equal footing. There is no hierarchy or seniority, established externally to the team, which carries influence into the structure of the group. The extent of the common past, or common organizational culture is that all members are students at the University of British Columbia, are enrolled in CPSC 319, and have completed or are currently enrolled in the pre-requisite course. All students were experienced with using computers for communication through electronic mail, with large numbers also familiar with Bulletin Board Systems (55%) and real-time Chat systems (44%). This commonality of background and lack of organizational hierarchy, however, does not indicate a homogeneous collection of students. A majority of the students (85%) fall in the eighteen-to-twenty-five age group, generally signifying that they are in their first post-secondary degree. However, the range of ages represented in the course extends into the 41-50 year range. The course includes students whose abilities and motivation range from those who will not complete the degree, to others enrolled in the cooperative education program, and up to and including future doctoral candidates. Most students in the course are enrolled in the Department of Computer Science, for their B.Sc. requirement. But the course also attracts students in Masters-level programs, and undergraduates in such programs as Commerce and Engineering. For such students, the course is not required, and they may be taking it specifically to gain experience in a given phase of a software engineering project, or in teamwork. These and other differences in experience, expectations, training, culture, personality, and more, make each team a unique combination, but without any pre-defined leadership structure. The teams have a common task: build an airline reservation system. They share a common setting and membership criteria, as students in the same course. The differences, therefore, are based on the variation in abilities, and in psychological and sociological compositions of the team.

The actual membership roster of each team is assigned by the course staff, employing the results of an information questionnaire (Appendix A), in an effort to balance the skill sets across teams. In past years, each team was required to select a group leader, to serve as the main contact between the team and the course staff, and to serve as the facilitator of the team meetings. The identity of the leader would change periodically, to balance the workload among the team members and to afford leadership opportunities to several different students. The past year saw a change in this procedure. Instead of requiring the teams to select a single leader, the groups were encouraged to find their own preferred structure and rotation of leadership. Most groups recognized the relative equality of qualifications among their members, and opted for a democratic rotation of leadership. Some teams flourished under this freedom, as members stepped forward and took responsibility for their areas of current or desired expertise. The best of these teams were able to overcome any unexpected obstacles; one team had a key member fall seriously ill and leave the project, yet managed to complete the system on time as planned. It was not a success for all teams, however. For some, the lack of a central authority meant no real vision emerged as the central concept of what the completed project should be. For others, the lack of any real or artificial hierarchy led to the sub-groups, and some individual team members, drifting apart amidst increasing misunderstanding and poor communication.

2.5 Real-World Scenario

The course simulates a real-world setting in some respects. There is an emphasis on group work, with students being collected into small project teams of six to ten members each. The teams are expected to work through the various phases of the project as an integrated unit. A non-trivial, realistic task is assigned to all teams, and they are expected to pass through all stages of the software engineering process. This requires that they wrestle with the identification and analysis of requirements, based on an existing system and its legacy code, make key design decisions, describe their proposed design and work plan in detail, properly defend and justify their choices, and finally implement, test and fully document the completed system. With a strong emphasis on project management skills, such as time and schedule estimation, status assessment, facilitation of efficient and successful meetings, and creation of project management artifacts, the course encourages the development of technological leadership skills. Giving the students the freedom to establish their own specific deadlines, tool set, and project marking scheme encourages them to reflect on the importance of the processes and activities in which they are engaging. The course instructor and teaching assistants play the role of main client contacts for each team; in our scenario, the staff represent the “CSAir” firm, seeking to upgrade its existing airline reservation system. As such, the course staff are the primary source of requirements information for the teams, although field trips to local travel agencies allowed a small group of students to witness a non-simulated instance of an airline reservation system.

The simulation of a real-world scenario breaks down, however, in several places. The main client contacts are also primary sources of technical assistance for the supported tool sets, organizing and giving the tool training sessions, and giving specific, detailed assistance when requested. Furthermore, the client (CSAir) holds an unusually acute interest in the internal project management artifacts generated by each team, and has hired nearly twenty teams to complete the same project.

The addition of the money-for-marks metaphor encourages students to think of their work as outside an academic institution, yet the metaphor breaks down when contracted money is withheld in varying amounts, to properly correspond to variations in the letter grades assessed for the work performed. Agreeing to pay one team a given amount for a deliverable, then awarding them only a percentage of the amount, based on the presence of some inconsistencies or errors seems incongruous with the nature of contract law, drawing student complaints and initially creating confusion. (Appendix D contains a dialogue taken from the newsgroup. One theme in the discussion concerns how well 319 compares with the “real world.”) Some teams assumed they needed to redo the deliverable until they achieved full marks, an activity for which they had not accounted in their schedules. Fortunately, this was not the case.

Another tension between the real and simulated environments is that the teams are composed of students, each taking a number of other courses, with differing schedules, and only two hours each week of guaranteed common time (one hour for the lecture and another for the weekly team meeting). Rather than many hours a day spent in the same location on the same project, the teams are perpetually caught in a strained situation of trying to find occasions in which supplemental meetings or work sessions can be organized for at least a subset of the members.

As mentioned previously, there is no inherent hierarchical structure to the groups - all group members enter the course on relatively equal footing with respect to experience and managerial abilities. As a result of this equality, and because the course is entirely based on group work - there is no ability to handle an individual without a team - the groups do not have the power to fire problematic members, or to recruit new ones from other teams. They must remain with the teammates they are assigned for the entire course, except when the teams are rearranged by the staff (as has happened in previous years).

A final difficulty, which separates the course simulation from a real software engineering environment, is the tendency to shorten the software life-cycle. The duration of the course is less than eight months, providing insufficient time for a maintenance period. This issue has been addressed in past years by shuffling the teams half-way through the course, and having the new teams select one of the existing projects upon which to build in the second half. The teams, in such a scenario, then need to account for maintainability, as they perform planned and surprise enhancements.

2.6 Tools

The course provides support for a small tool set that the teams are invited to employ to complete the project. The existing system is available for the students to use, to become familiar with the minimum expected functionality. A graphical front end and a database-driven back end are expected, with documentation published in a Web-based format. Source management and other programming utilities are also provided. Teams are free to go outside the supported group of tools, but must accept the risks that accompany such a decision, including increased difficulty in obtaining answers to technical questions, and no guarantee of compatibility among tools and components.

Two toolkits for the creation of graphical user interfaces are supported. The Simple User Interface Toolkit, SUIT, has been in use for several years in the course. It was created at the University of Virginia, and its goal is to facilitate learning the concepts behind GUI creation and operation, with particular emphasis on callback functionality [Pausch, Conway and DeLine 1992]. It consists of C libraries that provide a simple, basic widget set, and graphics routines that facilitate user-defined widgets and new functionality. The implementers write C code to create the widgets, then can manipulate the positions and properties of screen elements through an interactive editor. The course also provides support for the TCL/TK graphical interface creation system. It has a richer tool set than SUIT, and is geared more for industrial use than for an educational setting. It is an interpreted environment, but has the ability to interact with C functions [Ousterhout 1994].

The Oracle database management system is supported, for providing the back end storage environment. Access to information stored in the Oracle database is through SQL. The Oracle tool set includes Pro*C, a compiler that recognizes SQL directives embedded in C code, and makes the necessary translations to convert the code to standard C language commands. Thus, students can code the database functions into C procedures, and link them to either of the GUI toolkits described above. Oracle also includes SQL*Plus, a reporting environment that allows students to interactively perform queries, record inserts and deletes, and other SQL actions.

No tools are provided by the course for Web publishing; it is expected that students will be able to make use of either tools that exist elsewhere, or a basic ASCII text editor to create their on-line documents. The course has used a Web server, however, from which administrative information, tutorials, and other supplemental content items have been available. The Web server allows each team to have a private and a public home page, for deliverables, documentation, and other information.

The course provides support, in terms of tool availability and knowledgeable answers, for a variety of C and C++ compilers. Simple UNIX debuggers and the RCS revision management system are discussed in lecture, and the teams are encouraged to manage their projects with these and other utilities.

2.7 Issues

The nature and goals of the course raise several issues that remain unresolved. These issues, ultimately, do not need to be conquered, but rather must be taken into account, as future changes and opportunities arise. These include scalability, time requirements, process assessment, artifact generation, privacy, meeting support, and user interface design.

Because the course runs for the standard academic year, September to April, and is required for the computer science undergraduate degree, scalability and staff time requirements present concerns. Approximately 150 students are enrolled in the course each year. Thus, nearly twenty separate teams are required to accommodate all the students. This presents the course staff with the daunting task of carefully following the progress of each team and each individual for the duration of the course. The instructor is aided by as many as six teaching assistants and markers, in an attempt to follow the teams and individuals. A later chapter addresses this issue in an environment of on-line communication, suggesting that textual annotation may be of some assistance in reducing the time burden this places on the markers.

This close attention is important because the course emphasis is on the process, rather than on the actual end product. The final project, and all intermediate stages, count for between a quarter and a third of the grade of each individual. Of greater interest are the group dynamics and processes that are employed in the project management, and the software engineering methods and processes that are followed during the project. Important questions include whether or not the individual students learn through various experiences, and whether or not the team as a whole makes changes as needed, working towards greater unity in goals, and efficiency in process.

The course projects produce multiple artifacts, from Web-based documents, to charts and graphs, and from source code and executables, to user manuals and test drivers. Each team therefore requires the ability to store, access, collaboratively generate, employ and assess each of these artifacts. They require sufficient disk space, and adequate tools. They must be able to work off-line or elsewhere, and link to or transfer the necessary files, which may reside on off-campus computers. Teams must be able to track multiple versions of documents and code, and electronically co-ordinate their activities.

Privacy is an issue at the group level. The course is somewhat competitive, with each team feeling the need to guard their specific “trade secrets” from the other teams. It is necessary to be able to hold meetings in private, either in person or on-line. Wherever the location of the meetings, some form of support system is required for management purposes. For example, HTML documents are used for posting minutes for past meetings and agendas for upcoming ones. There is still a need for tools supporting brainstorming, voting, discussing and weighing relative merits, and other meeting activities. This year, for the face-to-face meetings, the course supplied “Meeting Kits” to each team, with paper, markers, scissors, and more. Teams were encouraged to use them to brainstorm ideas, to create interface mock-ups, and to apply creativity in their meetings. If the meetings are held in an on-line environment, brainstorming, drawing, automatic logging, and voting tools (essentially an electronic "Meeting Kit") would help participants to collaborate remotely, to better express their ideas, and to conduct their business.

A less technological issue that arises is how to encourage the teams to grasp the full potential of a graphical user interface. The students are provided with an existing text-based interface to an airline reservation system. This system is menu-driven, with the user typing the desired item number to access the different functions. Each function prompts for items to be typed, such as row number, seat position and so on. Error messages may not be intuitive or sensitive to the current context, and may require re-typing the entire input data. It is a simple system, which provides the basic functionality, in a structured - if not always intuitive - way. Each year, several of the final graphical interfaces are merely window-based and mouse-operated versions of the text-based system. They have two or three pull-down menus, to access the various functions. Within each function, individual items are typed, and the TAB key or a mouse movement takes the user to the next field. Error messages are uninformative or not sensitive to the current screen context. Several steps may need to be repeated unnecessarily. The teams that built these GUI systems have clearly not perceived the benefits available in a graphical system, minimizing movement between keyboard and mouse, minimizing typing, and creating a more intuitive interface. Perhaps more focus in lecture times would achieve a greater awareness of human-computer interface issues.

This chapter has very briefly summarized the CPSC 319 software engineering project course, by providing an overview of the course structure, information about its historical context, a discussion about the importance of group work and group dynamics, and a set of thought-provoking issues inherent in its current structure. Many of these themes are revisited in later chapters, either extending them, or addressing them and offering possible solutions.

Chapter 3 -- Existing and Emerging Tools

3.1 Tools for Course-Based Group Work

There are currently several existing and emerging computer-based communication facilities that can prove to be useful in an educational setting emphasizing group project work. Some are complete systems, geared at delivering part or all of the course in an electronic environment, including basic content, assignments, discussion areas and grading. Several of these networked educational systems are discussed below, with special emphasis placed on two particular systems, the Virtual-U and WebCT. Other communication facilities are generic and flexible in purpose, not explicitly aimed at the educational environment, but still useful for course work.

Electronic mail is a popular generic communication tool. It is sufficiently flexible to permit private conversation, one person to another, or to broadcast messages from one sender to a large number of recipients. The communication is text-based in nature. The individual messages collectively create an electronic archive of the various threads of discussion among the participants. Most systems place the onus of structuring this archive of past discussions squarely on the individual reader, requiring them to organize the messages, often into a hierarchical folder system. From an archiving perspective, this creates an inefficiency, with the same text being stored in multiple locations, arranged according to the tastes of each individual. Because e-mail creates a separate copy of every message for each recipient, it is not possible for the sender to remove an unwanted message from the archive; once sent, the words cannot be recalled, or prevented from being seen by others. The ability of e-mail to refer to past messages, even to include them in the replies, places the e-mail conversation in its proper context, relative to any previous comments. Its asynchronous nature, which is discussed in greater detail in Chapter 1, provides temporal and spatial freedom; simultaneous virtual presence is not required. The negative effect, however, is to render e-mail awkward for simulating time-efficient meetings.

Mailing lists, or List-servers, use e-mail technology to create a discussion group. Existing systems may forward each message to all recipients, or collect them into a daily digest archive, which is sent to the recipients once a day. Such a system reduces the variation between individuals in their archiving, by having all messages for a time period tied together. In the digest format, the sender can remove a message after it is sent, and before it has been seen by other recipients, by deleting it from the archive before the daily mailing.

The newsgroup discussion area is another well-known generic communication tool. In an educational setting, it is ideally suited for facilitating class discussions beyond the boundaries of lecture halls, where only one student may speak at a given time, and placing them instead in a much broader electronic context. This provides greater opportunity for discussion, from more students, and potentially increases the amount of interaction among students, and between the students and the instructor. Individuals can opt to ignore, as desired, any discussions that digress from the subject matter. The newsgroup is stored at a single central archive, or a small number of distributed archives, but does not rely on the individual readers. Unlike e-mail, this generic tool allows unwanted or inappropriate messages to be canceled by the poster, and no one else, effectively removing the message from the newsgroup archive. Within this archive, readers can list messages by subject, author, or time of post. A newsgroup, like e-mail, provides temporal and spatial freedom, but also suffers from drawbacks for group work. The same drawback found in e-mail, the inability to support time-efficient meetings, is compounded by the public, or class-wide nature of a newsgroup. Small teams working together would either find all of their messages publicly broadcast, or would require a separate newsgroup, reserved for team members, to provide some form of privacy and security. Unfortunately, such control is not a feature of news systems.

From the point of view of the receiver, there is a similarity between an e-mail message and a newsgroup post - both are collections of usually plain text, and both bear a reduced number of cues regarding the social contexts of the sender [Sproull 1993, Harasim et al. 1995, Sproull & Kiesler 1995]. This similarity allows them to be handled in the same way by the software applications. One application that capitalizes on the similarity, and takes it a step farther, is WebCard, developed at the DEC Systems Research Centre. It integrates text-based reading for news, mail messages, and Web pages. Folder hierarchies are the central organizing mechanisms in this experimental system, and the usual set of operations apply: move, copy or delete folders and contents. Mail messages are placed in appropriate, user-defined, topical folders. Newsgroups are separate, read-only folders, but an individual message can be copied into one of the topical folders. Browser capabilities allow Web pages to be displayed, and if desired, added to one of the folders as a link to the URL. Thus, the contents of a WebCard folder can be mail messages, news postings, or Web pages, integrating the three items into a single location, using a common application and interface. The arbitrary mix of modes this creates is described in greater detail by Brown [1996]. The system is only a prototype that shows the utility of such a system, and hence is not being evolved into a sophisticated browser. The integration abilities it demonstrates, according to Brown, are being developed into new browsers from Apple and Pixelogic.

Another generic, text-based communication tool that can be used in an educational setting is a real-time chat tool. The term “chat tool” denotes several tools, all with similar features but different names. These tools enable participants to talk to one or several other people, via computer keyboard and network, by typing a message and having it appear immediately on the screen of the other participants. Unlike the e-mail and newsgroup systems, a chat tool enables student teams to meet simultaneously; there is a common temporal requirement, but there is still freedom of spatial location. The conversations are not logged and stored automatically, as in the two asynchronous communication tools described above. This removes the archiving benefit of text-based communication. But other benefits, including greater potential for participation, remain. These tools were also discussed in Chapter 1.

The tools discussed above are generic ones, and are sufficiently flexible to be applicable in a variety of situations, including educational settings. They are all text-based. The advantage of plain text is its simplicity - through keyboard input, it is available to everyone with a relatively low level of technology, and can be annotated and automatically analyzed. Other support tools exist, ones that are dedicated explicitly to educational purposes. Their common traits include support for structured learning in the presentation of course materials, learning through communication, including collaborative learning and teamwork, and various forms of assessment, grading, course management, and monitoring facilities. The sections that follow discuss some of these systems, focusing on a blend of well established, and emerging ones that are gaining in popularity.

The advantage of asynchronous communication is that it allows students to approach the material and discussions at their own convenience, to take their time, and to reflect on the things they are reading and learning. The goal of encouraging students to think about the material, in order to develop a personal understanding, is central to the Computer Supported Intentional Learning Environment (CSILE). It is one of the beacon technologies of the TeleLearning Research Network, and is being developed at the Ontario Institute for Studies in Education. This is a computer-based environment for collaborative learning, encouraging participants to communally build their knowledge on top of each other's ideas and insights. It consists of networked computers, linked by a central database where students create text and graphics on instructor-defined topics. The instructor is a participant in the discussions, which are not intended to be for question-and-answer sessions with the course staff. Rather, the intent is for all participants to contribute to the growing knowledge and understanding of one another. Notes placed into the database are not intended for single recipients, but rather for all readers involved in the topic of discussion, in one physical class, or scattered across a continent. The full set of functionality of CSILE is available on Macintosh computers, although a multi-platform Web-based utility allows basic access to the database from outside, to read and create notes, and to participate in discussions. Activity information is automatically logged, enabling the instructor to track student participation. Scardamalia has published several more detailed reports about CSILE [Scardamalia & Bereiter 1994, Scardamalia & Bereiter 1990, Scardamalia & Bereiter 1989, Bereiter and Scardamalia 1992, Bowen, Bereiter and Scardamalia 1991].

The success of the CSILE system in encouraging collaboration in the development of understanding has led to it becoming the model on which other systems, with slightly different emphases or target users, are created. The Collaborative and Multimedia Interactive Learning Environment (CaMILE) is one such system, developed at the Georgia Institute of

Technology, which strives to facilitate learning in complex domains. Some of the domains to which CaMILE has been applied include engineering [Guzdial et al. 1995], and sustainable technology [Hmelo et al. 1995]. Students are able to create multimedia presentations, share them with their peers, and discuss different perspectives and interpretations. The heart of the collaboration is the NoteBase, an asynchronous messaging feature that allows students to post thoughts and comments about various topics. Students are encouraged, by a context-sensitive system prompt, to think about their collaboration in the NoteBase. When a new message is being created, students are asked to indicate the type of message they are making, identifying it as a question, comment, rebuttal, or new suggestion, among others. A separate window offers suggested phrases for the chosen type of post. The author can also indicate, to some extent, who has permission to read the post - one person, a small group, or everyone. Documents, sounds, images, video, and Web pages can all be linked to the message. A separate MediaBase contains multimedia content, grouped by topic to be covered. CaMILE also has an Electronic Book component, to allow easy links between the textbook and supplemental multimedia material, and to enable students to make and share personal annotations and insights. Refer to Hmelo et al. [1995], or Guzdial et al. [1995] for more about CaMILE.

While the CSILE system can be employed at several different levels of education, from elementary school to graduate school, the Virtual Classroom (TM) specifically targets university and college-level course work. It is being developed at the New Jersey Institute of Technology, as a collaborative learning component of the Electronic Information Exchange System research. The Virtual Classroom is a software platform on which several experiments are being conducted, seeking to improve the quality of education of students, and provide better, or at least more convenient, access. Results indicate greater equality of participation in the discussions. The main measure of student learning, grades, showed no significant improvement as a result of the Virtual Classroom, but neither did the grades decline significantly. The heart of the Virtual Classroom is its asynchronous conferencing system. Several separate conferences may be created, for individual topics in each course. These are geared for class-wide discussion and interaction. Students may choose to make anonymous posts to the conferences, or reveal who they are. A private messaging system, akin to e-mail, allows one-to-one communication among students, or with the instructor. On-line content and activities can be provided, depending on the amount of emphasis placed on the networked component of the course - it can be entirely on-line, or supplement a regular classroom setting. On-line exams and other student assessment features are also included. More information about the Virtual Classroom is in [Hiltz 1995, Turoff 1995, Hiltz 1995B].

CSILE, CaMILE, and the Virtual Classroom represent three systems geared for collaborative learning that are actively employed in post-secondary courses. Several institutions now offer degree programs, which can be completed entirely on-line. The Connected Education (TM) system, which has been operating since the mid-1980's, offers individual courses, or a full Master's degree program. The system consists of an on-line campus, a library, a book ordering facility, a social Café, electronic lectures, and even counseling [Levinson 1989]. The British Open University offers courses in mixed mode, with discussions held through computer conferencing. It has a common conference where all participants can post messages. The large class (over 1300 students) is subdivided into sixty smaller discussion group tutorials. One tutor leads discussions in each of these smaller conferences, restricted to between twenty and twenty-five specific students [Mason 1989, Mason 1990]. The EKKO system, part of the Electronic College in Norway, offers free initial access, to encourage prospective students to learn about the system and what it has to offer. Once enrolled, the discourses are held through electronic mail, an asynchronous conferencing system, bulletin boards, and directories of registered participants. Course content, administrative information, tutoring, examinations, registration, and social interaction all occur within the EKKO conferences and e-mail [Paulsen and Rekkedal 1990].

Other systems exist, which are restricted to specific geographic localities or are geared for non-university situations. The following and many more educational networking systems are described in greater detail in [Harasim et al. 1995]. SITP, the Southern Interior Telecommunications Project, links together over one hundred public schools, in one geographic region of British Columbia. It provides peer collaboration, and information resources, in a variety of subjects. Experts, topic moderators, on-line mentors, and simulations encourage student enthusiasm and participation. A national version, called SchoolNet, was launched in 1993 across Canada, with the goal of linking together all 18,000 schools in the country using the Internet. An attractive set of features includes hundreds of topical experts who moderate discussions, offer advice, and answer questions. Asynchronous areas, similar to newsgroups, allow students and teachers to discuss many topics, and access is provided to national and international library catalogs, databases, and newspapers. In the United States, the AT&T Learning Network employs a specialized electronic mail system to link together students across the country and around the world. This system is structured in a "learning circle," with eight classrooms working on a common topic, discussing discoveries with the other participating classes, and summarizing the findings in a final report. KIDSNet, the National Geographic Kids Network, is another non-university system that encourages the students through collaborative learning, enhancing their writing and communication skills as they perform cooperative, geographically dispersed experiments and share their data.

The above descriptions, of various education-related applications for individual and collaborative learning over computer networks, are not exhaustive of all available systems. Rather, these represent the most significant ones, which are directly related to education. Other software applications, focusing on other aspects of collaboration, communication, and group decision-making will be discussed in the final chapter. The next two sections describe two of these systems, the Virtual-U and WebCT, in greater detail.

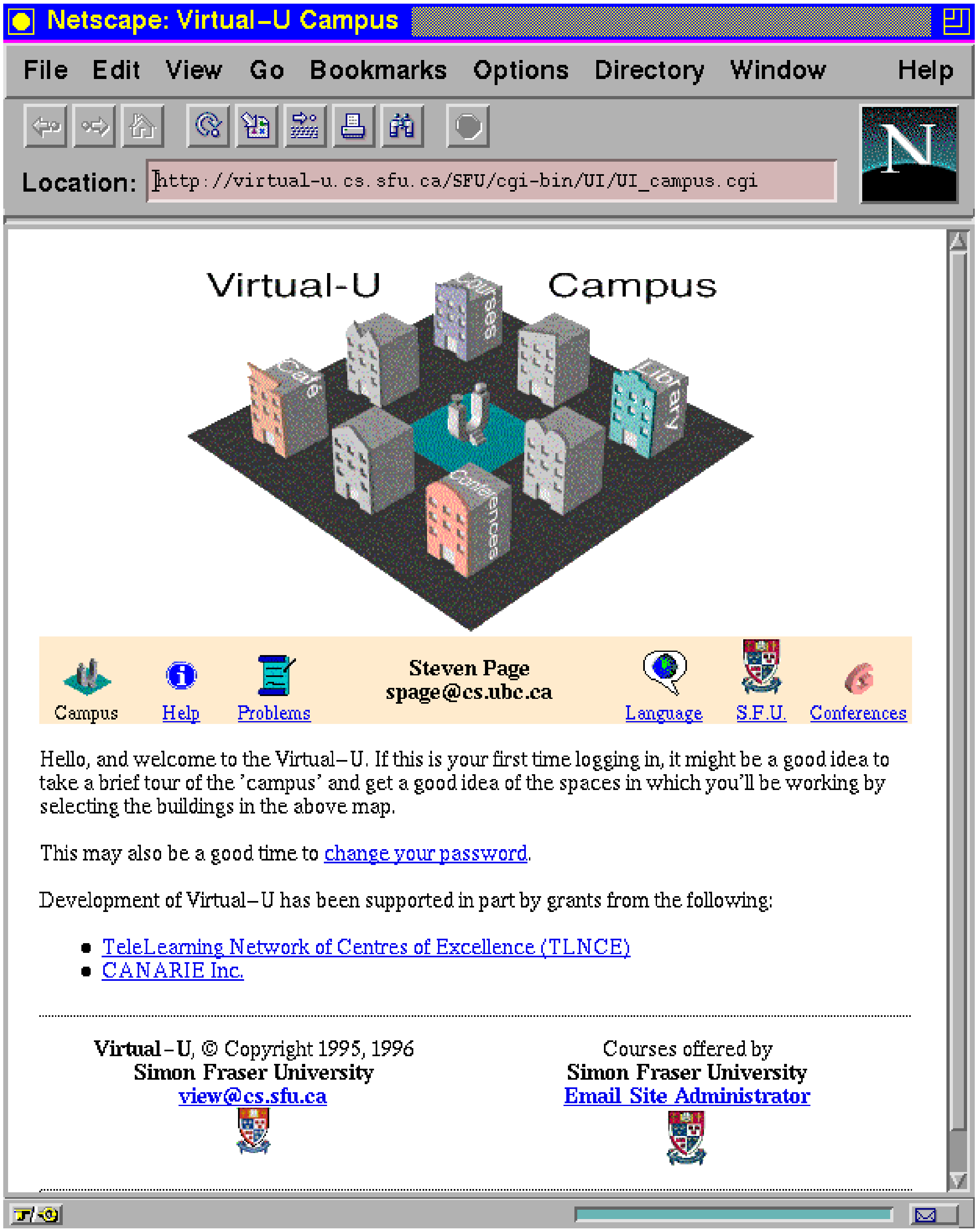
3.2 The Virtual University

The Virtual University (Virtual-U) is a software platform, based on the World Wide Web, for telelearning-based and distance education courses. It is an integrated application, facilitating the delivery of educational content, communication, and learning activities on an individual, collaborative, or mixed basis. The system is being developed at Simon Fraser University, in Burnaby (the version considered here is the beta release 1.02). It is one of the beacon technologies emerging from the TeleLearning Research Network, a collection of a hundred researchers from nearly 30 organizations and universities, distributed geographically across Canada, and thematically across disciplinary boundaries. Its development combines research and expertise from the fields of education, computer science, business, sociology, engineering, publishing, and more.

The Virtual-U encourages participants to relate to the system cognitively as to a traditional learning environment. Such a user model is encouraged through a spatial metaphor, with all features organized to evoke thought patterns similar to those on a traditional university campus. Key physical spaces in a real university correspond to a set of Web pages, which are virtual spaces in the Virtual University. A central campus map serves to orient the participants conceptually. The map presents the main destinations as separate, distinct buildings, at the compass points surrounding a central square. The buildings are three-dimensional, representational icons, with the titles written across the tops. This removes from the student the burden of recognizing the function of each building solely by visible characteristics, eliminating any requirement that the participants know the physical buildings that are represented. The map is illustrated in Figure 2.

The spatial metaphor facilitates the development of a mental model of the environment. It encourages the participants to envision social interaction as occurring in the "Café" building, and to associate research facilities and reference material with the "Library" building. The effectiveness of the spatial metaphor breaks down to a certain extent, however, with more abstract groupings of system functionality into buildings that do not have direct physical equivalents in a real university. Course-related information, assignments, and material, are available through the "Courses" building in the Virtual-U, which collapses the multiple departments and buildings in the real universities into a single conceptual place. Travel around the campus also places a potential cognitive break between the physical university and the virtual representation. Traveling between locations in the real-world environment requires leaving one building, thereby returning to the main campus, and entering the desired new building. As a Web-based application, however, the Virtual-U permits the participant to circumvent the entire sequence and arrive directly at the new location. This ability can be provided directly within the system, through hyperlinks.

Figure 2: The Virtual University Campus



http://virtual-u.cs.sfu.ca/SFU/cgi-bin/UI/UI_campus.cgi

Steven Page
spage@cs.ubc.ca

Hello, and welcome to the Virtual–U. If this is your first time logging in, it might be a good idea to take a brief tour of the 'campus' and get a good idea of the spaces in which you'll be working by selecting the buildings in the above map.

This may also be a good time to change your password.

Development of Virtual–U has been supported in part by grants from the following:

- TeleLearning Network of Centres of Excellence (TLNCE)
- CANARIE Inc.

Virtual–U, © Copyright 1995, 1996
Simon Fraser University
view@cs.sfu.ca

Courses offered by
Simon Fraser University
Email Site Administrator

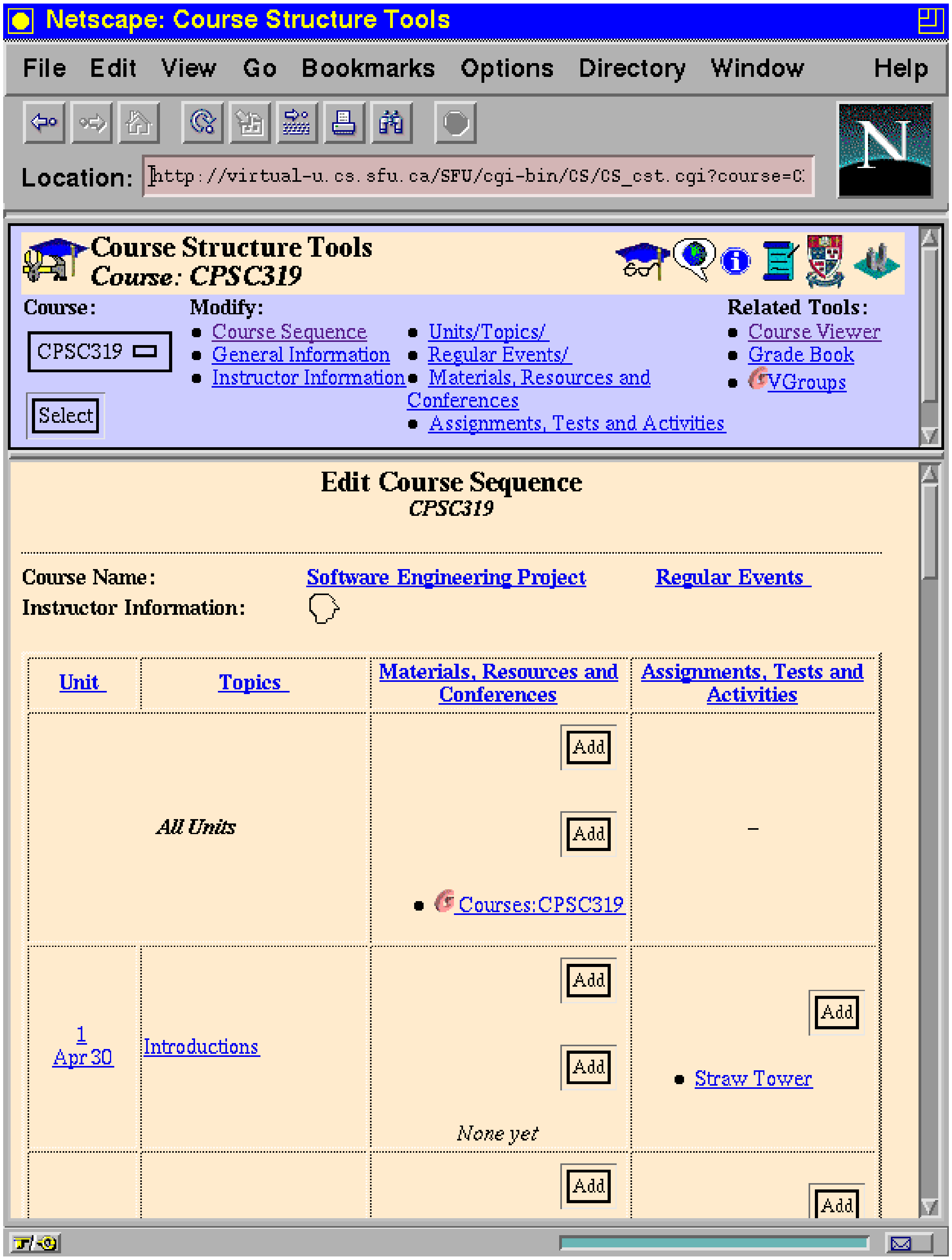
Alternatively, participants are able to establish such links themselves, as a standard feature of Web browsers, such as a Bookmark or a Favourite. These "portals" between places circumvent the spatial metaphor. Since the Virtual-U does not generally provide portals, the spatial metaphor serves to orient new participants to the Web-based system. As they gain experience, and rely less on the spatial metaphor, participants can then create their own links at their convenience.

The campus metaphor of the Virtual-U is suggestive of a university setting in particular, but the system is intended for broader use. The platform is sufficiently flexible that it is expected to encompass post-secondary education, workplace training courses, and lifelong learning. The Virtual-U comprises a complete learning environment, with a tool set for supporting course work, from the perspectives of both the instructor, and the student, on an individual or group basis. Instructors are provided with facilities for planning the course sequence, managing the course as it unfolds, and assessing the work of the students. Students are provided with primary course material, supplemental material, additional course activities, and discussion facilities.

While the Virtual-U serves as a common link among all courses on the system at a given site, the course instructors have control of the content of courses for which they are responsible. The system administrator is responsible for setting the course up with a unique identifier, through which the course is known. It is the responsibility of the instructor, however, to specify the general details. An entry screen allows the instructor to specify the full course name, and the department in which the course is offered. Space is provided, as well, for a paragraph regarding the objectives of the course, or a brief elaboration of the material to be covered. The instructor must also specify the duration of the course and the number of units involved. This value impacts subsequent screens within the Course Structuring Tool, as described below. Finally, the instructor must specify the mode of delivery of the course. The options include purely on-line, with all material presented and submitted electronically, or mixed mode, with class meetings supplemented through on-line material and discussions. A similar entry form allows the instructor to define personal information, which all courses can access. This information includes the name and department of the instructor, plus office location, phone and FAX numbers, and an e-mail address. There is also an opportunity for the instructor to specify Web URLs for an individual home page, or an image, enabling the student to know and recognize the instructor. Provision is made for only a single instructor of the course, and there is no opportunity to specify similar items about the TAs within this general information area.

The set of Virtual-U features that facilitate planning are called "Course Structure Tools." They allow the instructor to modify topics, materials, and activities in the on-line environment. The Unit Definition tool enables the instructor to define and modify the individual units within the course. The system allocates space for the number of units specified in the general course information, above. The instructor specifies the unit topic and start date, provides a brief summary, and a more lengthy description of the goals and issues within each unit. All units are listed sequentially. In the overview (Figure 3), the unit title is visible, and details are accessed with a click. The sequence can only be altered by modifying the existing descriptions and topics; there is no automatic re-ordering.

Figure 3: Virtual-U Course Structure Tools


Netscape: Course Structure Tools
File Edit View Go Bookmarks Options Directory Window Help
Location:
Select
Add

```
http://virtual-u.cs.sfu.ca/SFU/cgi-bin/CS/CS_cst.cgi?course=C
```

Course Structure Tools
Course: CPSC319

Course: CPSC319

Modify:
- Course Sequence
- General Information
- Instructor Information
- Units/Topics/
- Regular Events/
- Materials, Resources and Conferences
- Assignments, Tests and Activities

Related Tools:
- Course Viewer
- Grade Book
- VGroups

Edit Course Sequence
CPSC319

Course Name: Software Engineering Project — Regular Events
Instructor Information:

Unit	Topics	Materials, Resources and Conferences	Assignments, Tests and Activities
All Units		Add Add • Courses:CPSC319	–
1 Apr 30	Introductions	Add Add *None yet*	Add • Straw Tower
		Add	Add

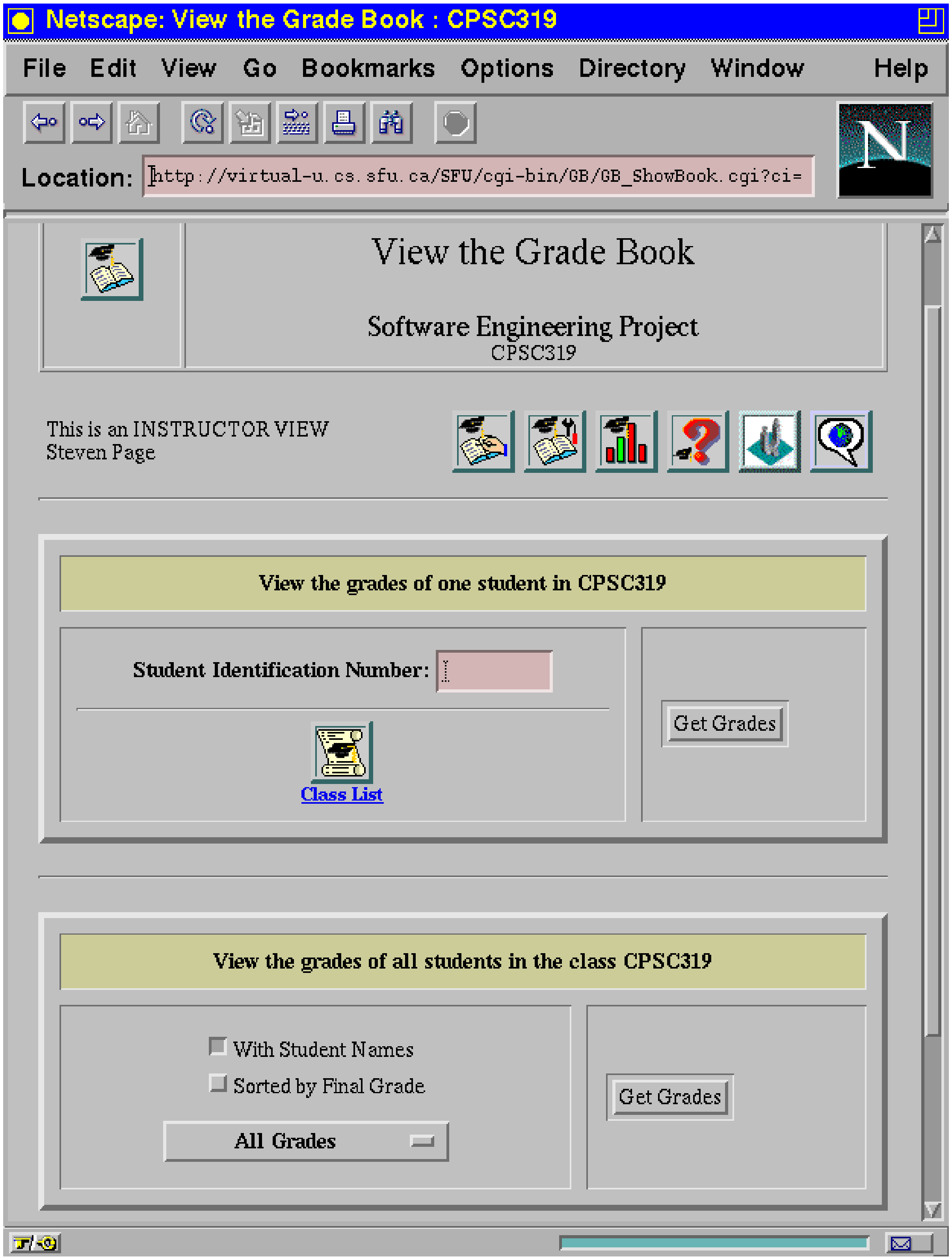
Another Course Structure tool, Activities Definition, provides the instructor with the opportunity to define, modify, and delete activities, assignments and tests for the course. Each activity is associated with a specific unit of the course. The instructor discloses the name of the activity, and provides a brief description. The mark can be entered, representing the value of this assignment, test, or activity within all activities in the course as a whole. In prompting for the mark, the maximum allowable value for the assignment is listed, based upon the percentage of the course grade already allotted to other assignments or activities. If the activity requires physical presence, such as an exam, the instructor specifies the location and date at which it will be held. Conversely, if the activity is on-line, the instructor is able to identify the Web URL. The start and due dates can also be defined, for each activity.

The Virtual-U provides another Course Structure tool, Material Definition, that enables the instructor to add, modify, or delete links to resources or materials. As with the activities described above, each resource is associated with a specific unit of the course, or can be specified as relevant to all units within the course. The material’s title, author, and description can also be specified. If the material or resource is available electronically, the instructor lists the Web URL. Finally, an assessment of the importance of the material to the course can be offered, declaring the material to be required reading, or merely a recommended secondary source. VGroups conferences, discussed later, can also be created, either for the course as a whole, or during specific units. With this arrangement, it is possible for the course to have a separate set of conferences for each of the topics to be discussed. The tool requests that the instructor specifies the “conference ID”, which is found only in the “Conference Information” area in VGroups. This identifier is an integer value of at least five digits, which must be retrieved from another location of the system, making the act of identifying it to the Course Structure tool burdensome. Fortunately, it is only needed in this single instance, for each defined conference being added in this way. Neither the students nor the course staff needs this information in order to join the conferences during the course.

The final Course Structure tool in the Virtual-U allows for the specification of the details of regularly occurring events. Unlike the others, this tool draws a distinction in its appearance, depending upon whether the course is being offered in mixed mode, or as an on-line-only course. For courses that occur strictly in an on-line environment, the instructor is only presented with the office hours field. The mixed mode courses, however, are given several fields for the instructor to enter information concerning regular occurrences. Items are not considered part of the course until they are explicitly chosen by the instructor. The time and day of the week, and the physical location, as applicable, can be listed. The tool accepts up to three lecture days, and two meeting and lab days. The instructor may also specify up to three tutorial days, three days of personal office hours, and five days for the TAs.

The course management facilities allow the instructor to see the course from the perspective of the students, through the Virtual-U Course Viewer tool. Traditional management activities, such as ensuring that all enrolled students have been issued user identification names and passwords within the system, are handled by the system administrator instead of the individual course instructors. Instead, the course management features present the instructor with a course sequence and overview, which are described in greater detail in the student section below. Currently, there is no way for the instructor to obtain reports based on student and resource usage, beyond the class activity reports discussed in the student assessment facility, below.

Figure 4: The Virtual-U GradeBook



Netscape: View the Grade Book : CPSC319
File Edit View Go Bookmarks Options Directory Window Help
Location:

http://virtual-u.cs.sfu.ca/SFU/cgi-bin/GB/GB_ShowBook.cgi?ci=

View the Grade Book

Software Engineering Project
CPSC319

This is an INSTRUCTOR VIEW
Steven Page

View the grades of one student in CPSC319

Student Identification Number:

Class List

Get Grades

View the grades of all students in the class CPSC319

☐ With Student Names
☐ Sorted by Final Grade

All Grades

Get Grades

The Virtual-U provides instructors with an on-line student assessment facility, entitled GradeBook. This tool, optionally in English or French, provides the course staff with the opportunity to record and review grades assigned to students for each of the course activities, and to manipulate the grades in several different ways. A limited subset of the contents of GradeBook is available to the students enrolled in the course, as discussed in the Student section, below. The instructor, however, maintains control over precisely how much grade-related information the students can access.

The instructor navigates to the GradeBook through hyperlinks available in the Courses building from the main Campus view. The main screen of the GradeBook (Figure 4) begins, by default, in the View Grades mode. This mode enables the instructor to examine the marks given to people in the class, either one student at a time, or all students at once. The grades are numeric values, of what each student scored on a given activity. Details for a single student can be accessed by entering the student number, or by calling up a list of all students (names and student numbers) enrolled in the course, and selecting the desired one from the list. The resulting display shows the grades for this student, for all required activities, tests, and assignments in the course, with the overall percentage earned to date. Optionally, the instructor can list the grades for the entire class, which gives a tabular display showing the grades for all students, in all required course activities. An overall percentage to date is also displayed. Additional features enable the instructor to create a few variations of the simple reports, by including the student names in the table of grades, or by sorting the students by class standing. The instructor may also display the grades for all students for a single assignment, the set of final grades, or all grades in the entire course. Simple statistics are calculated, including lowest and highest grades for each activity, and average scores.

The View Grades mode described above is one of four modes available in the GradeBook. A second method of viewing class results is the View Distribution Barcharts mode. Rather than individual numeric grades, this mode delivers a graphical representation of the scores for the class as a whole, providing a rapid overview of the distribution of the grades by activity. For each course activity assigned a grade, or for the final course grades, this mode displays the activity title, assigned start and due dates, and weighting of this activity within the course. This information comes from the assignment or activity descriptions specified in the Course Structure tool, as described above. Basic statistics are also presented for each activity: minimum and maximum scores, and the class average. The barcharts themselves are standard histograms, where height varies according to number of students with activity scores that fall within the given range of grades. The instructor is given five pre-set configurations of grade intervals for the display, choosing between two, four, five, ten, or twenty intervals. Since the grades in the histograms are presented as percentages, and the intervals within each option encompass equal ranges of grades, the two-interval option would have the first bar range from 0-50 percent, and the second from 51 to 100 percent. At the finest level of detail, the rightmost bar in the twenty-interval option includes 96 to 100 percent.

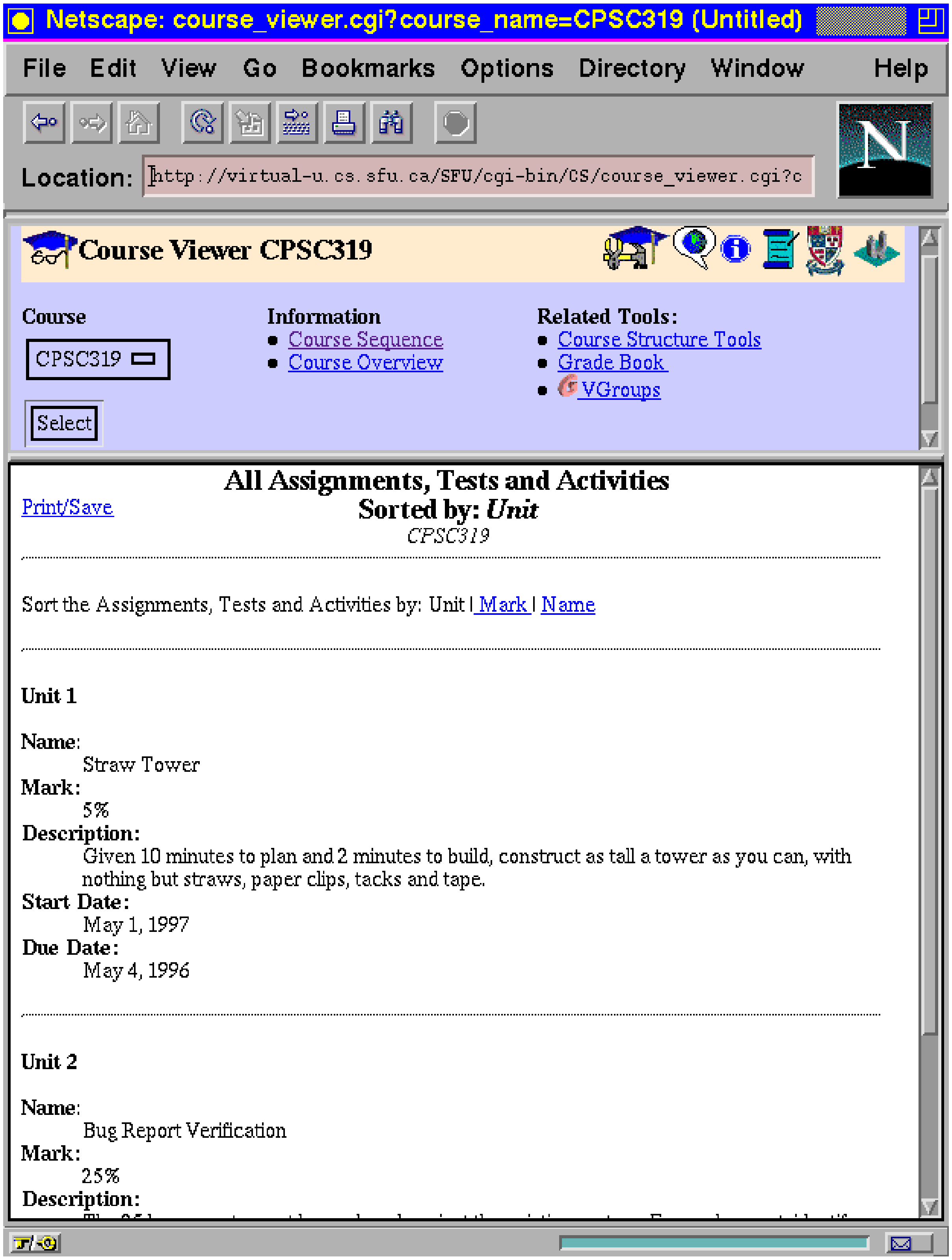
The two GradeBook modes discussed above serve to display the raw grades and their distribution across the entire class. This data must be entered, if it is to be manipulated by the system in these ways. Entry is accomplished in the Edit Student Grades mode. As with the View Grades mode above, Edit Grades enables the course instructor to enter or adjust student marks on either an individual or a class-wide basis. When editing the scores for an individual student, the instructor has the opportunity to record comments about the student, on two levels. Private comments have the date, and the name of the instructor or marker making the comments, directly and automatically attached to the comments, and are visible to markers and instructors only. This allows the course staff to alert and remind one another of any special circumstances for individual students. This is the only way of denoting deviations from the marking scheme within the GradeBook, and must be dealt with manually, on an individual basis. Summary comments about assignments and activities may also be submitted, which the student can see within the Virtual-U, receiving important feedback about their assignment and grade. A more rapid manual method of entering student grades is listing all students in the class in tabular form. The student's current mark for the assignment is listed, and can be updated. This allows several grades to be recorded or updated quickly, but does not provide an opportunity to present the students with feedback comments on an individual basis.

The final GradeBook mode relates less to the grades, and more to the environmental and administrative settings within the tool. The set of options includes minor features, such as defining what information students are able to see, rearranging the order in which the assignments are displayed, and other general settings. The instructor can indicate what access the students have to the GradeBook viewer, specifying whether or not they may access the class distribution bar charts, the final letter grade ranges, or the summary comments made about individual assignments by the instructor and markers. Of greater significance in this mode is the ability to alter the existing ranges for each letter grade, establishing new ranges, even eliminating some letter grades. The range for a given letter grade is defined as less than or equal to its associated value, and greater than the value of the next step down. For example, an A+ value of 100 and an A of 89 means that a value of 90 falls in the A+ range, while 89 is considered an A. It is notable that this is the reverse of the normal approach to letter grades, which are generally defined by their lower bound - for example, achieving 85 or more is an A grade. Instead, the GradeBook defines them by their upper bound.

For students, the Virtual-U provides an opportunity to attend virtual classes, to access on-line information resources, hypermedia and multimedia course materials, to work on individual and collaborative activities, and to participate in discussion groups. As seen in the set of instructor tools, courses can be presented entirely on-line, with text-based material and multimedia resources linked into the appropriate units. A broad range of library and Internet resources are also easily accessible by the students. The "Café" provides a space for social interaction on non-course-related topics.

The Course Viewer allows students to see the sequence of course units and events. A tabular layout, much like that in the Course Structure tools in Figure 3, lists the unit numbers and details, with the scheduled start date. This table includes Units, Topics, Activities, and Materials/Resources/Conferences columns. Finer grained details are available through hyperlinks in this table, and elsewhere in the Course Viewer. A hyperlink on the course title jumps to the Course Overview facility, which is described below. The Regular Events hyperlink displays the set of weekly activities defined by the instructor, such as lectures and office hours. Each column title in the table contains a hyperlink that summarizes all related administrative information. Overviews of the units, descriptions of the topics to be covered, and unit-by-unit summaries of the additional materials and activities are listed by following the corresponding hyperlink in the column titles.

Figure 5: The Virtual-U Course Viewer


Netscape: course_viewer.cgi?course_name=CPSC319 (Untitled)
File Edit View Go Bookmarks Options Directory Window Help
Location: http://virtual-u.cs.sfu.ca/SFU/cgi-bin/CS/course_viewer.cgi?c

Course Viewer CPSC319

Course

CPSC319

Select

Information
- Course Sequence
- Course Overview

Related Tools:
- Course Structure Tools
- Grade Book
- VGroups

All Assignments, Tests and Activities
Sorted by: *Unit*
CPSC319

Print/Save

Sort the Assignments, Tests and Activities by: Unit | Mark | Name

Unit 1

Name:
Straw Tower
Mark:
5%
Description:
Given 10 minutes to plan and 2 minutes to build, construct as tall a tower as you can, with nothing but straws, paper clips, tacks and tape.
Start Date:
May 1, 1997
Due Date:
May 4, 1996

Unit 2

Name:
Bug Report Verification
Mark:
25%
Description:

The detailed view of course activities is shown in Figure 5. Some of these detailed summaries can be presented in different orders. For example, all course activities are summarized by clicking on the Activities column title, and can be sorted by unit, activity name, or grading scheme value. There are also active hyperlinks within the rows and columns of the tabular display. Full details may be obtained for an individual unit, by following the hyperlink on the unit number. Similarly, detailed information about each topic, activity, or resource may be displayed by clicking on the appropriate item in the table.

The Course Viewer also presents students with a link to the Course Sequence tool. It allows students to access on-line content, in the form of Web pages or supplemental resources, through the appropriate hyperlinks in the Virtual-U. Such content is not restricted to solely text-based material. The hyperlinks could take the students to interactive tutorials, images, video clips, or full multimedia presentations. The Virtual-U does not provide tools to either instructors or students for the production and editing of such material. However, once such items are published in Web-based format, they can easily be incorporated into courses for delivery at the appropriate times.

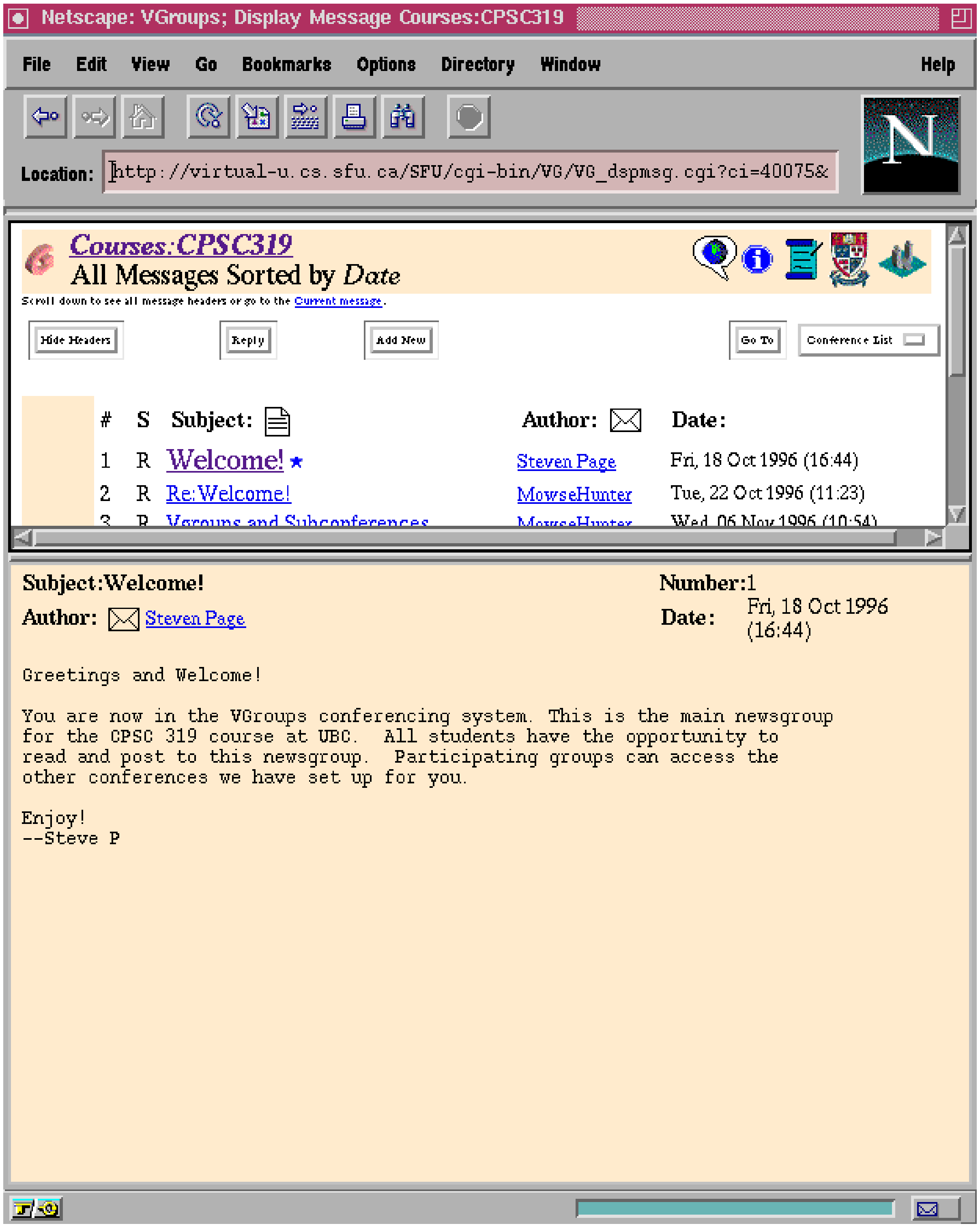
Through the Course Viewer tool of the Virtual-U, students can also access the Course Overview. This material is a compendium of the details provided by the instructor in the General Information area of the Course Structure tools. The course name, identifier, delivery mode, and duration are displayed, plus the names of the instructor and department. All the defined regular events are listed. The version current at time of writing (Beta 1.02) has a bug at this point; the days of the week on which the regular events are scheduled are one day later than they should be (a Tuesday becomes a Wednesday, for example). It is scheduled to be fixed in the next release. Much of the information in this tool is also accessible through the

Course Sequence and Course Viewer tools, since it also lists the course description and objectives, and a set of hyperlinks to information about the topics, materials, resources, conferences, assignments, tests, and activities for the course.

As mentioned previously, students enrolled in a course have limited access to the Student Assessment facility, the GradeBook, to which they navigate through the Courses building. Once at the GradeBook, students have access to their raw grades for all activities. Student access to more detailed grade information is at the discretion of the course instructor. When permitted, the students are able to examine the distribution of assignment scores across the entire class, as well as simple statistics like high and low scores, and class average. The instructor may also permit the students to have access to the final letter grade ranges for the course, and to see any individual comments the markers have for the students for each assignment and test.

One of the most significant features within the Virtual-U is the VGroups facility, a computer-based communication and conferencing system. Each conference constitutes an environment in which participants may hold discussions. The message, posted by a participant, is the centrepiece of the communication occurring in a VGroups conference. In that respect, VGroups is similar to a common newsgroup or bulletin board system. The conversation within the conferences is asynchronous. As with a newsgroup, the messages are threaded by subject, permitting the conversations to be followed as entities separate from one another. The VGroups interface (Figure 6) allows the readers to sort the messages by date, by subject, and by author.

Figure 6: In a VGroups Conference



http://virtual-u.cs.sfu.ca/SFU/cgi-bin/VG/VG_dspmsg.cgi?ci=40075&

Courses:CPSC319
All Messages Sorted by *Date*

Scroll down to see all message headers or go to the Current message.

#	S	Subject:	Author:	Date:
1	R	Welcome! ★	Steven Page	Fri, 18 Oct 1996 (16:44)
2	R	Re:Welcome!	MowseHunter	Tue, 22 Oct 1996 (11:23)
3	R	Vgroups and Subconferences	MowseHunter	Wed, 06 Nov 1996 (10:54)

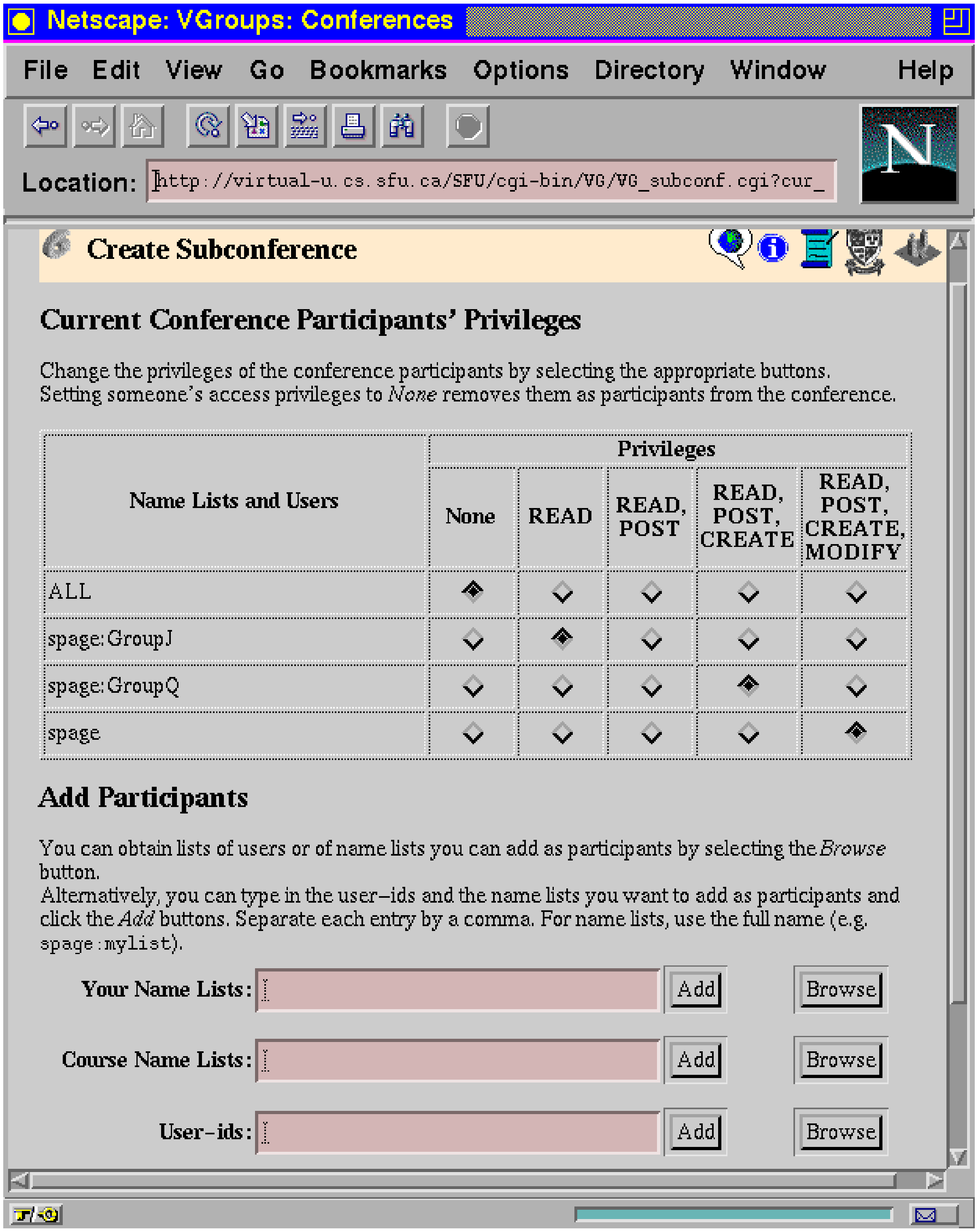
Subject:Welcome! **Number:1**
Author: Steven Page **Date:** Fri, 18 Oct 1996 (16:44)

```
Greetings and Welcome!

You are now in the VGroups conferencing system. This is the main newsgroup
for the CPSC 319 course at UBC.  All students have the opportunity to
read and post to this newsgroup.  Participating groups can access the
other conferences we have set up for you.

Enjoy!
--Steve P
```

Figure 7: Assigning VGroups Privileges


Netscape: VGroups: Conferences
File Edit View Go Bookmarks Options Directory Window Help
Location:

```
http://virtual-u.cs.sfu.ca/SFU/cgi-bin/VG/VG_subconf.cgi?cur_
```

Create Subconference

Current Conference Participants' Privileges

Change the privileges of the conference participants by selecting the appropriate buttons.
Setting someone's access privileges to *None* removes them as participants from the conference.

Name Lists and Users	Privileges				
	None	READ	READ, POST	READ, POST, CREATE	READ, POST, CREATE, MODIFY
ALL	◆	◇	◇	◇	◇
spage:GroupJ	◇	◆	◇	◇	◇
spage:GroupQ	◇	◇	◇	◆	◇
spage	◇	◇	◇	◇	◆

Add Participants

You can obtain lists of users or of name lists you can add as participants by selecting the *Browse* button.
Alternatively, you can type in the user–ids and the name lists you want to add as participants and click the *Add* buttons. Separate each entry by a comma. For name lists, use the full name (e.g. `spage:mylist`).

Your Name Lists: Add Browse

Course Name Lists: Add Browse

User-ids: Add Browse

Messages are posted to the conferences, and participants can read and respond to other posts as well. The conferencing system surpasses a standard bulletin board system, however, in three areas: it enables the creation of sub-conferences, it provides privacy through access restrictions, and it emulates aliasing, allowing people to specify a large number of participants through a single named list.

Sub-conferences can be created as areas to discuss topics that are related to - yet significantly removed from - the topic of the parent conference. A sub-conference can be created for each project team within a class, and sub-committees within the project teams can then create child conferences, another level down. Alternately, there can be a single root conference for the course, and each key weekly reading can be discussed in a separate sub-conference. This ability to create new conferences takes the sorting by threads one step farther - the conference now becomes a filter against unrelated topics, or messages that would be more appropriate elsewhere. This filter, it must be noted, is at the cognitive level - the system does not filter out inappropriate material, but rather the conference itself provides the clues about what discussions are appropriate.

Participation in any conference is controlled through access privilege levels. This filtering of the participants is at the discussion group level, but there is also an individual level, discussed later. The character of the conference is shaped by whether sub-conferences can be created with the current one as the parent, and by whether or not cross-posted messages will be accepted. Within the framework of each conference, each participant is assigned one of four distinct levels of participation. The most basic level is that of "Observer," a conference participant that is permitted only to read messages that are posted to the conference by others. (These categorizations are my own; they do not occur within the

Virtual-U system.) Those at the “Participant” level are also able to read conference messages, and have permission to post messages themselves. The “Initiators” can read and post messages, but also have authority to create additional sub-conferences, defining their subjects and participants. The final category is “Moderator.” This participant is able to read and post messages, and create new sub-conferences, but is also able to modify the parameters of the current conference. Such modifications may be alterations to the conference description and stated purposes. The “Moderator” is also able to alter the privileges of the other participants, including or excluding people as desired and as appropriate. A fifth possibility is to not be accorded any privileges for a conference. The access levels can be seen in Figure 7, which shows the privileges being assigned for a new sub-conference. These access levels combine to create privacy in the conferences. Students involved with one project team in the class can thereby be restricted from participating in the conference of another team, by being denied any privileges. If the teams do not have important secrets, they can permit others to join as Observers, able to read but unable to distract the team by posting unwanted messages.

The second level at which the conference participation is filtered is that of the individual. Despite all the roles a participant can take within a conference, membership cannot be forced. The “Moderator” may assign a student to a given conference, but that student must explicitly join the conference before being able to read any messages. Naturally, once a member of the conference, the student can participate to the fullest of their privileges, but the membership must be accepted. In a course context, it would be useful for the instructor to be able to automatically join the students to the new conferences. The instructor could then be assured that all students were aware of the new conference, and students could then explicitly “un-join” if desired, removing it from their personal list of conferences;

currently, all participants must explicitly join them. Even after joining a conference, however, the participant has the option of “Unjoining” it. Those who do not wish to interact with the others in the conference are not forced to.

The ability to create sub-conferences places these discussion areas in a hierarchical structure. There are two distinct super-hierarchies within VGroups, which encompass all others. The Public hierarchy consists of discussions, on general interest topics that are open to any participant in the Virtual-U. Topics may include a feedback forum, a source of assistance with aspects of the Virtual-U, heated discussion of the latest hockey results, and more. These areas are independent of any academic studies, and their management then falls to the system administrator, as opposed to a course instructor. The second super-hierarchy is the Course grouping, within which every course in the Virtual-U has a home conference, moderated by the class instructor. The full structure of nested sub-conferences resides within the base conference of each course, and the discussions are generally limited to topics relevant to the material being covered. The VGroups interface then allows the participant to navigate among various conferences by rising to a parent conference or descending into the tree structure a level deeper, into sub-conferences.

The creation of a new sub-conference requires the creator to identify a title, a short description, a more detailed summary of the purpose of the conference, and the access privileges. The temporal bottleneck in creating a new conference is in identifying the participants, and specifying their level of permissions. To facilitate this process, the Virtual-U has implemented the concept of User Name Lists. This permits a course to identify all enrolled students within a single name list. Students can also specify Name Lists for members of their project teams, and others for friends with similar interests. The Name Lists are collections of Virtual-U user identifiers, and can also contain other pre-existing User Name Lists. When the sub-conferences are defined, creators can proceed more quickly by assigning privileges to an existing Name List. The result is that all people on the list are then accepted into the sub-conference, with the defined access levels, as though they were individually created.

Like the rest of the Virtual-U, VGroups text is Web-based, in HTML format. This gives to the composers of the messages the power to embed anything that can be activated through a hyperlink. Messages can contain images, video clips, sounds, additional files or text, and more. This feature thereby takes full advantage of the power of the Web-based environment.

The Virtual-U encourages extra discussion and participation with the Café building. Within the Café, students are encouraged to share ideas and opinions, and explore new forms and methods of self-expression. This is accomplished through asynchronous discussions - the Café contains hyperlinks to the set of newsgroups at the institution, and to the Café conference within the VGroups Public hierarchy. Noticeably absent is any real-time interaction, as with a Talk or Chat program. One additional feature to the Virtual-U Café, that is still in development, is a Virtual Gallery for artistic expression in an electronic format.

The Virtual University provides an integrated software platform for planning, organizing, and delivering courses on the World Wide Web. Instructors are able to structure, plan, and manage their courses, students can access course materials, academic discussion groups, and social activities. The VGroups conferencing system provides asynchronous communication, with a few features above and beyond a standard newsgroup reader.

3.3 WebCT

The World Wide Web Course Tools (WebCT) system is a software platform for the creation and delivery of educational materials and activities in a standard World Wide Web environment. It is a system that facilitates individual and collaborative learning activities in two modes of on-line course delivery: on-line-only mode, with all materials and interactions through an electronic intermediary, and mixed mode, with traditional classroom teaching supplemented through on-line content, discussion, or activities. WebCT is being developed at the University of British Columbia and is currently used in courses at UBC, and around the world in forty different countries. The system consists of a customizable set of tools, features that provide course instructors and designers with a similar interface between courses, and is completely accessible through standard Web browsers.

The server component of the WebCT system must be installed on a UNIX machine, although the development of a Windows NT server is expected to reach initial testing in the second quarter of 1997. Only the system administrator, one of four types of WebCT user, deals with the server side of the system. As a Web-based learning environment, any networked machine with a Web browser can access the system. It is recommended that a browser with Java support be employed, thereby enabling users to access the full tool set.

The focal points for the users of the WebCT system are the individual courses. A course is an independent entity, relying neither on a larger area (such as a campus or a department site) nor on other courses; all related tools, features and content for the course are available through the course home page, and no aspect of the system is available for general use outside the courses. The administrator is the one who must define or delete courses from the server. Administrator involvement ends at this point. The other three types of user in

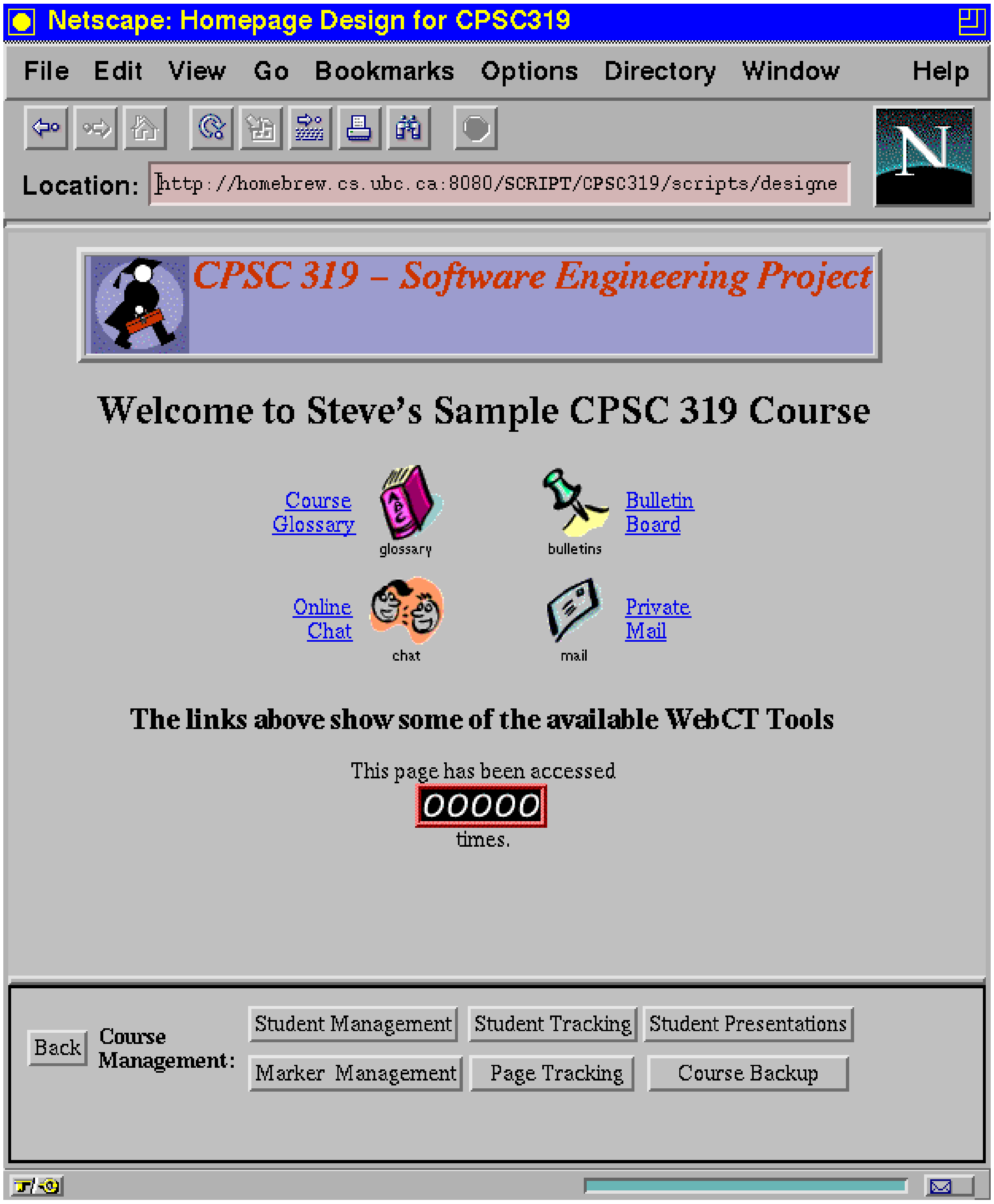
WebCT are designers, markers, and students. Access to tools or materials within a course is restricted through a user identification code, and corresponding password. Students that are enrolled in multiple WebCT courses will potentially have multiple user names and passwords; likewise for an instructor who is involved in multiple courses. The communication tools, discussed later, are primarily geared for conversations and discussions occurring within, not between, courses. Some communication, however, is possible between students in different courses, provided the courses in question are on the same WebCT server. This is true for the Chat and electronic mail tools.

As alluded to above, there are four categories of user in the WebCT system. The administrator account is unique at any given installation. This super-user class accesses the administration page of the system, from which courses are created and deleted, allocating disk space as appropriate. Control of the course settings and content creation then passes to the second category of user: the course designer. Each course is given a single designer account, the owner of which can add educational content to the course, monitor student progress, manipulate and calculate grades, and participate in the discussions. One power of the designer is to create as many marker accounts as needed, each able to grade quizzes and affect the grades in the student assessment facility (although they cannot affect course content materials). This is the third class of user. The course designer also has the power to create and revoke accounts for the final class of user, the enrolled students. These accounts can access, but not alter, the course content material, except where explicitly permitted (for example in a discussion or a project team area). There is no “visitor” access to the main course area, although a course Welcome page can be created, which does not require a system account to access. Through this feature, information can be given to prospective students and other interested parties, although they cannot preview current topics and assignments.

The WebCT system gives the course designer (or instructor) a set of Course Planning tools and features. The first of these is control over the “look and feel” of the course. The system places limits on the appearance of the interface by its very nature as a Web-based application, by which tools are available for inclusion within the course, and by the assortment of available options for system configuration. This provides courses with a certain similarity of interface. Within these imposed boundaries, however, the course designer has control over which educational and assessment tools will be available for use in the course. For example, at their discretion, instructors may deny students the use of the chat facility, the progress tracking statistics, or any other tools. A default set of tools can be made available through a toolbox found on every page in the course. This allows some subset of the system features to be readily available as needed. The designer retains the option of superseding the default tool set on any particular page of content, including the course home page (see Figure 8), content pages, and individual tool pages.

The designer is also able to define a default colour scheme for all course pages, through the Presentation tool. This default scheme can then be overridden for individual pages, to provide greater impact and effect. Default sets of graphic images and tool icons are available, some conveying a formal appearance, others a casual appearance. Optionally, the designer may upload new images for icons and banners. Except for style and appearance attributes, the interface remains generally unchanged.

Figure 8: The WebCT Course Home Page


Netscape: Homepage Design for CPSC319

File Edit View Go Bookmarks Options Directory Window Help

Location: http://homebrew.cs.ubc.ca:8080/SCRIPT/CPSC319/scripts/designe

CPSC 319 – Software Engineering Project

Welcome to Steve's Sample CPSC 319 Course

Course Glossary — glossary

Bulletin Board — bulletins

Online Chat — chat

Private Mail — mail

The links above show some of the available WebCT Tools

This page has been accessed 00000 times.

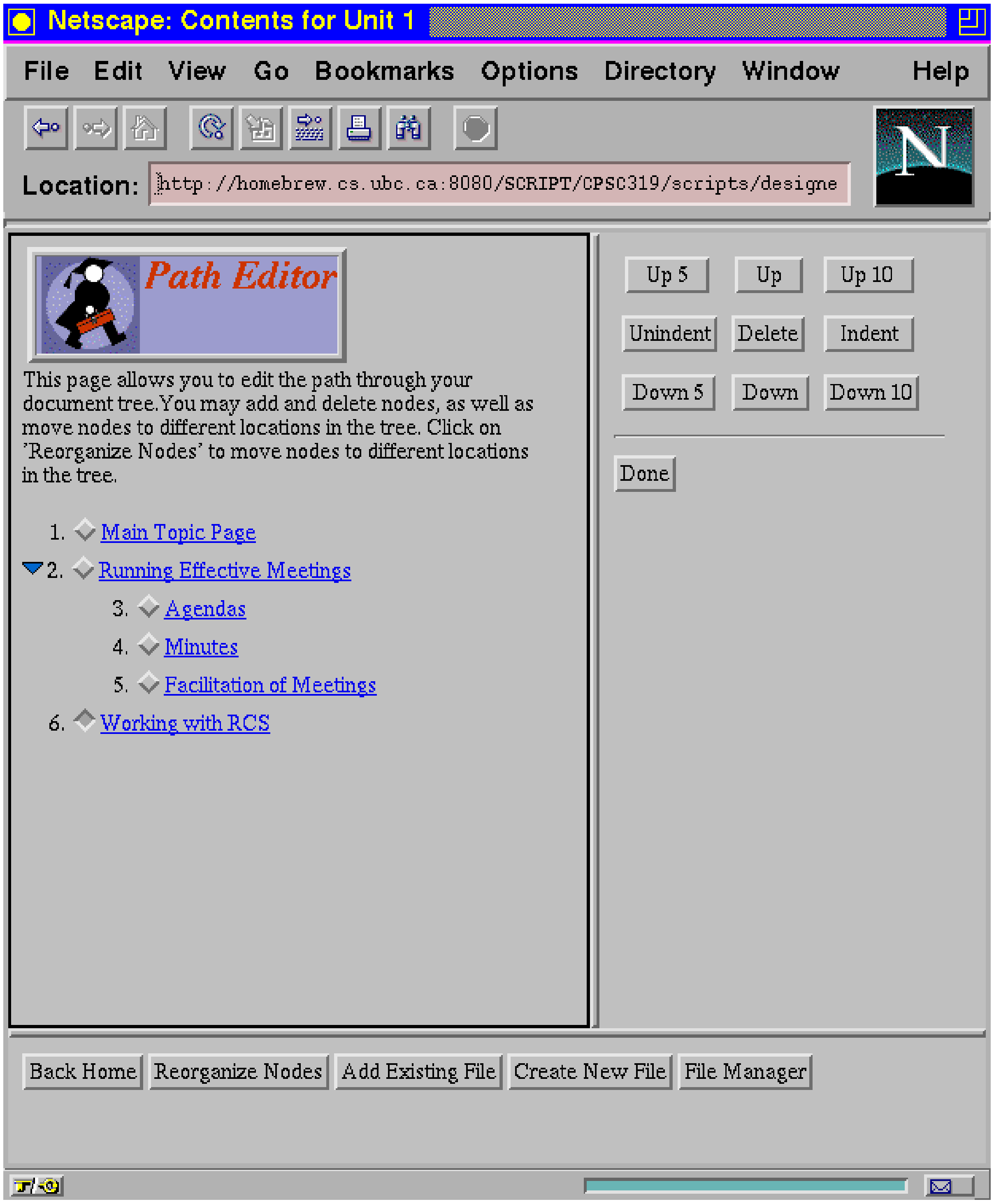
Back

Course Management: Student Management | Student Tracking | Student Presentations | Marker Management | Page Tracking | Course Backup

The designer also has the ability to establish the terminology and generate a searchable glossary of terms for the course material. WebCT automatically creates the links from any occurrences in the educational content to the appropriate definitions, under the control of the designer; when creating the glossary, the system finds all occurrences of the words in the content and allows the designer to select whether or not each one will link to the glossary. This automatic linking highlights the terms within the body of the text, as glossary links, while the system handles all administrative issues behind the glossary, leaving the original source untouched. A student can then obtain the definition simply by clicking on the term wherever it occurs in the text. Alternatively, students can search the entire glossary. WebCT also enables an index of topics to be quickly and easily created. The alphabetical index contains links to the pages of content within the course that deal with a particular topic. The index items can also be annotated by the designer, to provide a more precise indication of the aspects of the topic that are covered in each page.

No facility is provided by the WebCT system for the creation of course content, such as an HTML editor. Other editors exist, which enable hyperlinked documents, images, and Web pages to be created, and which require no knowledge of HTML. Once the content exists, it can easily be linked into the course. WebCT assists this process by providing an uploading facility, by associating sophisticated links and features with the content, and by enabling the material to be structured linearly and hierarchically. The Path Editor tool, illustrated in Figure 9, enables the designer to structure the sequence or hierarchy of the content pages. The course “path” is structured as a tree of topics, whose nodes can be expanded, collapsed, inserted, deleted, and rearranged.

Figure 9: The WebCT Path Editor



This page allows you to edit the path through your document tree.You may add and delete nodes, as well as move nodes to different locations in the tree. Click on 'Reorganize Nodes' to move nodes to different locations in the tree.

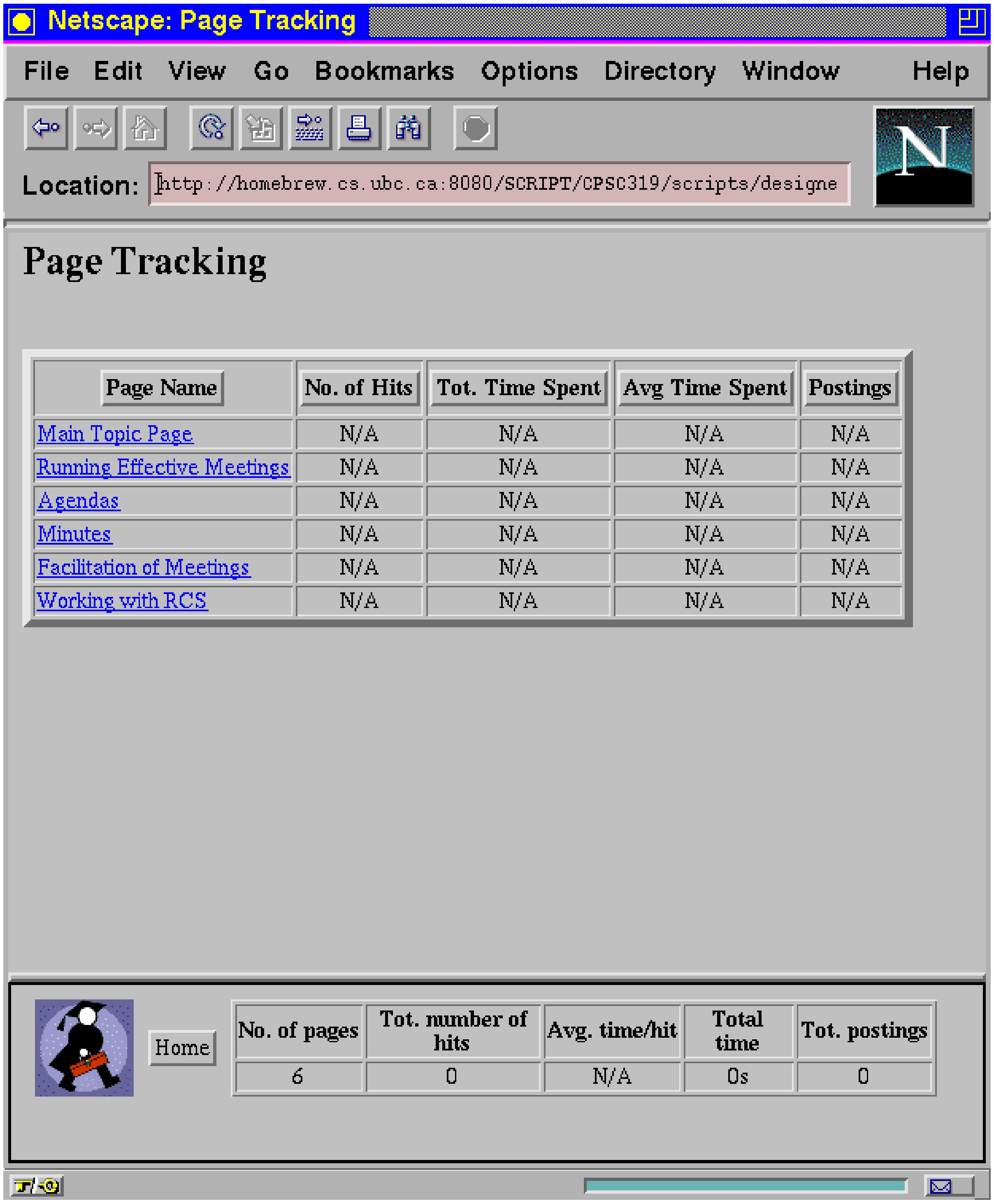
1. Main Topic Page
2. Running Effective Meetings
   3. Agendas
   4. Minutes
   5. Facilitation of Meetings
6. Working with RCS

WebCT includes a file management tool, through which the designer can upload, download, and manipulate course files. Standard file and directory functions are available, such as copy, rename, delete, and create. HTML files can be edited within the file manager, and compression and decompression algorithms make it possible to conserve disk space or archive groups of files. A final course planning facility the system provides is the backup utility. With this, the designer can download the entire course to another storage medium for safe keeping, or port the course to another WebCT server, possibly at another institution, to offer the same course content elsewhere.

The instructor is also provided with a suite of Course Management tools within WebCT. Two minor management features include the central launching point, in the form of the individual course home page, and the WebCT clipboard. The former allows the instructor to establish an area from which students can access the educational tools or pages of course content. It is an ideal location to display a course banner graphic, important links, and daily messages. The WebCT clipboard provides greater integration among the tools; information in the form of query results or existing text can be copied to the clipboard and inserted into another tool, such as a message to be posted to a bulletin board forum.

A significant WebCT feature provides instructors with the ability to track student progress, Web page popularity, usage, and more. This powerful facility enables the instructor to monitor students on an individual basis, within the context of the course, to determine the quantity of the course material they have accessed. It has been emphasized in the literature as a key feature, allowing the instructor to observe the progress of the class, restoring an important source of information that is lost when the class is held on-line [Goldberg 1996].

Figure 10: The WebCT Tracking Tool



http://homebrew.cs.ubc.ca:8080/SCRIPT/CPSC319/scripts/designe

Page Tracking

Page Name	No. of Hits	Tot. Time Spent	Avg Time Spent	Postings
Main Topic Page	N/A	N/A	N/A	N/A
Running Effective Meetings	N/A	N/A	N/A	N/A
Agendas	N/A	N/A	N/A	N/A
Minutes	N/A	N/A	N/A	N/A
Facilitation of Meetings	N/A	N/A	N/A	N/A
Working with RCS	N/A	N/A	N/A	N/A

No. of pages	Tot. number of hits	Avg. time/hit	Total time	Tot. postings
6	0	N/A	0s	0

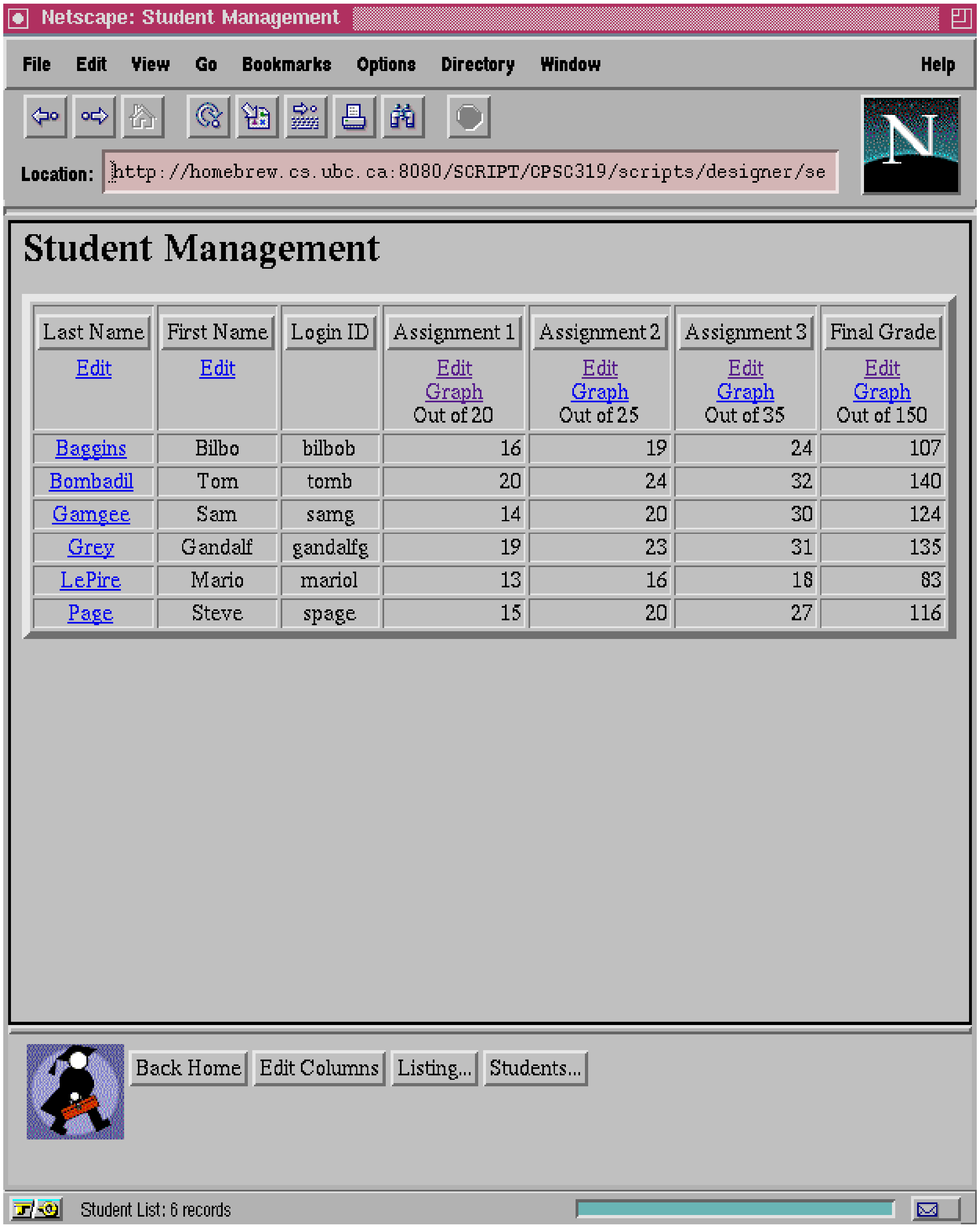
There are several specific measures of student progress that are captured by WebCT for the benefit of the instructor, the results of which are presented in tabular form for each student and each measure. Figure 10 shows the table of basic information collected about each student, although more details are available on an individual basis. The date of the initial course access for each student is provided, as is the date and time of each student's most recent access to the system. This enables the instructor to ascertain which students have not signed on recently, or at all. The total number of access of each student is also provided, a measure that couples with the times of accesses to provide evidence of the level of participation of the student. Some courses may require participation in the on-line discussions as one component, to encourage greater participation and facilitate collaborative learning. Such courses benefit from the ability of WebCT to track this participation by displaying the number of postings to the discussions that the student has made and has read. These numbers serve to indicate to the instructor the involvement of the student in the discussions. They are listed for all students in the course, and the instructor is able to easily sort the class on the basis of any measure. It is simple, then, to determine which students are the "quietest" in the on-line discussions, and which have fallen behind in their readings.

More details about the individual students are also available. By clicking on the student's name, the instructor calls up the full student profile. In addition to summarizing the information mentioned above, the student profile allows the instructor to track the distribution of the accesses the student has made to various areas of the course, such as pages of notes, glossary items or learning goals. Instructors can also track the percentage of content pages that the student has accessed, and list a sequential history of when the student accessed each of the pages. The power of the tracking tool extends beyond reporting merely on the students, however; It also provides the instructor with insight into the use to which the students are putting the course content pages. By reporting the total number of accesses, on a class-wide basis, to each Web page, coupled with a calculation of the total time and mean duration of each of these accesses, the tool enables the instructor to speculate about which topics are the most interesting, and which are the most difficult. The instructor can also quickly see which pages of notes are sparking the most postings to the conference system.

The student management component of course management includes primarily the student progress tracking capability described above. It also allows the instructor, the final authority for course access accounts, to create or revoke as necessary the accounts for individual students, and define project teams. WebCT provides the ability to accomplish this task either manually, entering the information one student at a time, or by preparing a simple script of the class list to automate the process and reduce the time required.

The course designer is the only one who can access the course planning and course management tools described above. The student assessment features, however, are accessible to more people. Should the instructor create one or more markers for the course, they would share with the instructor the ability to access the student assessment facilities of WebCT, the Grades Database. As discussed below, students have a limited view of the contents of the assessment facility, focusing on their own achievements. Markers are permitted to grade quizzes on-line, and to manually enter other grades within the assessment tools. Figure 11 illustrates the Student Management screen, with the student grades. The instructor can edit the grades, or view a histogram of the results.

Figure 11: The WebCT Grades Database



http://homebrew.cs.ubc.ca:8080/SCRIPT/CPSC319/scripts/designer/se

Student Management

Last Name Edit	First Name Edit	Login ID	Assignment 1 Edit Graph Out of 20	Assignment 2 Edit Graph Out of 25	Assignment 3 Edit Graph Out of 35	Final Grade Edit Graph Out of 150
Baggins	Bilbo	bilbob	16	19	24	107
Bombadil	Tom	tomb	20	24	32	140
Gamgee	Sam	samg	14	20	30	124
Grey	Gandalf	gandalfg	19	23	31	135
LePire	Mario	mariol	13	16	18	83
Page	Steve	spage	15	20	27	116

Student List: 6 records

WebCT provides an automated facility for creating and grading timed on-line quizzes. A quiz can be generated from a database of potential questions. True/false, multiple choice, matching and fill-in-the-blank questions are graded automatically by WebCT, but short-answer questions require marker intervention. The instructor can hand-select questions from the database, or have the tool randomly and uniquely generate each quiz. A history of the performance of each question is kept, and its results can be tracked to compare with previous years. The quizzes are set to become accessible at a predetermined date and time, ensuring all students write it within a specific time period. On beginning a timed quiz, the students are shown a clock that counts down the time remaining for them to complete it. A status page informs the marker about who has completed the quiz, who has begun but not yet finished, and who has yet to begin. Upon completion, the results of the quiz remain on-line, where they may be graded by one of the markers. The completed quiz also includes an indication of the amount of time the student took to write it. If the tool is used fully, and the quizzes are graded within WebCT, the scores are added automatically to the Grades database, and are made available to the individual students, along with any comments from the marker.

The on-line quiz tool is but one component of the student assessment facility of WebCT. Grades for quizzes, tests, and other assignments that are completed outside the WebCT system - perhaps oral exams, or assignments done on paper - can be entered manually into the system, one at a time, in a spreadsheet-like form. The instructor has control over which components of the Grades database are visible to the students, and the relative worth of each item in the collection of assignments and tests; when the weights are specified for each item, WebCT is able to compute the grade to date, and the final score, of each student from the raw scores. The Grades database also provides the course staff with a query tool, enabling them to obtain statistical information about the grades. Histograms can be generated for the distribution of the grades for each assignment, and the final scores. The query tool also enables the instructor to determine results by specific students and by specific grade ranges. It is then simple to determine which students are failing, and contact them for extra help, for example. Comparative histograms can also be generated for queried subsets of the students.

For students, WebCT provides the opportunity to access text and multimedia course materials, attend on-line classes, complete activities individually or collaboratively, discuss course-related issues, and receive feedback on their progress. The suite of tools available for enrolled students includes content access, support and enhancement facilities, communication tools, and feedback opportunities.

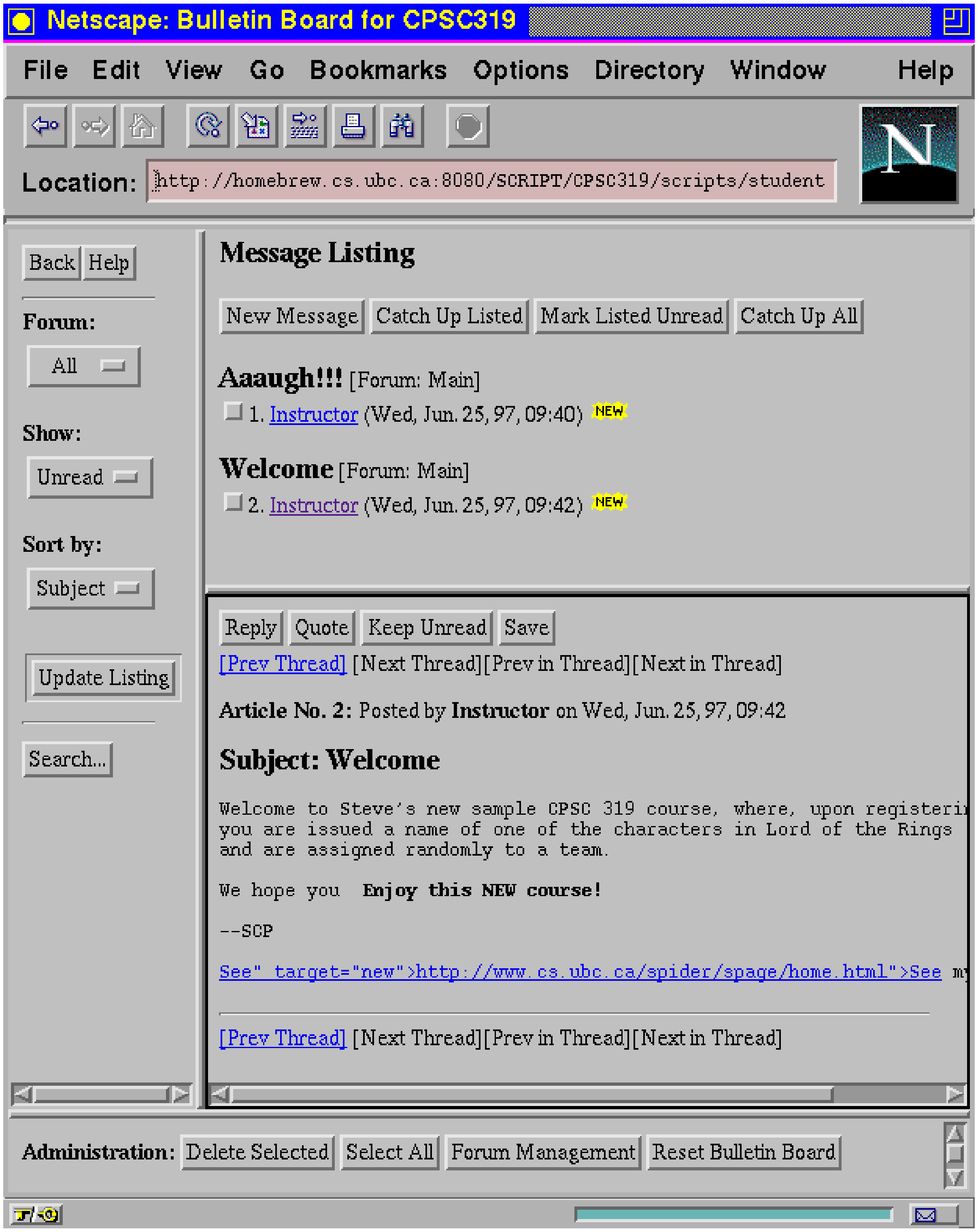
WebCT provides the necessary support for a course to be delivered completely via networked computers, or to include components of both the traditional classroom and the networked environments. These two modes of delivery may require different quantities and styles of content in the on-line material, but the content delivery features of WebCT are sufficiently rich that either style can be reasonably accommodated. Course content, in the form of text, images, or multimedia, is structured through the Path system described above. This feature enables the instructor to align the educational materials in a hierarchical format, by sections and sub-sections. Students can then navigate the course content logically, by topic and section. The Path, when viewed alone, without jumping to any of the pages of content, provides an outline of the major topics in the class, a syllabus that aids in reviewing the material covered. Student navigation of this material is further assisted by the Session Resumption tool. The function of this tool is to place the student at their last accessed page of educational content. By keeping a log of the material accessed by the students, the system remembers where they were in their last reading of course material. By then jumping immediately to this location in the following session, WebCT enables the students to restore their last context quickly, maximizing their study time.

Student access to course content is enhanced through a series of support tools provided by WebCT, which allow the students to search the materials and gain additional information. The Glossary tool, described above, provides information, and reminders of terms and concepts that were covered previously. Students access glossary items by clicking on highlighted words, and the descriptions appear in a small pop-up window. An alphabetical index to key concepts can also be created by the designer. Students then have the ability to link from the index directly to the relevant pages, or perform a search within the index tool. The search examines the content pages for occurrences that match the search criteria. The list of links that is returned is prioritized based on the location of the match (in the title or the body of the pages). An image archive can also be created by the designer, supplementing the text in the content pages with visual representations. Text that is associated with each image can then be searched by the students, resulting in a set of scaled-down versions of the full images. The larger images are accessible with a mouse click. WebCT also provides a facility for tying in external references to the course content. These references are items that are external to the course content itself, and may include off-site Web-based resources, textbooks, academic papers, and more. The system does not, however, give instructors the power to prevent external links from being created by students, for example in the discussion forum.

WebCT provides three communication tools for use in the courses, the first of which is the Course Conferencing System. This tool facilitates communication among all participants in the course, including students, markers, and the instructor, by functioning as an asynchronous course newsgroup. Messages are broadcast to all other course participants, creating a class-wide discussion group. The WebCT interface allows the messages to be sorted by the message subject, date, and author, and a simple search facility permits participants to scan message headers and bodies for specific content. The system supports embedding HTML tags within the body of the messages, enabling messages to refer to a home page or other electronic reference materials. The conferencing tool is based on the concept of "forum." There is no limit to the number of fora that the designer may include in a WebCT course, each one a separate discussion of a distinct topic.

A separate forum exists for each page of course content, thereby enabling the students to post questions and comments about a specific set of material. Messages to a page-specific forum can be created by entering the Conferencing System from the content page in question. The system then automatically populates the subject field of the message to refer to the current content page. In a normal forum, the subject line must be filled manually; leaving it blank will cause an error condition, which presents the typist with a blank message form. The entire message must then be re-typed. The designer can also create a private forum for a group of students collaborating on an activity. Only the designer has authority to create a forum, or to remove posted messages. Figure 12 shows the Bulletin Board, the threads and messages, and the currently selected one.

Figure 12: The WebCT Bulletin Board



Message Listing

Aaaugh!!! [Forum: Main]

1. Instructor (Wed, Jun. 25, 97, 09:40) NEW

Welcome [Forum: Main]

2. Instructor (Wed, Jun. 25, 97, 09:42) NEW

[Prev Thread] [Next Thread][Prev in Thread][Next in Thread]

Article No. 2: Posted by **Instructor** on Wed, Jun. 25, 97, 09:42

Subject: Welcome

```
Welcome to Steve's new sample CPSC 319 course, where, upon registeri
you are issued a name of one of the characters in Lord of the Rings
and are assigned randomly to a team.

We hope you  Enjoy this NEW course!

--SCP

See" target="new">http://www.cs.ubc.ca/spider/spage/home.html">See m
```

[Prev Thread] [Next Thread][Prev in Thread][Next in Thread]

A second asynchronous communication feature provided with WebCT is an electronic mail facility. This tool permits one-to-one communication, restricted to participants within a given course; it is not a full mail server that communicates with other sites around the world. As a result, it is geared for individual communication among students, and between students and course staff, and not for general purpose use. Students can employ this tool to coordinate with others during collaborative activities, ask questions, and hear from the instructor on an individual basis. The same searching and sorting facilities exist for this tool as for the conferencing tool.

WebCT provides a Chat tool, for synchronous communication among staff and students enrolled in the course. The tool establishes six “chat rooms” for the students in the course, in which discussions can occur. Five of the six rooms are reserved for sole use of the people involved in the course, while the remaining room provides an opportunity for students involved in different courses on the same WebCT server to interact in real time. These interactions may be purely social, or may involve discussions and debates that are related to some aspect of the material being covered in the course. Of the five rooms that are designated solely for course participants, four are logged, with a record kept of the discussions held in them. The instructor is then able to examine this transcript, to determine what topics are being discussed, what questions are being raised, who is participating and what is being said. This time-sequential transcript is one artifact of the WebCT system that will be discussed in the Annotation chapter later. The tool enables the course instructor to assign discussion groups, and to hold virtual tutorials on-line.

There are several student learning enhancement and verification tools that are included within WebCT. The Learning Goals tool enables the instructor to associate specific learning objectives with each page of course content. These objectives are then available to the students, providing them with the ability to assess whether or not the objectives are being met in each individual case. Students can record personal thoughts and questions about each page of content as well, through the Page Annotation tool. As the students read the material, they can utilize this feature to note any questions that arise, or any insights. Such annotations provide the students with reminders for study or review; the notes remain private to the individual student, and exist only for the duration of the course. Each page of course material can also include a multiple-choice quiz that enables the students to assess whether or not they have grasped the main points of the material. When activated, the Student Self Evaluation tool presents the student with the set of questions. When the quiz is completed, the answers given are automatically marked, as either correct or incorrect. An explanatory paragraph may then be returned with each answer, clarifying why the response was correct or not. These self-assessment quizzes are provided for student understanding only; the grades are not recorded in the Grades database. A final learning enhancement feature of WebCT is the Compile tool, which automatically generates a study guide for the student. All content pages within the course are listed for the student, who can then select which pages to include in the study guide. The system collects all selected material into a single Web page, which may then be easily printed or reviewed.

Students are also provided with access to their grades to date, and any instructor comments. At the discretion of the course designer, students may also view more detailed information regarding the distribution of grades among all students, and basic statistics (high and low grades and means). Designers may also make available to the students the information gathered on them by the Progress Tracking tool. There are some ethical privacy issues around collecting this information, and by choosing to make what is collected on them available to the students, WebCT allows instructors to elect to not hide this information from the students. By making this information available, students become aware of exactly what the instructor knows about them and their usage and participation patterns. It is not possible, however, to give students the choice of whether or not to allow the collection of this information. Facilities are also provided within the system for each student to make a personal home page. A simple authoring tool enables them to add text, images and links to their home page, which is accessible only within the course in which it is created. Students may also be given space for presentations, through Web-based pages. These presentation areas are intended to allow students to place projects, papers, illustrations and other course activity artifacts within the course space. Privileges for this area can be given to single students, small project teams, or the class as a whole.

These few paragraphs do not fully describe WebCT. Refer to [Goldberg 1997, Goldberg 1997B, Goldberg 1997C, Goldberg 1996, Goldberg and Salari 1996] for more complete information. This section has summarized the major features of the system: course planning, course management and tracking, student assessment, communication, and student learning tools.

Chapter 4 -- Studies and Annotation

Previous chapters have discussed several software applications that deliver educational material to students over networked computers. The Virtual University and WebCT were examined in greater detail than the rest. This chapter surveys several research studies completed on computer-based education systems. The twin issues of annotation and linguistic analysis are then examined.

4.1 Studies

In October of 1995, the National Science Foundation (NSF) held a workshop that sought to establish a research agenda to guide developments in computer science into advancing the state of existing and emerging educational technology. In the report published as a result of the workshop, the motivation for this coupling of computer science and education was that, when properly realized, information technology has proven capable of increasing cognitive skills, encouraging student motivation, and increasing the amount and quality of learning. The problem, however, is that the technology has not yet reached its full potential, be it through advancement of the level of technology, or through inroads into the educational systems [Guzdial and Weingarten 1995]. The report lists several important areas of research, into evaluative measures, sociological factors, curriculum studies, and pedagogical models. Several specific areas of computer science, relating to educational technology, are targeted for future research by the NSF. The workshop identified Human-Computer Interaction as one area needing research, into designs in which students at various levels will learn, and come to understand the virtual spaces in which they are learning. Software systems that contain assessment methodologies and flexibility regarding platforms and levels of learners were identified as needs. Other issues in the areas of networking, databases, policy, and artificial intelligence were also highlighted, emphasizing access to knowledge coaches and just-in-time learning. This research is required to advance the state of educational technology, and overcome some of the existing problems. Identified shortcomings are based on the independence of many of the current tools, which are often unable to use existing materials, and which fail to integrate into existing sociological and curricular structures [Reisbeck 1995].

Partly in reply to the NSF report, several institutions have been developing educational systems and conducting studies of their effectiveness. As shown in preceding chapters, a primary interest in my thesis is the emphasis on collaborative learning within on-line systems. The studies summarized below have in common some aspect of computer-based cooperative work among the students.

The CaMILE system, discussed in greater detail in Chapter 3, has served as the medium through which several studies have been conducted recently. Two in particular have focused on learning through collaboration in areas of high complexity. The first applied CaMILE to an engineering course, where the complexity is caused by design issues, resulting in a large number of related concepts and domains of knowledge [Guzdial et al. 1995]. The course utilized the asynchronous broadcast facility of CaMILE to have the students discuss assigned topics, and to work towards a group design for an assigned problem. The researchers were disappointed by the results, which showed that students rarely used the features that directly enhance collaboration. Rather than using the system to discuss important issues related to their group goals, the students operated individually and posted reports of their current status to the message system. Edwards and Mynatt [1997] call this “autonomous collaboration." The above study concluded that the absence of full collaboration was a result of the classroom culture in which the study was performed - the goal in the identified culture is primarily to do the activity itself, and learning from the activity is secondary in emphasis.

The second study into the collaborative features of CaMILE examined the system in an interdisciplinary course setting, where students cooperated to identify key issues, integrate their understandings and analyses, and synthesize them into a proposed solution [Hmelo et al. 1995]. CaMILE was employed to encourage students to reflect on the process of solving the problem, and on the contribution toward the ultimate solution of each participant, through asynchronous dialogue. Group collaboration was examined through student logs, and group discussions. Learning was assessed through pre- and post-tests. The study found a lack of collaboration among the students, despite the team assignments and case studies. The students seemed unaware of any assistance that CaMILE could provide to alleviate their difficulties in cooperating on the assignments, and in meeting outside class time. The reason for this, it was suggested, is that CaMILE was an optional tool, rather than a required component. As a result, its use was minimal, and several attempts by the course staff to begin discussion topics failed. The exchanges between participants were most meaningful when the thread was created by students, perhaps in a more private forum, but even these situations held little evidence of collaboration.

The creators of WebCT have performed a number of studies, comparing the acceptance of the tool set among first-year and third-year students. This was not a repeated-measures experiment, but rather looked at two separate populations. The study [Goldberg 1997C] showed a greater acceptance of the on-line tools among the third-year students than among the first-year ones. This was particularly true for the conferencing tools. Subjective measures also revealed greater comfort levels in the on-line environment among the more advanced students.

Hiltz studied the effects on learning and collaboration within the Virtual Classroom system from the New Jersey Institute of Technology [Hiltz 1995]. The Virtual Classroom is discussed in greater detail in Chapter 3. Over a period of ten years, she and her collaborators examined its use in several different courses, from such diverse fields as Computer Science, English Composition, Sociology, and Statistics. Evidence from these courses indicated that there was generally no significant difference between the level of mastery of the course material by the students; mastery by students using the Virtual Classroom was equal to, and occasionally above that obtained in the same course taught only in a traditional setting. Based on convenience and participation, students reported a higher subjective level of personal satisfaction with the computer-based course compared to the traditional classroom. Hiltz also drew some interesting conclusions about class sizes in on-line environments. She asserts that ten is the lower bound for the lively interaction that makes collaborative learning effective, and that an upper bound of thirty students is imposed by the time commitment required of the instructor [Hiltz 1995]. The presence of teaching assistants may raise this upper bound, but this entails a significant commitment and high level of knowledge and experience on the part of the teaching assistants.

Hiltz raises an important issue, by suggesting an upper bound on the class size of thirty, for an effective on-line environment. To create a proper level of responsiveness on the part of the instructor, a daily routine must be established of checking the discussions and the communication facilities, replying to any unanswered questions, maintaining a presence in the debates, and monitoring the students for the expected levels of participation. This is the minimal commitment, providing instructor participation in the discussions, and basic tracking of student progress. An on-line environment requires more, not less, instructor commitment [Westrom and Pankratz 1997].

Some instructors go beyond this participation, into a meta-communication level. The instructor, in such a course, maintains a presence in the discussions and debates, encourages participation, answers questions, and tracks student progress, but may be more interested in the educational process than in the end product of the learning. Process is more easily analyzed at a meta-level, where the instructor can gain insight into what is happening behind the messages, than by looking at the messages themselves. The term "process" contains some ambiguities. It denotes advancement, progressing through gradual changes toward a goal or an end result. In an educational setting, the "learning process" connotes a sequence of events through which the student gains a greater understanding of the subject matter, moving to deeper comprehension and mastery. "Process" also refers to the progression through the stages of learning: anticipating beforehand, doing an activity or practicing a technique, and reflecting on the activity after completion [Hunter 1995].

In whatever context it is defined, the process of achieving goals can be as important and even more rewarding than the final products themselves. It has been said that people "often enjoy the process, and not just the product; they want to take an active part" [Fischer et al. 1991]. In non-academic settings, problems in assessing processes stem from a lack, not of documentation or artifacts, but rather of information regarding the context from which they emerged, and the processes by which they were created [Conklin 1993]. In an educational situation, an artifact-oriented perspective evaluates papers and exams to determine what the student has learned by the end of the course, with milestones along the way contributing to a fixed final grading scheme. Some courses, however, shift the paradigm of the educational setting to emphasize processes. These courses must take into account the interactions, conversations, debates, assumptions, and paths taken through the course by the individual students and teams. In her report, Hawkins [1995] raises excellent questions pertaining to software-based tools and how they might assist both teachers and students to assess the development of higher cognitive and thinking skills. How, indeed, can this information about process be captured, assessed, evaluated, monitored, and guided? Such an activity is more difficult than grading an exam question as simply correct or incorrect, or weighing the merits of an essay. I would suggest, however, that much process information is captured in the messages and discussions that occur in the on-line environment. The problem is not where to find the information, when a collaborative telelearning environment is being used, but rather how to extract it from the mass of text in which much of it resides. With the use of some textual annotation techniques, this information can be obtained.

4.2 Textual Annotation

Textual annotation is a form of assessment of student learning, one that helps the instructor identify students that are struggling or falling behind, and those who clearly understand the material and key concepts. Ideally, if there are face-to-face meetings involving all members of a project team, the instructional staff would closely observe each member, on a team-by-team basis, throughout the duration of the entire project. In this manner, the staff would have a strong understanding of all team members, their contributions, their comprehension of the material, and the processes through which the groups completed the assigned tasks. This staff understanding is obtainable in person, through several data collection methods. The most direct method, least distracting to participants, is to simply attend the meeting and observe the proceedings. Progress logs, kept by the individual students, and self- and peer-assessment questionnaires also allow the students to directly shape the understanding of the instructor. Recording each meeting, on cassette or video, for later review (or video annotation, discussed later), also provides process information about the team being observed. As class size increases, unfortunately, the course staff is less able to participate in the meetings as fully, or to gain as complete an understanding. A larger number of teams, with more participants in each, increases the information and time strains on the staff. In a collaborative, telelearning environment, course participants generate a large volume of text in the course archives, from public discussion areas, and team conferences. The text is in the form of messages from students to the instructor or to other students.

Networked learning environments frequently include internal mechanisms for tracking the actions of students and the usage patterns of course materials. Strictly in the context of forms of communication, Chapanis [1975] has identified 136 measures used in past studies of passing information between a source and a seeker of knowledge. His literature survey did not focus directly on annotation and analysis, but rather examined how communication occurs between people, through a variety of media. Extensions of his measures are currently being employed to collect just such student analysis. Harasim et al. [1995] forecast the development of an analysis component in networked class tools that would quickly summarize for the instructor statistical indicators of the activities of each student in the communication channels. The summary may include such information as the frequency of participation of individual students, including the number of new threads of discussion they have begun, the number of messages that have been posted as replies in other threads, and in which other activities they have been involved. As described in Chapter 3, WebCT already offers many of these features in its student tracking information. It lists the number of articles each student has read, and categorizes the postings the student has made into original (new threads) and follow-ups (replies in other threads). WebCT also tracks the history of student activity, the number of hits on each page of course content, and more. It does not, however, support annotation or textual analysis. An interesting question, that shall not be addressed here, is whether or not the monitoring of student activities in this way negatively impacts the students, by distracting them or causing significant shifts in their established learning patterns and goals. Does the collection of this material result in a reduction in the educational experience of the students?

One issue that arises is the abuse of the tracking system by the students. A system that is completely automated and deterministic tracks the number of messages of each student in each discussion area, but pays no heed to the importance, or even the relevance, of the student comments. Knowing this, students might post messages that have no connection with the topic at hand, or only write messages saying “I agree with the previous poster.” The automated system will dutifully indicate that the student is participating in the discussion. In this very realistic situation, textual annotation or linguistic analysis tools would serve to illustrate more accurately the level of interaction and participation of the student, rather than just the frequency.

There is a distinction that must be drawn between the analysis of student progress and participation that is described above, and the determination of process and understanding of individuals or teams of students. Evaluating, for example, the level at which the students are beginning to adopt some of the key terminology of the subject matter in their discussions requires some form of linguistic analysis, a more sophisticated measure of student learning than the tracking statistics described above [Harasim et al. 1995, Goldberg 1996]. This linguistic analysis, or perhaps more appropriately textual analysis, and message annotation present one solution for dealing with the problems of mining cognitive information from the body of messages. Such techniques are common in research in the social sciences, where transcript analysis and annotation provide insight into the minds of the study participants. As a course progresses, the on-line discussion accumulates course announcements, student questions, answers from the instructional staff or other students, intellectual arguments, heated debates, and statements of opinion. This collection of computer-based messages, an archive of postings or a digest of mail messages, provides a transcript of the progression, and the processes, of the class or the project team over the duration of the course. WebCT also logs the interaction in the Chat tool, providing another source of transcripts. The question is how to benefit from the existence of this transcript, to gain the insights promised.

Currently, no telelearning system provides this functionality as a *de facto* feature. Wolfe has created an annotation application that runs in parallel to the Virtual-U, giving the instructor or researcher a meta-view of the discussions [Wolfe 1996]. It is a Web-based system that permits a researcher to make annotations regarding the content of each message in the communication conferences. By selecting appropriate check boxes, a description is created of the content of each message, or whether the message began a thread or was a later follow-up, whether or not the content matched the subject line, and whether or not the poster employed some of the more sophisticated features that are available. The system is sufficiently flexible that researchers can define new categories as required, by creating a new reference code for the desired quality found in a message. The emphasis of the annotation tool, however, is more on obtaining insights into the threads and conferences, than into the students and teams. The tool analyzes the trends and statistics of the conference as a whole, but places less emphasis on identifying the authors of each message.

A commercially available program, which also goes beyond tracking through statistical observation and analysis, is the Non-numerical Unstructured Data-Indexing, Searching, and Theorizing package called QSR NUD*IST. It is a generic yet powerful analysis system, for the examination of qualitative data, currently in Version 4. It facilitates an understanding of material from a variety of unstructured, qualitative data sources, such as interview transcripts, logs, documents, correspondence, or other material not in computer-readable form, such as books, and even non-textual data, such as video recordings, through flexibility in the research methodology.

NUD*IST presents no limit to the number or type of data sources defined in its Data Document system, one of two databases it creates (the other, the Index system, is described below). Computer-based data can be searched, edited, and stored, with the only restrictions being imposed by the capabilities of the machine on which the system is running. Results of queries into the content of the on-line documents can be saved directly to the Index system, enabling rapid retrieval for later analysis. This provides a facility for testing hypotheses: a question can be posed, the data searched via the Index system, and the results stored, thereby adding to the knowledge base. Data sources that are not in computer-readable form are referenced indirectly by the system, in the form of notes and comments about each item. To annotate video recordings, the CVideo software, an extra plug-in component, is required.

The Index system is the heart of NUD*IST, a powerful database-driven feature that supports the creation of hierarchical taxonomies of pointers into the data, which can be used to sort the student messages using a set of user-defined properties. The resulting tree structure of the indexing system can be rearranged as needed. The power of NUD*IST is blunted slightly, because of the overhead required to effectively use the Index system. Comprehensive indexing can result in the number of index nodes, that point into the documents, dwarfing the number of documents themselves by an order of magnitude or more. This is counteracted through command files, which serve as scripts for the system, providing the ability to automate some functionality. Once the index structure is established, automated searches can link new documents into the structure with minimal effort.

The above provides a brief summary of the features available in qualitative data analysis software, specifically with respect to the NUD*IST system. For more detailed information about NUD*IST, consult Richards and Richards [1991]. Further information about other qualitative analysis methodologies can be found in Richards and Richards [1993] and Fielding and Lee [1991].

Qualitative data analysis tools permit annotation and analysis of unstructured data, in a variety of formats. In a telelearning environment, even free-form discussions have some structure to them. There is a temporal sequence among the messages, ordering them by time and date of posting. Other sequences also exist, ordering the discussion by author or subject thread. The power of a system like NUD*IST is that this inherent structure is not necessary, but since it exists, its presence can be exploited. As mentioned, the temporally ordered collection of messages serves as a transcript of the discussion held by students in the class, or in one team. The metaphor breaks down to some extent, considering that an interview transcript is entirely sequential, while several of the on-line discussions happen in parallel. This results in a fragmented discussion, when ordered temporally, with topics changing in random fashion. Alternatively, ordering the messages by topic continually jumps forward and backward through time, as one topic begins before another concludes. A qualitative analysis tool resolves this dilemma by allowing a researcher to easily access both contexts in attempting to gain an understanding. Postings made within asynchronous communication facilities are not the sole resources, in a networked classroom, for gathering transcripts. A logged, real-time chat room (as in WebCT) also provides a conversational transcript. The culture in a Chat room is generally informal, and may not be treated seriously in most cases, making the transcript less useful. Real-time meetings, however, would produce valuable transcripts for annotation and analysis.

Other annotation tools exist, which facilitate the analysis of observational data collected on video and audio. These serve to reduce the gap between our ability to collect large amounts of observational data, and our ability to analyze it. It is easy, for example, to record meetings on video tape. Vanna and Timelines are two systems that offer a set of tools for indexing key events by time stamp of the video tape, and making annotations accordingly. A more successful tool is MacSHAPA, which is specifically geared for analyzing what Sanderson and Fischer call “Exploratory Sequential Data Analysis” [1994]. The MacSHAPA system places control of the VCR at the fingertips of the computer user, possibly with the image displayed on the computer screen, and enables data to be captured from video in this way. The video tape is indexed to the data by capturing the time codes from the video. Based on these codes, qualitative observations and annotations can be created as unstructured text, and later searched as reference pointers into key moments of meetings - perhaps a debate or a key realization. The time codes are central to the linear sequence of events, and any annotations that are made. The system is less useful for non-linear analyses. Quantitative data can also be captured, and analyzed statistically, through MacSHAPA. Like NUDIST, MacSHAPA is flexible enough to allow user-defined categories to be created in the database, which can then be used manually to encode the events and annotations. Unlike NUD*IST, this tool is not a commercial product, but rather the implementation of researcher hypotheses about capturing, analyzing, and utilizing observational data. It is available with limited peer support, and full documentation. More about this research product can be found in [Sanderson 1994, Sanderson, McNeese and Zeff 1994, Sanderson et al. 1994].

Instructional staff are not the only ones who stand to benefit from transcript annotation and analysis; the students in the teams may also be interested in seeing their progress to date and their current direction more clearly using these techniques. Without the ability to simply and automatically generate index structures and basic analysis assessments, I do not believe that process analysis would interest most students, without great effort in convincing them of its benefits. Given analysis tools with substantial learning curves, or large amounts of overhead to examine the discussions and meetings, they are unlikely to make the effort. The required time to learn the system, analyze and annotate each message, and collect and report on the results would outweigh the perceived benefits to their group work. The benefits such analysis offers them, in their team collaborative projects, include an organizational memory about the processes they employed to make their decisions, an awareness of the relative participation and contribution toward the corporate goals of each team member, and an understanding of where more effort is still required. While this has not been tried, employing the automated scripts of NUD*IST might reduce the overhead sufficiently to enable students and teams to benefit in these ways.

Chapter 5 -- Lessons

Chapter 2 describes the CPSC 319 Software Engineering course, highlighting the goals and methodologies employed in the team project. Other chapters examine various tools for delivering course content in an on-line environment, focusing particularly on generic communication tools such as electronic mail and newsgroups, and on the Virtual University and WebCT. This final chapter describes some of the specific ways in which the technology has been employed in the CPSC 319 course. From the observations in this particular course setting, and from the information given in previous sections, some lessons are identified. They are drawn from the perspectives of three distinct groups: the instructors, the students, and the researchers or developers of new systems.

As described in Chapter 2, CPSC 319 is a course populated by undergraduate students, generally enrolled in the computer science department. This creates an expectation that the students will be comfortable with computers, and will be familiar with basic communication tools like e-mail and news readers. Little or no encouragement is required for the students to communicate within teams through e-mail. They naturally begin to announce meetings, define divisions of labour, and coordinate their activities through this medium. The course provides a newsgroup that is dedicated to class discussion and asking questions of the instructor. Participation is not a requirement of the course; the newsgroup serves as an optional forum, a useful facility for a class that is too large (approximately 150 students) to easily hold meaningful discussions in a lecture hall. Some course content is placed on-line, accessible from the course home page. Available content includes administrative information, lecture notes, tutorials on various tools and concepts, sample code fragments, and links to other available on-line information resources for further reading. A real-time chat tool is available to the students through their regular computer accounts, as are the newsgroup and e-mail facilities. This year, for the first time, the course provided access to a telelearning tool. Each student was given access to VGroups, the Virtual University conferencing system. At their discretion, teams could opt to take advantage of the control, privacy, and structure provided by VGroups. Teams were observed, during the past two years, in a "field study" mode [McGrath 1995]. Some observations and lessons are detailed below.

5.1 Instructors

From the point of view of the instructor and staff of CPSC 319, there are three concluding observations drawn. The definition of the task to accomplish, the creation of teams, and the provision of on-line communication facilities do not guarantee participation, either within the team, or on an individual basis. A second observation is that in a group setting, it may be necessary to ensure a minimum level of course content is learned by individual students. And third, to overcome the challenges presented by the large classroom setting, instructors should develop methods of monitoring the processes and progression, on the basis of teams and individuals.

5.1.1 Participation

The first lesson learned, from the perspective of course instructor, is that student participation in an electronic forum is not guaranteed, just because the forum is available. This should not be a surprising observation, as it has been noted by other studies of computer-mediated communication in educational settings. Hiltz noted that, with respect to the students using the Virtual Classroom, students will place the majority of their efforts in the activities that correspond directly to the grading scheme; making on-line interaction an optional activity brings only a small subset of the students into the discussion [Hiltz 1995]. The experience of the British Open University, with optional participation in their course-related conferences, was that the students divided into three usage patterns, with approximately equal distribution [Mason 1990]. One group of students, the Active group, participated in the conferences by posting at least one message. A second group, of Passive readers or "Lurkers," participated by reading the postings, without contributing any themselves. The third group, the Non-Participants, made no use of the conferencing system.

Since participation creates active and collaborative learning situations, a greater overall level of participation is beneficial to all students. Instructors can encourage more students to contribute in the on-line activity by clarifying its objectives and indicating how it relates to the rest of the course. Access to adequate computer resources is essential for increased student participation. Ensuring equal, or even widespread student involvement may require of the instructor a careful monitoring of student activity, especially in the initial stages of the course, intervening when necessary. Harasim et al. [1995] suggest that voluntary participation ensures a low rate of involvement. They suggest that a reward for involvement be offered, in the form of marks.

The experience of CPSC 319 centres around the course newsgroup. There were no facilities in place for monitoring the electronic mail conversations of the students and teams. The policy of the department is that students are permitted to utilize their e-mail accounts for personal matters. Any monitoring of their messages raises ethical dilemmas regarding privacy rights, all the more because of this policy. The newsgroup, on the other hand, is a voluntary, class-wide, public forum for discussing concepts and questions. The newsgroup archives of the past two years (1995-96 and 1996-97) were examined, encompassing the two most recent offerings of the course. It was expected that the ratio of Active students would be greater than that reported of the British Open University. The basis of this expectation was the concentration of students in the course from a Computer Science background, who were familiar with computers. Prior to beginning the 1996-97 course, over 80% of students described themselves as somewhat or very comfortable in using e-mail, newsgroups, and the Web. This expectation was confirmed, as a high percentage of students did indeed post to the newsgroup: a total of 141 out of nearly 300 students (48%) posted at least one message. Thus, the Active group has a greater proportion than reported in [Mason 1990]. This is shown in Table 1, which refines the Active group into three sub-categories: one-time-only people, with a single post all year, occasional participants, who posted between two and nine times, and regulars, with a double-digit number of messages. Student totals come from the registrar's class list.

Table 1: Student Newsgroup Posts

Item	1995-96		1996-97		Combined	
Students with 1 post	36	24 %	22	15 %	58	20 %
Students with 2-9 posts	42	28 %	21	15 %	63	22 %
Students with >9 posts	6	4 %	14	10 %	20	6 %
Students with 1+ posts	84	56 %	57	40 %	141	48 %
Total # students	149		144		293	

It was expected that the number of Passive students who followed the discussion, making no posts, would also be large, given the high level of newsgroup activity, and the student computer comfort. The number of Non-participants is then expected to be small, although given the voluntary nature of the participation, it is unreasonable to expect the last category to be zero. Information regarding the number of students who did not read the course newsgroup may be obtained through their news reader file, indicating which articles they have read from each newsgroup. As with e-mail, however, this raises ethical issues.

Instead, the class was given a questionnaire, to gather an understanding of the usage and utility of various course tools. The questionnaire is listed in Appendix C. One question asked students about their reading habits concerning the newsgroup. Table 2 summarizes their responses. As anticipated, the set of Non-participants is small but non-zero (6.3%). Given that nearly one student in two posted to the newsgroup, and barely 6% admit to never reading the discussions, it can be inferred that 40-50% of students probably read as Passive participants. An interesting question is how this data correlates to the attendance and participation in the weekly lectures, but that will not be addressed here.

Table 2: Student Newsgroup Reading Habits

Answer	# Students	Pct
I never read any articles	4	6.3%
Only messages from course staff	13	20.3%
About 1 article in 3	20	31.3%
About 2 out of every 3 articles	7	10.9%
I read every article	20	31.3%

Table 3: Number of Posts by Type of Participant

Participant Type	1995-96		1996-97		Combined	
Students: One-time posters	36	6.7 %	22	3.3 %	58	4.8 %
Students: Occasionals (2 - 9)	162	29.9 %	85	12.8 %	247	20.5 %
Students: Regulars (10+)	118	21.8 %	276	41.7 %	394	32.8 %
Students: Totals	316	58.4 %	383	57.9 %	699	58.1 %
Staff: Instructor	95	17.6 %	171	25.8 %	266	22.1 %
Staff: TAs, tech support	113	20.9 %	85	12.8 %	198	16.5 %
Staff: Totals	208	38.4 %	256	38.7 %	464	38.6 %
Non-Course personnel	17	3.1 %	23	3.5 %	40	3.3 %
Grand Totals	541	100 %	662	100 %	1203	100 %

Students were not the only people posting messages to the course newsgroup. The course instructor, the technical support staff and teaching assistants, and several people with no course involvement at all, occasionally posted messages. Table 3 summarizes the numbers associated with each group mentioned, showing their share of the total discussion.

Despite a 20% increase in the total number of posts made in 1996-97 over the previous year, the ratios held surprisingly constant. Despite this similarity, however, there are some notable differences, between the two years, with respect to the participants in the newsgroup discussions:

· As seen in Table 1, there was a large drop in the number of students who made at least one comment on the newsgroup, from 84 to 57. Yet despite the lower number, there was a large increase in the number of “Regular” students who made ten or more posts during the course.

· The set of student “Regulars” posted 118 messages in 1995-96, representing 37% of all student posts, and 21% of all newsgroup posts. The Regulars held much greater influence over the shape of the newsgroup discourse the following year; their 276 posts represented 72% of all student posts, nearly 42% of the entire newsgroup, doubling both ratios.

· There was a change in the shape of the participation of the course staff, as well, as there was a change in instructor between the two observed years. The differences in the instructors created changes in the staff style of participation in the newsgroup. In 1995-96, the course was taught by a faculty member who frequently traveled on business, while a Ph.D. candidate, closer to the age of the students and with more time to devote to the course, was the instructor in 1996-97. The latter year provided an instructor who could respond to newsgroup questions in a more timely fashion, and whose authority may have been perceived as less entrenched. He was able to respond the same day to questions posed in the newsgroup. This resulted in the instructor posting 171 times, accounting for 67% of staff posts, and 26% of all messages posted. This compares with the previous year, 1995-96, in which the instructor posted 95 messages, representing 46% of staff posts, and 18% of all posts.

· Table 4 identifies the content of the posts of each instructor. A posting may fall into zero or more of the categories listed. Clearly in the second year of observations, the instructor placed a greater emphasis on the newsgroups, employing the system as a forum for making announcements, and replying to a larger number of questions.

Table 4: Content of Instructor Posts

Posts included:	1995-96		1996-97		Combined	
Announcements	31	33 %	81	47 %	112	42 %
Questions	8	8 %	3	2 %	11	4 %
Answers	42	44 %	88	51 %	130	49 %
Clarifications	24	25 %	31	18 %	55	21 %

The shape of the student participation also changed. There were fewer participants, but a significant increase in activity per student (3.76 messages per participating student in 1995- 96, increasing to 6.72 the next year, p=0.02). This suggests that the newsgroup was put to different use in the second year, than in the first. It would be reasonable to expect fewer threads of messages wherein a question is asked, an answer is given, and the thread ends. The above observations suggest the newsgroup became more geared to class debates and discussions, at least among the active posters, than previously. This is indeed the case, as seen in Table 5. It identifies four message thread categorizations, which generalize the related threads by the number of posts. An announcement is considered to be a one-message thread, which requires no follow-up. Two-message threads are generally questions, or perhaps requests for clarification, and the related response. A short discussion falls between three and five messages in length, indicating a limited exchange of ideas among participants, while a long discussion exceeds five messages, suggesting a more lengthy debate, as participants build on one another, or maybe argue over trivia. These are artificial categorizations, since not all threads grouped in the "Announcements" category were necessarily announcements. The common element is the thread length of 1. In 1995-96, over three-fourths of the threads were of length 1 or 2. This percentage drops to two-thirds of the threads in the second year, suggesting an increase in discussion on the newsgroup.

Table 5: Thread Length

Category	Length	1995-96		1996-97		Combined	
Announcements	1	121	50.8%	92	37.4%	213	44.0%
Questions & Answers	2	62	26.1%	76	30.9%	138	28.5%
Short Discussions	3 - 5	39	16.4%	59	24.0%	98	20.2%
Long Discussions	> 5	16	6.7%	19	7.7%	35	5.2%
Totals		238		246		484	
Avg. posts per thread		2.27		2.69		2.49	

With the greater frequency of instructor participation mentioned earlier, there was an initial expectation that the average length of the threads would be shorter in 1996-97 than in the previous year. Table 5 shows clearly that this expectation was not met. Factors which may have had a role in this unexpected result include the newsgroup achieving a minimal set of participants (14 "Regular" students), timely replies by the instructor encouraging greater student usage, different perceptions of the new instructor in 1996-97, and even more gregarious students - a year with talkative students will create more messages than otherwise, regardless of instructor involvement. The data suggests that the newsgroup served as a source of course-related announcements. There is also evidence it provided a mechanism for interaction between the staff and the students, for asking and answering questions. These features are evident in both the first and second years. On the other hand, the increased number of regular participants (6 to 14), the increased average number of messages per active student (3.76 to 6.72), and the increased average length of thread (2.27 to 2.69) all suggest that in the second year the newsgroup became more of a forum for a subset of the class to discuss course-related issues and concepts. As mentioned in earlier sections, it is in the active exchange of ideas that collaborative learning occurs. Unfortunately, the ratio of active participants to the whole class (Table 1) indicates that, even in a friendlier, discussion-oriented forum, voluntary participation offers no guarantee that all students will benefit from interaction.

In 1996-97, students were provided with access to the VGroups computer conferencing system, part of the Virtual University. The tool was given to them as an optional communication facility; it was not required that they participate in the conferences, but they were provided with accounts, and an initial conference hierarchy was defined for every team that expressed a desire to communicate through it. Teams, designated by their course-assigned letter (Teams A to R) were each given a private conference, just for them and their TAs, and were provided with permission to create whatever sub-conference structure they wanted or needed. A generic area was dedicated to course-wide discussions, similar to the course newsgroup. Nine teams (C, H, J, L, M, N, P, Q, R) elected to explore the benefits of VGroups to the internal communication processes. Table 6 summarizes the number of messages posted by each team in the VGroups conferences. Surprisingly, VGroups was quickly abandoned, within a few days or weeks, by almost every team that tried it. Team R, which led all teams in number of posts to their conference, communicated very actively through VGroups for two of the eight months of the course, before stopping. Team Q had the longest period of time over which they employed the system, but their usage was sporadic. Two teams, C and L, indicated a desire to use the system, yet never posted a single time. The remaining nine teams, not listed in the table, informed the course staff that they had as a team elected to not experiment with the VGroups system. Even the course conference, to which all students were provided with access, was little used.

Table 6: Team Usage of VGroups

Team	# Posts	1st post	Last Post
C	0	-	-
H	1	13-Nov-96	13-Nov-96
J	3	22-Oct-96	01-Nov-96
L	0	-	-
M	4	02-Nov-96	15-Nov-96
N	3	19-Oct-96	27-Oct-96
P	3	29-Oct-96	31-Oct-96
Q	14	28-Oct-96	07-Feb-97
R	51	21-Oct-96	06-Dec-96
319 Conf.	12	18-Oct-96	11-Dec-96

In surveying the students regarding their concerns, to ascertain what factors caused them to either stop using, or ignore altogether the VGroups system, two reasons emerged. First, students felt that adding another communication tool, separate from all others they were using for their collaborative course work, required that they check yet another place for course-related information. Over 42% of respondents to the survey (listed in Appendix C) indicated that VGroups was just one more source to check for messages and information, an unnecessary addition with a newsgroup, Web site, and personal e-mail already. Second, many felt the delay to load a Web browser and navigate to their VGroups conference constituted time spent for nothing, with no new capabilities gained. As shown in Table 7, fully half the students perceived VGroups as a redundant duplication of communication capabilities they already had. These responses were selected from a list of possible views of the VGroups system, from which the students were asked to check all that applied. Clearly, providing additional tools to assist in group communication does not ensure they will be used by the students.

Table 7: Student Reasons to Not Use VGroups

Reasons Indicated for not using the system	Percentage
It duplicated existing communication tools	50.0 %
One more source to check for communication	42.2 %
Nobody else used it	37.5 %
It was too slow	21.9 %
No other course material was there	20.3 %
It is not a UBC product	17.2 %
No reason!	12.5 %
It sounded too hard to learn	10.9 %
Researchers are doing studies with it	6.3 %
A chance the interface might change	4.7 %
Never heard of it	3.1 %
Other reasons	17.2 %

5.1.2 Minimum Learning

A second lesson learned is that instructors should take care to ensure that all students reach the minimum set of learning objectives. In a group setting, sub-teams are created to address various aspects of the project. A common tendency was to form the sub-teams according to existing skill sets, or desired learning outcomes on an individual basis. The potential exists, with this situation, to encourage distinct learning streams, where some students gain experience in group planning and management, others in documentation creation, while still others experience the software engineering life cycle from the perspective of a coder. If there are key concepts that all students in the course are to learn, these should be taught separately from the group project itself. In CPSC 319, key learning goals were that students develop an understanding of, and some experience with, group skills, project management, meeting management, current software engineering tools and practices, technological leadership in problem solving, and issues motivating graphical user interface and database development skills.

To reach these objectives, lectures were given on each of the above subjects, at the desired basic level of understanding. Thus, all students were exposed to the important themes of the course. Three quizzes were given in class, to encourage students to learn the basic material, and to assess the level of their comprehension. The course Web site contained more specific information, and links to other on-line resources, for students and teams who wished to learn more. The goal of having the students experience these activities was reached by having deliverables, such as project management artifacts (minutes, agendas, actual versus estimated time requirements), which the teams were required to produce.

Another approach that addresses the concern that students will learn only their specific part of the material is described by Hymel et al. [1993]. They suggest that students be required to teach to the rest of their group some of the material they have learned. Such an activity, they show, ingrains the material more deeply into the mind of the teaching student, and exposes the fellow team members to the material. A quiz could again be used to assess the understanding of the team, and the success of the teaching student.

A final approach is taken from the observations of the previous section. Optional discussion group activities are unlikely to attract all students. These groups, however, present an ideal opportunity for the students to wrestle individually with some of the key issues and concepts that are being addressed in the course. A required discussion group, with a portion of the grade dependent on student involvement, would help to ensure exposure to the topics and individual student interaction with each of them. Twice, in the course newsgroup in 1996-97, the course staff explicitly invited students to reflect on some key issues in the projects being built, and to share their thoughts. Both times, the conversations went nowhere, with either no replies, or else follow-ups indicated a misunderstanding of the topic at hand.

On a third occasion, one topic arose because it directly and immediately impacted the work the teams were doing. A long and fruitful discussion arose in this case, between the instructor and several students. The implications and impacts on the work of the students were clear, sparking a lively debate. Yet when the instructor mentioned that the additional work would not be required, and conceded that omitting it would not affect the grade, most students left the discussion thread. This was because participation was not required, and the students had the answer to their questions of how the issue would impact their grades. By the end of the discussion, it was clear that one student, who had continued to wrestle with the issues until he fully understood, had learned the lesson well, while also demonstrating his problem-solving skills. A second student was also an integral participant in attempting to understand the full ramifications of the problem. See Appendix D for an annotated transcript of the conversation. Hopefully, the passive readers also learned from the exchange. Such issues could be taught to all students in lecture, but the lessons are more meaningful when the students arrive at the conclusions themselves, getting the process, not just the end product of the knowledge. This could be encouraged by having required on-line discussion groups, where the TA raises such issues, and students are expected to participate in the discussions. In 319, the project teams could be assigned to discuss some important topics in their meetings, or all students under a given teaching assistant may form a larger, perhaps on-line discussion group.

5.1.3 Monitoring

The third observation, from the perspective of course instructors, is that it is necessary to be able to determine what students are doing, and how well they are progressing through the project milestones and the course content. An imposing impediment to successfully monitoring student progress is the size of the class. A large enrollment increases the instructor time requirement, to track students on an individual or a team basis. If the course emphasizes active participation in on-line discussions, a minimum of ten students are needed for quality discourses, but about thirty is the maximum that an instructor can handle properly, with due attention. Several systems provide the instructors with automatic monitoring facilities, helping them to identify which students are falling behind, or are participating too much or too little. The WebCT tracking facility is discussed in particular, in Chapter 3. It allows the instructor to identify students that have fallen behind in the readings, or are participating the most or the least in the discussions. Another solution, discussed in Chapter 4, may be automated or manual annotation of such time-sequential information as text-based messages, or video recordings of team meetings. Another tool, a generic one that would require some adaptation for specific educational situations, is the Phoaks system (People Helping One Another Know Stuff). It is a generic engine, suited for processing on-line conversations, which mines key items from the data and presentes the results [Terven et al. 1997]. The current implementation can cull key information, such as Web site recommendations, from over 100,000 news posts daily, far larger than a class would produce. Applying the Phoaks system to course on-line discussions may be overkill, but the sequence of processes it follows certainly applies. The system searches the posts for a specified textual pattern, and preserves surrounding information, to retain the context of each occurrence. Rules are used to categorize every occurrence of the search pattern, which is then handled in a specified, category-dependent manner. The potential is that the system will identify key terms or concepts, classify their usage, whether in a questioning role, opinionating, and so on, and automatically deal with them accordingly, perhaps updating monitoring measures about the author, or filing the post in an index system.

5.2 Students

From the perspective of a student enrolled in CPSC 319, there are four observations to note, regarding the state of the computer-based tools that are made available for the course. To take full and effective advantage of the available communications tools, a critical mass of participants must be reached, whether for internal team coordination, or class-wide discussions. To be effective as a team, adequate electronic meeting facilitation tools must be available. Third, there should be strong group work facilities, enabling them to collaborate on artifact creation and maintenance. Finally, estimation is an important part of software engineering, and tracking and revision tools should be available.

5.2.1 Critical Mass

A key observation, from the perspective of the students, is the requirement of a critical mass of participants to make a communication medium effective. This is one of the eight key groupware challenges identified by Grudin [1995], and it is a reasonable assumption that communication tools in an academic setting would also require a certain minimum level of participation to be effective and practical. Hiltz suggests the minimum is ten students [Hiltz 1995], but applies this number to the class size, not to the number of participants, and assumes all students will participate.

In CPSC 319, as has been shown above, the newsgroup was well used, with 662 messages over the course of 1996-97. VGroups, however, was used little in the first half of the course, and almost not at all in the second half. A significant reason for this was the lack of students using the system - the “critical mass” was not reached, and those students who wanted to make use of it were forced to communicate with their peers through other means. Table 7 lists the reasons students cited for not using the system. It is notable that the third most common reason was that no other students were using VGroups. Even the course-wide conference received only 12 posts (Table 6). One cause of the observed phenomenon is that teams were asked to decide, on a group-by-group basis, whether or not they wished to use VGroups. Individual students, in the teams that decided against its use, then had no reason to check the course-wide conference. Instead, these students used the course newsgroup for such discussions. With this large number of students holding their discussions elsewhere, outside the VGroups conferences, pressure mounted on the remaining students to also abandon VGroups in favour of the newsgroup and their private e-mail systems. A discussion conference for the entire class seemed pointless, when most students were not participating, and eventually all students abandoned the conference. Such a student reaction is not unique to the VGroups conferencing system. The pre-requisite course, CPSC 310, was offered in Mixed Mode last year, with on-line material and communication tools provided by WebCT. During the course, there were only 27 messages posted to the WebCT discussion forum. One poster summed up the lack of activity in this way:

Student 1: Does anyone ever read [this WebCT] bulletin board?
Student 2: No, there’s a news group for us, ubc.courses.cpsc.310.

Indeed, during the same course period, over 650 messages were posted to the course newsgroup, completely separate from WebCT. In both courses, students preferred the newsgroups to the Web-based conferences, perhaps going with what they already knew how to use, more likely because the Web-based communication tools were seen as duplicating other tools that were perceived to work as well or better. One student called on his fellow classmates to discuss which to use, and summed up a very common view as follows:

> "I think it's about high time that we decided on a forum for all our discussions, be it this bulletin board or the newsgroup. As far as I know, there's no way to post one message to both forums simultaneously. This way, we'll ensure that everyone is seeing the same information, questions, etc. For example, the info on where to submit our assignments was posted to the news-group, but not to this board.
>
> My personal preference would be to settle on the newsgroup. Most people have a [dial-in text-based (10 hours a month free)] account, and thus, can dial-in from home to read news 'n' stuff. I realize that [the new service of graphical access (5 hours a month free)] allows us to use Netscape, which in turn lets us see this bulletin board, but I don't think anyone would want to use up twice as much of their dial-in quota while reading text only."

Table 8 shows that student satisfaction with the course newsgroup in 1996-97 was high. They were asked five questions, regarding their attitudes about the effectiveness of the course newsgroup in allowing them to ask questions and discuss issues. While not unanimous, students generally seemed pleased with the newsgroup as a platform for class discussions. Over half the respondents agreed or strongly agreed that it provided timely information, was useful in issue discussion, and gave them a voice to which the course staff listened.

Table 8: Student Attitudes Towards the Course Newsgroup

Question	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
It provides course information in a timely fashion	4.7 %	12.5 %	31.3 %	45.3 %	6.3 %
It has been good at answering my questions	3.1 %	10.9 %	42.2 %	32.8 %	10.9 %
It is useful for discussing course-related issues	3.1 %	12.5 %	23.4 %	45.3 %	15.6 %
It is dominated in a bad way by a small group of people	7.8 %	28.1 %	43.8 %	12.5 %	7.8 %
The course staff listens to us in the course newsgroup	1.2 %	4.7 %	25.0 %	35.9 %	32.8 %

5.2.2 Meeting Facilitation Tools

A second lesson learned from the perspective of the students is that when used in a formal team environment that expects student teams to mimic industry work groups, a collaborative system requires tools to more effectively support meeting facilitation. In CPSC 319, there was an emphasis placed on developing proper project management skills, and learning how to run effective team meetings. The progress of the students towards more effective meetings was tracked via bi-weekly deliverables, wherein the teams were required to provide evidence of determining an agenda, keeping meeting minutes, and tracking their own project progress. The meetings involved status reports from team members, presentation of issues and risks for group discussion and decision, seeking to understand problems and their solutions, assigning tasks and action items, and voting on key decisions with no “correct” solutions.

In a mixed-mode setting, such as CPSC 319, students are able to hold their team meetings face to face. A system through which they can collaborate, then, should provide the capability for them to broadcast their agendas prior to the meetings, and post their minutes following the meetings. Access to an HTML editor, and a common team file space can provide this functionality to the teams in a Web environment. If the course is entirely on-line, however, teams need features built into the system that will allow them to hold their meetings in a formal, efficient way. Such functionality would also be useful to teams in mixed mode. One student, apparently misunderstanding the use of the word “conference” in VGroups, posted to his team “how about an on-line meeting tonight, to discuss [this issue]?” Clearly the interest is present, in mixed mode classes, to use such features. When the team discovered that VGroups was completely asynchronous, making meetings more difficult, they became less enamored with the conferencing system, and stopped using it altogether shortly afterwards.

To hold formal meetings on-line, teams need access to real-time communication facilities, such as chat rooms. They need to be able to exclude non-members from the chat room, to ensure greater privacy and less disruption. An on-line meeting is less likely to be dominated by a strong personality, and all participants are less inhibited, but text-based interaction results in more time elapsing before decisions are made, perhaps as a result of typing speeds [Kiesler et al. 1984]. If the chat room can be logged, then the team can easily capture discussions, creating a perhaps long and wordy organizational memory, to preserve the reasoning behind their decisions. An organizational memory is particularly useful on a long-term project, or when the software product enters the maintenance stage of the life cycle (although even a full-year course has proven too short to achieve this stage of the software life-cycle). A more succinct way of preserving this information, omitting the conversational details while still preserving the logic and the key information, is the Issue-Based Information System (IBIS), described by Yakemovic [1990]. It uses a brief system of manual (or computer-based) notation, structuring discussions into issues, arguments, and positions. It is a refinement of an existing process (taking meeting minutes), and uses simple, familiar technology (a plain text editor or pen and paper). Yet the system captures the "why" of decisions, by ordering the large amounts of unstructured information into a usable form. A software implementation of the system has been developed, using hypertext to easily access the stored information [Conklin and Begeman 1988]. The utility of such a system extends beyond maintenance, into the analysis and development phases as well. Providing such a capability, or some other forms of decision support tool, would likely enable on-line teams to better structure their meetings. A voting mechanism is also an important meeting support tool that is absent from collaborative education systems. If the voting process is not automated, but rather continues through generic communication facilities, the appointed counter of votes faces greater communication overload. Automated voting facilities could also permit powerful scaling and ranking methods, enabling teams to analyze several options, on several different scales, and help to identify where there is disagreement, or what issues still need to be resolved [Harasim et al. 1995, Swigger et al. 1995]. For such activities, text-based tools may be inadequate; more sophisticated group tools, encompassing a wider set of modalities and media may be required.

5.2.3 Artifact Creation

A third student-related issue is the presence of tools for the creation and maintenance of course-specific artifacts and deliverables. When team work is involved, on-line systems must provide file space and links to the areas reserved explicitly for each group. Some studies have observed that collaborative work is sometimes done independently by students who then tell their teammates “here is what I have done” and merge all their work (see Chapter 4). Collaborative document creation tools and groupware exist already, for both autonomous and truly collaborative modes [Edwards and Mynatt 1997, Lai, Malone and Yu 1988, Roseman and Greenberg 1996]. In a software engineering setting, collaborative tools are needed for document composition, so students can compose and maintain requirements analyses, interview transcripts, module decomposition, and more. Other artifacts require tools that enable source code generation and maintenance (revision control), and project management artifacts (GANTT charts). If the course is entirely on-line, such tools as those mentioned above are vital for students to be able to complete their projects. In CPSC 319, three quarters of the students expressed belief in the importance of working in the same physical location as a teammate (Table 9). Given the mixed mode of the course, this suggests a combination of student preference for working in the same place, and perhaps a lack of tools supporting collaborative work. Students were not surveyed, however, about their attitudes regarding whether or not they were provided with enough tools to complete their projects.

Table 9: Student Attitudes of Working in the Same Room as Teammates

Category	Percentage
Very Unimportant	4.7
Unimportant	4.7
Neutral / No Opinion	15.6
Important	29.7
Very Important	45.3

5.2.4 Student Self-Tracking

The fourth observation of student needs, specifically within projects, is the ability to track progress against estimations. Students, in creating their proposals, are requested to estimate what resources will be needed to complete their project, including technological, human, and temporal requirements. It was not expected , in CPSC 319, that students be expert forecasters of the time investment required to complete their project, as even seasoned software developers tend to estimate poorly, and have difficulty in monitoring their schedule [Brooks 1982]. On the contrary, they are expected to be incorrect in their estimation because they are learning a new skill, and have little or no experience or data from previous people and years on which to base them. Tools should be provided, therefore, to enable them to monitor their progress, and to make schedule adjustments accordingly.

The CPSC 319 course was structured to encourage students to take this tracking seriously. Bi-weekly deliverables required them to report their current status, and if it did not match their estimated schedule, to explain the differences, and describe in what ways the schedule would be adjusted accordingly. If the schedule was pushed too far off course, students were encouraged to determine which project components could be left out. With proper monitoring and justification, students could drop some functionality of their final projects, while losing neither project nor management grades.

Students using collaborative education systems geared for group work would benefit from the ability to track their progress through the on-line system. This might come in the form of a time tracking and estimation assistance utility. A timesheet feature, with either the ability to detect and populate fields automatically, or to have each member manually enter their tasks and time commitments, would facilitate team progress tracking. Teams could then determine, at an earlier point, whether or not they were falling behind schedule. One team, anticipating these benefits, assigned one of their skilled coders to implement a time sheet facility for their internal use, run off a private, secure Web server. Team members could then record their hours for each assigned task. The team culture developed an expectation of faithfully submitting time sheet information. Another script then automatically generated up-to-date charts and reports, enabling them to see where the schedule was slipping. The team expended much effort in creating this system, but ultimately benefited from it in their project management. Providing this functionality within the course tool set would enable more groups to reap similar benefits.

5.3 Researcher/Developer

From the perspective of a developer, creating new tools and systems for an educational setting that relies heavily on group work, and from the point of view of a researcher into the effectiveness of these tools in meeting their defined educational expectations, there are four observations. There is interest in having the systems accessible beyond the borders of the campus. Second, by integrating many aspects of course-related work into a structured environment, convenience of access, and learning benefits are foreseen. Third, the question of who controls the message space has a potentially important impact on the pragmatic usage of the tools. Finally, the Virtual University and WebCT address many of the important aspects of an integrated and collaborative learning system, but still have room for improvement.

5.3.1 Accessibility

One lesson learned from the perspective of system developer is about accessibility. The educational system should be available across a variety of hardware platforms, and should not be restricted to machines that are physically present on campus. When courses are delivered with a mixture of traditional and computer-based elements, students must gather in a common location for the traditional aspects, and it is possible to have supplemental material accessible strictly from campus machines. The evidence from CPSC 319, however, is that providing the material in a format that can be accessed from off-campus computers results in wide-spread and frequent usage from home or work. Only the course Web server was explicitly made available across hardware platforms in both observed years, but the generic tools of e-mail and newsgroup readers are available through Telnet sessions or Internet connections. Table 10 shows the frequency with which students performed computer-mediated and course-related communications from a computer located off-campus. Over 80% capitalized on this capability, at least occasionally. Four in every ten respondents to the survey reported using off-campus resources once a day or more, to access their personal e-mail or the course newsgroup.

Table 10: Frequency of Off-Campus Access

Approximate Frequency	Responses	
Never	12	18.8 %
Once a Month	8	12.5 %
Once a Week	16	25.0 %
Once a Day	11	17.2 %
Many times a Day	15	23.4 %
Always	2	3.1 %

The two systems examined in detail in earlier sections, the Virtual University and WebCT, are both Web-based platforms. Thus, they are accessible from any Web browser, on any hardware platform, anywhere in the world, although Java-capable browsers are recommended to access all available features [Goldberg 1997]. Both systems capitalize on the flexibility of the Web to meet this recommendation of off-campus accessibility, and deliver their content, administrative functions, grading and communications facilities without even the requirement of a Telnet session.

5.3.2 Integration

The Virtual U and WebCT partially meet the second recommendation from the perspective of researchers and developers, that the systems be integrated. Integration can mean blending computers into the classroom to assist learning, a difficult challenge [Hunter 1995]. A different meaning, simpler perhaps, is integrating the features and utilities into one location, which are provided by WebCT and the Virtual U. One hypothesis concerning the lack of usage of the VGroups system in CPSC 319 is that, as far as the course was concerned, it was nothing more than a computer-based conferencing system. Had course content been available through the Virtual University, instead of from a separate Web server, and had administrative information and other course activities also been accessible there, the students would have been more inclined to communicate through VGroups. Speed may have been an issue, as well, since the software was run on a server at Simon Fraser University, not at UBC. Obviously, there are more factors in play, as seen earlier in Table 7, but it is notable that over one in five students stated that the lack of other course materials was a factor in their decision not to use it. Thus, beyond the conferencing benefits (which were not perceived to be important by the students), there were no other reasons to use the Virtual University. Students provided more evidence of the importance of centrally integrating all tools, content pages, and communication facilities. As shown in Table 11, three quarters of the students felt it was somewhat or very important to have everything in one place.

Table 11: Student Attitudes Regarding Feature Integration

Category	Responses	
Very Unimportant	2	3.1 %
Unimportant	1	1.2 %
Neutral / No Opinion	13	20.3 %
Important	15	23.4 %
Very Important	33	51.6 %

Some researchers have suggested that integrating some concept-structuring tools into on-line communication facilities enables students to better frame their thoughts. Reflecting on the purpose of the asynchronous messages, when they are being created, encourages students to compose more meaningful comments [Hmelo et al. 1995]. A structuring tool may take the form of a memo template to be filled in, or computer-suggested phrasings and concepts to include in the body of the messages. With such pre-structuring, the senders would benefit by clarifying their thoughts, the recipients would benefit from clearer messages, that are more easily organized, classified and archived, and everyone would benefit from a reduction in the ambiguity of communication. Table 12 summarizes the thoughts of students about pre-structured templates, as indicated by the responses to the survey (see Appendix C). Half the respondents felt it would benefit their team communication.

Table 12: Student Attitudes about Pre-structured Message Templates

Category	Responses	
Very Unimportant	5	7.8 %
Unimportant	5	7.8 %
Neutral / No Opinion	21	32.7 %
Important	19	29.7 %
Very Important	14	21.9 %

5.3.3 Control

A third issue for consideration among researchers and developers of on-line collaborative learning systems is who controls the asynchronous message space. This issue could extend as well to who has power to create and/or correct course content materials. Perhaps more important than the creation, is the removal of messages and content - who has the power, or the right, to remove or correct erroneous information? There are pragmatic reasons why the removal of messages may be desired. If the discussion contains a posting that is exceedingly harsh, insulting, or uses inappropriate language, the instructor may wish to have it removed, to preserve a tone of civility in the discussion. If students post solutions that are later determined to be wrong or embarrassing, they might wish to have them removed before the whole class sees their mistakes. Having the power to remove the posts may have different importance in different classes. A course with a civil or unemotional culture to the messages may apply a lot of pressure to have the offending post stricken from the record. In a small class, discovering an erroneous post and removing it may come too late, as most of the students may have already seen it. A large class, on the other hand, is more likely to have students who had not yet seen the post. An instructor may prefer to correct mistakes by further discussion, rather than by removing the offending, embarrassing, or erroneous posts. To whom this power is given is an important decision for developers to make. There are four candidates: the system administrator, the course instructor, other course staff members, and the students. The answer to who has the right or the power can be any one or a combination of these four groups, or none at all. In the Virtual University, VGroups posts can only be removed from a conference by the system administrator. The instructor and teaching assistants are equal participants in the conferences. Outside the conference area, course content can be canceled by the creator, by removing the file or by breaking the links. The WebCT system places all authority in the hands of the course instructor, who is able to remove any posts or material deemed inappropriate. The students and teaching assistants do not have this power. The central difference between the two is in the control of the asynchronous message board. The administrator wields the power to remove messages in the Virtual-U, the course instructor in WebCT. In an ordinary newsgroup, the creators of the postings have the power to cancel any of their messages. As mentioned previously, e-mail systems send separate copies of the message to each recipient, removing the ability for anyone to cancel their messages, once sent. Perhaps a flexible structure is most sensible. In certain situations, the instructor may wish to wield this power, or may want it available to all students, or to no one. From the software system’s perspective, students, instructors, teaching assistants and administrators could share this authority, until the instructor or system administrator revokes the privileges for one or more of the user types. To some extent, this moves the issue to another location, without resolving it. Rather than dealing with who has the power to cancel postings, the question becomes who makes the decision regarding who has this privilege.

5.3.4 A Comparison of WebCT and the Virtual-U

The National Science Foundation has offered some suggestions regarding desired features in educational software, and suggested areas for future research [Guzdial and Weingarten 1995, Reisbeck 1995]. These are not necessarily suggested as features of vital importance, which must be incorporated into the existing or future course delivery platforms, but rather are offered as reminders of how far this technology has come, and how much more still remains to be accomplished.

Some recommendations are for specific system interaction methods. They stress the importance of using multiple media to convey the course materials and learning experiences, something that is being accomplished more and more, in CD-ROMs and through the sound and video links available through systems like the Virtual-U and WebCT. Alternative input modalities, like voice and gesture, would provide new possibilities for interacting with the computers and with other students in different locations. Adaptable user interfaces would enable the material to be presented at a level appropriate to each student, presenting a simple one for new users, and evolving to the full functionality and interface as the user becomes more familiar with the system.

Other capabilities are also put forward: supporting flexible pedagogies, organizational or social structures, and access to historical archives, as additional information is collected over time. The ideal system is never identified, but encompasses a full set of flexible tools for the breadth of student abilities, and the depths to which they might wish to learn about the topics, an adaptive interface, a full suite of instructional tools for administering, grading, and monitoring, natural conceptualizations of the virtual spaces, authoring tools for reusability and rapid content creation are also key features. Table 13 collects the set of capabilities and features listed by the NSF, and elsewhere in this document, and indicates what the Virtual-U and WebCT systems currently support. Table 14 repeats the exercise for feature sets of the two systems, and includes items listed in previous chapters and sections.

Table 13: NSF Checklist, Against Virtual-U and WebCT

Yes	●
Partly	⊙
No	○

Feature or Capability	**Virtual-U**	**WebCT**
Support for social and organizational structures	●	●
Activity management and evaluation	○	●
Relate virtual to physical objects, spaces, and activities	●	○
Delineate virtual public, group, and private spaces	●	●
Represent & organize information collected over time	⊙	⊙
User-customizable interface for ability levels	⊙	○
Embedded assessment of user abilities, to prescribe activities	○	○
Flexible for depth of student interest / needs	●	●
Flexible for breadth of student abilities / needs	●	●
Re-usable materials, in different years or courses	●	●
Gather & analyze data on student learning	⊙	●
Gather & analyze data on software performance	●	●
Deliver content in multiple media	●	●
Adaptable interface as abilities improve over time	○	○
Advanced querying, for imprecise searches & reformulation	⊙	⊙
Assessment of individual contribution in group situations	○	○
Capturing workflow and performance in groups	○	○
Voice and gesture input modalities	○	○

Table 14: Features Checklist of Virtual-U and WebCT

Yes ●
Partly ⊙
No ○

Feature or Capability	**Virtual-U**	**WebCT**
Asynchronous communication	●	●
one to one	○	●
one to many	●	●
class-wide	●	●
project teams	●	●
Synchronous communication	○	●
Public spaces, for social interaction	●	●
Private spaces for group communication	●	●
Private spaces for individual work	●	●
Grades: Student access to grading information	●	●
Grading facility	●	●
On-line quizzes and exams	○	●
Flexible grading schemes (among students & teams)	○	○
Student uploads and downloads	○	●
Group brainstorming tools	○	○
Group meeting tools	○	○
Group voting tools	○	○
Instructor power to cancel posts	○	●
Teaching Assistant power to cancel posts	○	○
Student power to cancel posts	○	○
Instructor planning tools: physical arrangements	●	○
Instructor planning tools: topic scheduling & ordering	●	●
Simple to rearrange syllabus or order of topics	○	●
Tools for collaboration of instructors	⊙	○
Monitoring	⊙	●
Student page accesses	○	●
Student communications	⊙	●
Individual page accesses	○	●
Linking of external references	●	●
Uploading or archiving of course content	○	●
Message annotation and analysis tools	○	○
Content editor (HTML)	○	○

Conclusions

CPSC 319 is a course that groups undergraduate students into formal project teams of eight to ten people, for the analysis, design, and creation of a large software system. Artifacts produced by the teams during the project include text and image documents, design and specification descriptions, source code, and plans and results for black-box, system, and integration testing. The curriculum stresses group dynamics and team processes, which are assessed with the aid of project management deliverables such as a project proposal, meeting agendas, meeting minutes, and evidence of progress monitoring.

Support is provided for a collection of tools with which the teams may complete their tasks. The main ones include SUIT, TCL, Oracle, RCS, C, and C++. By having a computer account with the Department of Computer Science, students have access to e-mail, news readers, Web browsers, and other general-purpose tools that can be used for CPSC 319. These tools enable them to communicate with one another and the course instructor.

There exist some tools that are specifically geared for delivering educational content, communications, and activities in an on-line setting. These are integrated systems that provide, on a single software platform, much of the functionality necessary for an effective learning experience. Two systems in particular, the Virtual-U and WebCT, are examined in detail. The Virtual-U emphasizes the VGroups conferencing system, as a virtual place where students can interact, exchanging ideas and insights, and building upon one another. WebCT has a richer feature set than the Virtual-U, particularly in the area of course management. The tracking capabilities of WebCT are particularly noteworthy, enabling course instructors to determine which students are falling behind in the readings or in the discussions. Both the Virtual-U and WebCT are capable of collecting large amounts of text from the on-line discussions, but neither system provides the tools to annotate the messages, or to perform linguistic analysis. Neither system includes group facilitation features, for team brainstorming, voting, or working. While both emphasize student collaboration, neither WebCT nor the Virtual-U includes features that explicitly enable instructor collaboration. Otherwise, both systems provide a competent integrated platform for course delivery.

In studying the course during the 1995-96 and 1996-97 academic years, I examined the newsgroup archives (both years) and VGroups discussions (only 1996-97). I found that, because of the voluntary nature of newsgroup participation, it could not be guaranteed that all students would participate, either actively or passively. The change in course format, workload, and instructors between 1995-96 and 1996-97 contributed to a significant increase in the participation of the students who were active newsgroup posters. However, by creating an activity that required student participation in the on-line discussion, it would be possible to address the need for ensuring that all students emerge from the course with a minimum understanding of the key topics. Finally, based on a survey of students, there is an interest in tools that do not duplicate anything the students would be using elsewhere. This applied specifically to VGroups, which was viewed as being a Web version of the course newsgroup, and was rejected on the basis of this and its lack of a critical mass of participants.

The current offerings in on-line educational systems are integrated, and accessible from different platforms and different locations. They provide several management and learning tools, and are becoming less dependent on straight text, all laudable achievements. However, by omitting group meeting tools, and the capability to perform textual annotation and linguistic analysis, there is still room for improvement in the current systems.

References

1. Baram, Giora, and Munir Mandviwalla. “Use of computer conferencing in teaching systems analysis and design.” SIGCSE Bulletin 2 (1996): 37-39.

2. Bereiter, C., and M. Scardamalia. “Two models of classroom learning using a communal database.” In Instructional models in computer-based learning environments. Ed. S. Dijkstra. Berlin: Springer-Verlag, 1992.

3. Bowen, B., C. Bereiter, and M. Scardamalia. “Computer supported intentional learning environments.” In Thinkwork: Working, Learning, and Managing in a Computer-Interactive Society. Ed. F.V. Phillips. New York: Praeger, 1991.

4. Brooks, Frederick P. The Mythical Man-Month. Reading, MA:Addison-Wesley Publishing Company, 1982.

5. Brown, Marc H. “WebCard: Integrated and uniform access to mail, news, and the web.” Report 139A, DEC Systems Research Center,July 15, 1996. Also at ftp://ftp.digital.com/pub/DEC/SRC/research-reports/SRC-139a.html (18 June 1997)

6. Chapanis, Alphonse. “Interactive human communication.” Scientific American 3 (1975): 36-42.

7. Conklin, E. Jeffery. “Capturing organizational memory.” In Readings in Groupware and Computer-Supported Cooperative Work: Assisting Human-Human Collaboration. Ed. Ronald M. Baecker. California:Morgan & Kaufmann Publishers Inc, 1993. 561-565.

8. Conklin, Jeff, and Michael L. Begeman. “gIBIS: A hypertext tool for exploratory policy discussion.” ACM Transactions on Office Information Systems 4 (1988): 303-331.

9. Curry, Joanne. “TL-RN at a glance.” TL-RN Update 1 (1996): 2-3.

10. Dede, Chris. “Integrating learning and working.” NSF Educational Technology Workshop. 1995. http://www.cc.gatech.edu/gvu/edtech/nsfws/integrated.html (18 June 1997).

11. DeMarco, Tom, and Timothy Lister. Peopleware: Productive Projects and Teams. New York: Dorset House Publishing Company, 1987.

12. Edwards, W. Keith, and Elizabeth D. Mynatt. “Timewarp: techniques for autonomous collaboration.” In CHI 97: Proceedings of the Conference on Computer-Human Interaction, Atlanta: ACM Press, 1997. 218-225.

13. Eisenberg, M. B. and D. P. Ely. “Plugging into the Net.” ERIC Review 2 (1993): 2-10.

14. Eveland, J.D. and T.K. Bikson. “Work group structures and computer support: A field experiment.” ACM Transactions on Office Information Systems 4 (1988): 354-379.

15. Ferraro, Anne, Edwin Rogers, and Cheryl Geisler. “Team learning through computer supported collaborative design.” In Proceedings of the 1995 Computer-Supported Cooperative Learning Conference. October, 1995. Also at http://www-cscl95.indiana.edu/cscl95/ferraro.html (18 June 1997)

16. Fielding, R. and R. Lee. Using Computers in Qualitative Analysis. Thousand Oaks: Sage, 1991.

17. Finholt, Tom, and Lee S. Sproull. “Electronic groups at work.” In Readings in Groupware and Computer-Supported Cooperative Work: Assisting Human-Human Collaboration. Ed. Ronald M. Baecker. California:Morgan & Kaufmann Publishers Inc, 1993. 431-442.

18. Fischer, G. et al. “The role of critiquing in cooperative problem solving.” ACM Transactions on Information Systems 2 (1991): 123-151

19. Gay, Geri, and Marc Lentini. “Use of communication resources in a networked collaborative design environment.” Journal of Computer-Mediated Communication 1 (1996). Also at http://cwis.usc.edu/dept/annenberg/vol1/issue1/IMG_JCMC/ResourceUse.html (18 June 1997)

20. Goldberg, Murray. “Student participation and progress tracking for Web-based courses using WebCT.” In Proceedings of the Second International N.A. Web Conference, Fredericton, October 1996. Also at http://homebrew.cs.ubc.ca/webct/papers/naweb/index.html (18 June 1997)

21. Goldberg, Murray. “An update on WebCT (World Wide Web Course Tools) - A tool for the creation of sophisticated Web-based learning environments.” In Proceedings of NAUWeb ‘97. Arizona, June 1997. Available at http://homebrew.cs.ubc.ca/webct/papers/nauweb/full-paper.html (18 June 1997).

22. Goldberg, Murray. “Communication and collaboration tools in WebCT.” In Proceedings of the Conference - Enabling Network-Based Learning. Finland, May 1997B. Available at http://homebrew.cs.ubc.ca/webct/papers/enable/paper.html (18 June 1997).

23. Goldberg, Murray. “WebCT and first year computer science: Student reaction to and use of a Web-based resource in first year computer science.” In Proceedings of the ACM’s ITiCSE Conference on Integrating Technology into Computer Science Education, Sweden, June 1997C.

24. Goldberg, Murray, and Sasan Salari. “World Wide Web Course Tool: An environment for building WWW-based courses.” Computer Networks and ISDN Systems 28 (1996) 1219-1231. Available at http://homebrew.cs.ubc.ca/webct/papers/p29/index.html (18 June 1997).

25. Grudin, Jonathan. “Groupware and social dynamics: Eight challenges for developers.” In Readings in Human-Computer Interaction: Toward the Year 2000. (2nd ed) Eds. Ronald M. Baecker et al. San Francisco: Morgan Kaufmann Publishers, 1995.

26. Guzdial, Mark and Rick Weingarten. "Overview and summary: Research in the union of computer science and education." NSF Educational Technology Workshop.1995. http://www.cc.gatech.edu/gvu/edtech/nsfws/frontmatter.html (18 June 1997).

27. Guzdial, Mark, et al. "Collaborative support for learning in complex domains." In Proceedings of the 1995 Computer-Supported Cooperative Learning Conference, October, 1995.Also at http://www-cscl95.indiana.edu/cscl95/guzdial.html (18 June 1997)

28. Harasim, L. and B. Yung. Teaching and Learning on the Internet. Burnaby, BC: Department of Communication, Simon Fraser University, 1993.

29. Harasim, Linda, et al. Learning Networks. Cambridge:MIT Press, 1995.

30. Hawkins, Jan. "Supporting teachers in changing roles." NSF Educational Technology Workshop. 1995. http://www.cc.gatech.edu/gvu/edtech/ nsfws/teachers.html (18 June 1997).

31. Hiltz, Starr Roxanne. "Teaching in a Virtual Classroom(tm)." In ICCAI 95: Proceedings of the 1995 International Conference on Computer Assisted Instruction, Taiwan, 1995. Also at http://www.njit.edu/njIT/Directory/Centers/CCCC/Hiltz 1995/Papers/Teaching.html (18 June 1997)

32. Hiltz, Starr Roxanne. "Impacts of college-level courses via asynchronous learning networks: Focus on students." Sloan Conference on Asynchronous Learning Networks, Philadelphia, October 1995B.

33. Hmelo, Cindy E. et al. "Technology support for collaborative learning in a problem-based curriculum for sustainable technology." In Proceedings of the 1995 Computer-Supported Cooperative Learning Conference, October, 1995. Also at http://www-cscl95.indiana.edu/cscl95/hmelo.html (18 June 1997)

34. Hollander, Edwin P. and James W. Julian. "Studies in leader legitimacy, influence, and innovation." In Group Processes. Ed. Leonard Berkowitz. New York: Academic Press, 1978.

35. Horning, J. J. and D. B. Wortman. "Software Hut: A computer program engineering project in the form of a game." IEEE Transactions on Software Engineering 3 (1977): 325-330.

36. Hunter, Beverly. "Facilitating use of the network: Integrating school, home, industry, and community." NSF Educational Technology Workshop. 1995. Also at http://www.cc.gatech.edu/gvu/edtech/nsfws/netcomm.html (18 June 1997).

37. Hymel, Shelley, Beverly Zinck, and Elise Ditner. "Cooperation versus competition in the classroom." Exceptionality Education Canada 1 (1993): 103-128.

38. Johansen, Robert, and Chistine Bullen. “Thinking ahead: What to expect from teleconferencing.” Harvard Business Review 3 (1984): 4-10.

39. Johnson, David W., and Frank P. Johnson. Joining Together: Group Theory and Group Skills. New Jersey: Prentice-Hall Inc, 1975.

40. Kiesler, Sara, et al. “Social psychological aspects of computer-mediated communication.” American Psychologist 39 (1984): 1123-1134.

41. Lai, Kum-Yew, Thomas W. Malone, and Ken-Chiang Yu. “Object Lens: A ‘spreadsheet’ for cooperative work.” ACM Transactions on Office Information Systems 4 (1988): 332-353.

42. Levinson, P. “Media relations: Integrating computer telecommunications with educational media.” In Mindweave: Communication, Computers, and Distance Education. Ed. R. Mason and K. Kaye. Oxford: Pergamon Press, 1989.

43. MacKay, Wendy. “Diversity in the use of electronic mail: A preliminary inquiry.” ACM Transactions on Office Information Systems 4 (1988): 380ff.

44. Markus, M. L. “Finding a happy medium: Explaining negative effects of electronic communication on social life at work.” ACM Transactions on Information Systems 2 (1994): 119-149.

45. Mason, R. “An evaluation of CoSy on an Open University course.” In Mindweave: Communication, Computers, and Distance Education. Ed. R. Mason and K. Kaye. Oxford: Pergamon Press, 1989.

46. Mason, R. “Conferencing for mass distance education.” In Proceedings of the Third Guelph Symposium on Computer Conferencing. Guelph, ON: University of Guelph, 1990.

47. McConnell, Jeffrey J. “Active learning and its use in computer science.” SIGCSE Bulletin 3 (1996): 52-54.

48. McGrath, Joseph E. “Groups and human behaviour.” In Readings in Groupware and Computer-Supported Cooperative Work: Assisting Human-Human Collaboration. Ed. Ronald M. Baecker. California:Morgan & Kaufmann Publishers Inc, 1993. 113-115.

49. McGrath, Joseph E. “Time, Interaction, and Performance (TIP): A theory of groups.” In Readings in Groupware and Computer-Supported Cooperative Work: Assisting Human-Human Collaboration. Ed. Ronald M. Baecker. California:Morgan & Kaufmann Publishers Inc, 1993B. 116-129.

50. McGrath, Joseph E. “Methodology matters: Doing research in the behavioural and social sciences.” In Readings in Human-Computer Interaction: Toward the Year 2000. (2nd ed)

Eds. Ronald M. Baecker et al. San Francisco: Morgan Kaufmann Publishers, 1995. 152-169.

51. McGrenere, Joanna. Design: Educational Electronic Multi-Player Games, a Literature Review. Report TR-96-12, Imager/MAGIC, Department of Computer Science, University of British Columbia, June 1996. Also at http://www.cs.ubc.ca/nest/imager/tr/mcgrenere.96b.html (15 July 1997).

52. Mynatt, Elizabeth D. et al. "Design for network communities." In CHI 97: Proceedings of the Conference on Computer-Human Interaction, Atlanta: ACM Press, 1997. 210-217.

53. Ousterhout, John K. Tcl and the Tk Toolkit. Reading, MA: Addison-Wesley Publishing, 1994.

54. Paulsen, M. F. and Rekkedal, T. The Electric Cottage: Selected Articles from the EKKO Project. Norway. SEFU, Norwegian Centre for Distance Education, 1990.

55. Pausch, Randy, Matthew Conway, and Robert DeLine. "Lessons learned from SUIT, the Simple User Interface Toolkit." ACM Transactions on Information Systems 4 (1992): 320-344.

56. Price, George R. "The teaching machine." In Of Men and Machines. Ed. Arthur O. Lewis Jr. New York: E.P. Dutton & Co, 1963. 131-139.

57. Reisbeck, Chris. "Tools for authoring educational technology." NSF Educational Technology Workshop. 1995. http://www.cc.gatech.edu/gvu/edtech/nsfws/tools.html (18 June 1997).

58. Resnik, Paul. "Phone-based CSCW: Tools and trials." ACM Transactions on Information Systems 4 (1993): 401-424.

59. Richards, T.J. and Lyn Richards. "The NUD*IST qualitative data analysis system" Qualitative Sociology 4 (1991): 307-324.

60. Richards, T. J. and M. G. Richards. "A description of the NUD*IST system." In Using Computers in Qualitative Research. Eds. N. Fielding and R. Lee. Thousand Oaks: Sage. 187-198.

61. Richards, Tom, and Lyn Richards. "Using computers in qualitative analysis". In Handbook of Qualitative Analysis. Eds. N. Denzin and Y. Lincoln. Thousand Oaks: Sage, 1993.

62. Roseman, Mark, and Saul Greenberg. "Building real-time groupware with GroupKit, a groupware toolkit." ACM Transactions on Computer-Human Interaction 1 (1996): 66-106.

63. Sanderson, P.M. Exploratory Sequential Data Analysis: Software. Technical Report EPRL-94-01, Engineering Psychology Research Laboratory, University of Illinois, Urbana, Illinois, 1994.

64. Sanderson, P. M. et al. “MacSHAPA and the enterprise of Exploratory Sequential Data Analysis (ESDA).” International Journal of Human-Computer Studies 41 (1994): 633-68.

65. Sanderson, P. M. and C. Fisher. “Exploratory sequential data analysis: Foundations.” Human-Computer Interaction 9 (1994): 251-317.

66. Sanderson, P.M., M. McNeese, and B. Zaff. “Knowledge elicitation and observation in engineering psychology: MacSHAPA and COGENT.” Behavior Research Methods, Instruments, and Computers 26 (1994): 117-124.

67. Scardamalia, M. and C. Bereiter. “Computer-supported intentional learning environments.” In Design for Learning: Research-based Design of Technology for Learning. Ed. B. Bowen. Cupertino, CA: Apple Computers Inc, 1990. 5-14.

68. Scardamalia, M. and C. Bereiter. “Technologies for knowledge-building discourse.” Communications of the ACM 5 (1993): 37-41.

69. Scardamalia, M. and C. Bereiter. “Computer support for knowledge building communities.” Journal of the Learning Sciences 3 (1994): 265-283.

70. Scardamalia, M. et al. “Computer supported intentional learning environments.” Journal of Educational Computing Research, 5 (1989): 51-68.

71. Shneiderman, B. Designing the User Interface: Strategies for Effective Human-Computer Interaction (2nd Ed). Reading, MA: Addison-Wesley, 1992.

72. Slavin, R. E. “Cooperative learning.” Review of Educational Research 2 (1980): 315-342.

73. Sproull, Robert F. “A lesson in electronic mail.” In Readings in Groupware and Computer-Supported Cooperative Work: Assisting Human-Human Collaboration. Ed. Ronald M. Baecker. California:Morgan & Kaufmann Publishers Inc, 1993. 403-406.

74. Stoloway, Elliot, and Mark Guzdial. “Designing for learners.” NSF Educational Technology Workshop. 1995. http://www.cc.gatech.edu/gvu/edtech/nsfws/dfl.html (18 June 1997).

75. Swigger, Kathleen, Brazile, Robert, and Shin, Dongil. “Teaching computer science students how to work together.” In Proceedings of the 1995 Computer-Supported Cooperative Learning Conference. October, 1995. Also at http://www-cscl95.indiana.edu/cscl95/swigger.html

76. Terveen, Loren G. et al. "Building task-specific interfaces to high volume conversational data." In CHI 97: Proceedings of the Conference on Computer-Human Interaction, Atlanta: ACM Press, 1997. 226-233.

77. Turoff, Murray. "Designing a Virtual Classroom [TM]." 1995 International Conference on ComputerAssisted Instruction ICCAI'95, Taiwan: National Chiao Tung University Hsinchu, 1995.

78. Turoff, Murray. "Computer-Mediated Communication Requirements for Group Support." In Readings in Groupware and Computer-Supported Cooperative Work: Assisting Human-Human Collaboration. Ed. Ronald M. Baecker. California:Morgan & Kaufmann Publishers Inc, 1993. 407-417.

79. Weinberg, Gerald. The Psychology of Computer Programming. New York: Van Nostrand Reinhold Company, 1971.

80. Westrom, Marv, and Tom Pankratz. "Creating Collaborative Communities Online." In Proceedings of NAUWeb '97. Arizona, June 1997. Available at http://star.ucc.nau.edu:80/~nauweb/papers/westrom.html (7 July 1997).

81. Wick, Carolyn, et al. Report on the Evaluation of VGroups. unpublished report, University of British Columbia Department of Computer Science and Simon Fraser University School of Computing Science, April 10 1996.

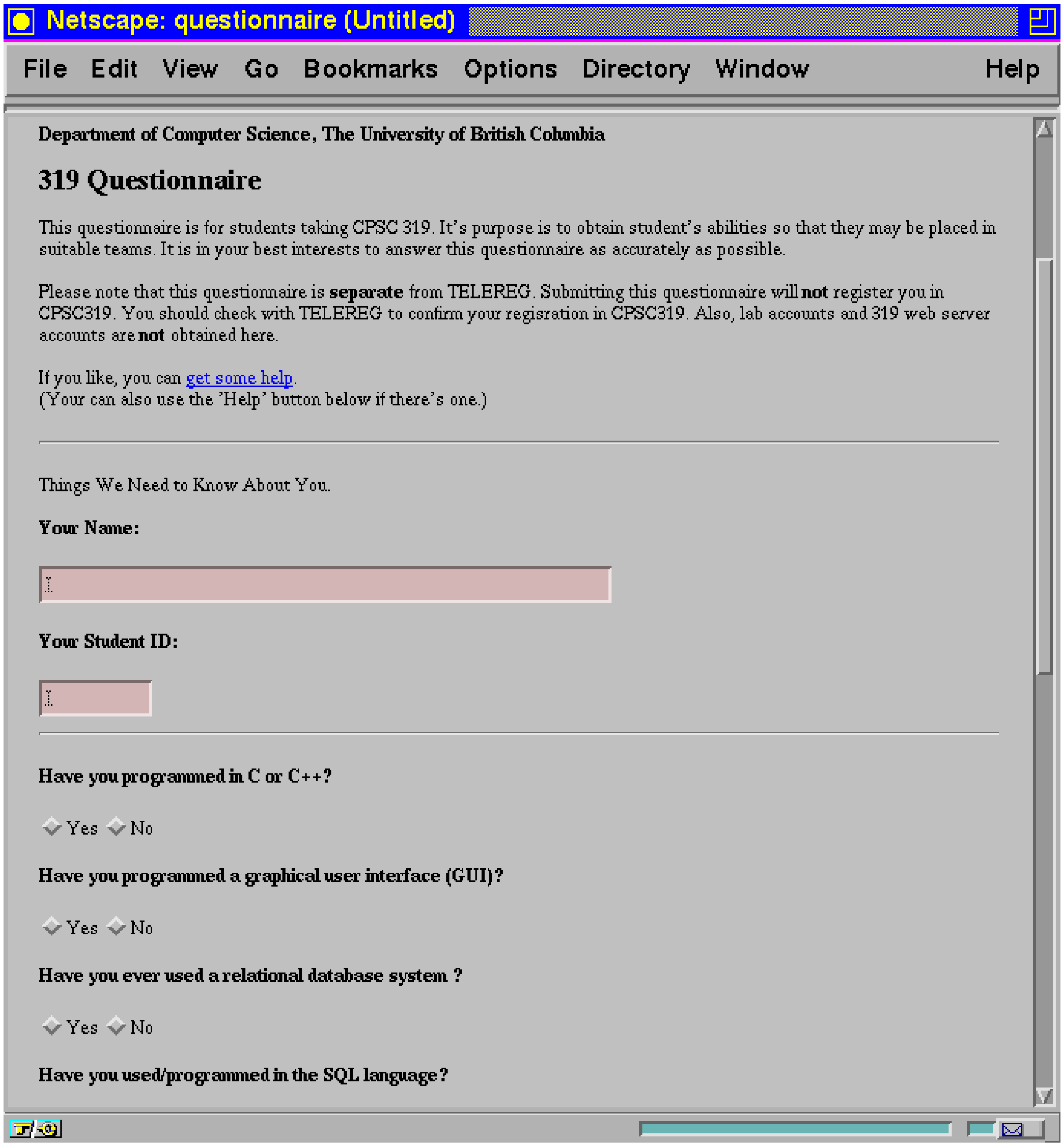
82. Wolfe, Richard. "Transcript Analysis." Lecture given at TeleLearning NCE Conference, Montreal , Quebec, November 1996.

83. Wortman, David. "Software Projects in an Academic Environment." IEEE Transactions on Software Engineering 11 (1987) 1176-1181.

84. Yakemovic, K.C. Burgess. "Report on a Development Project Use of an Issue-Based Information System." In CSCW 90: Proceedings of the Conference on Computer-Supported Cooperative Work. Los Angeles, 1990. 105-118.

85. Yerion, Kathie A. and Rinehart, Jane A. "Guidelines for Collaborative Learning in Computer Science." SIGCSE Bulletin 4 (1995): 29-34.

Appendix A -- Preliminary Skills Questionnaire


Netscape: questionnaire (Untitled)
File Edit View Go Bookmarks Options Directory Window Help

Department of Computer Science, The University of British Columbia

319 Questionnaire

This questionnaire is for students taking CPSC 319. It's purpose is to obtain student's abilities so that they may be placed in suitable teams. It is in your best interests to answer this questionnaire as accurately as possible.

Please note that this questionnaire is **separate** from TELEREG. Submitting this questionnaire will **not** register you in CPSC319. You should check with TELEREG to confirm your regisration in CPSC319. Also, lab accounts and 319 web server accounts are **not** obtained here.

If you like, you can get some help.
(Your can also use the 'Help' button below if there's one.)

Things We Need to Know About You.

Your Name:

Your Student ID:

Have you programmed in C or C++?

Yes No

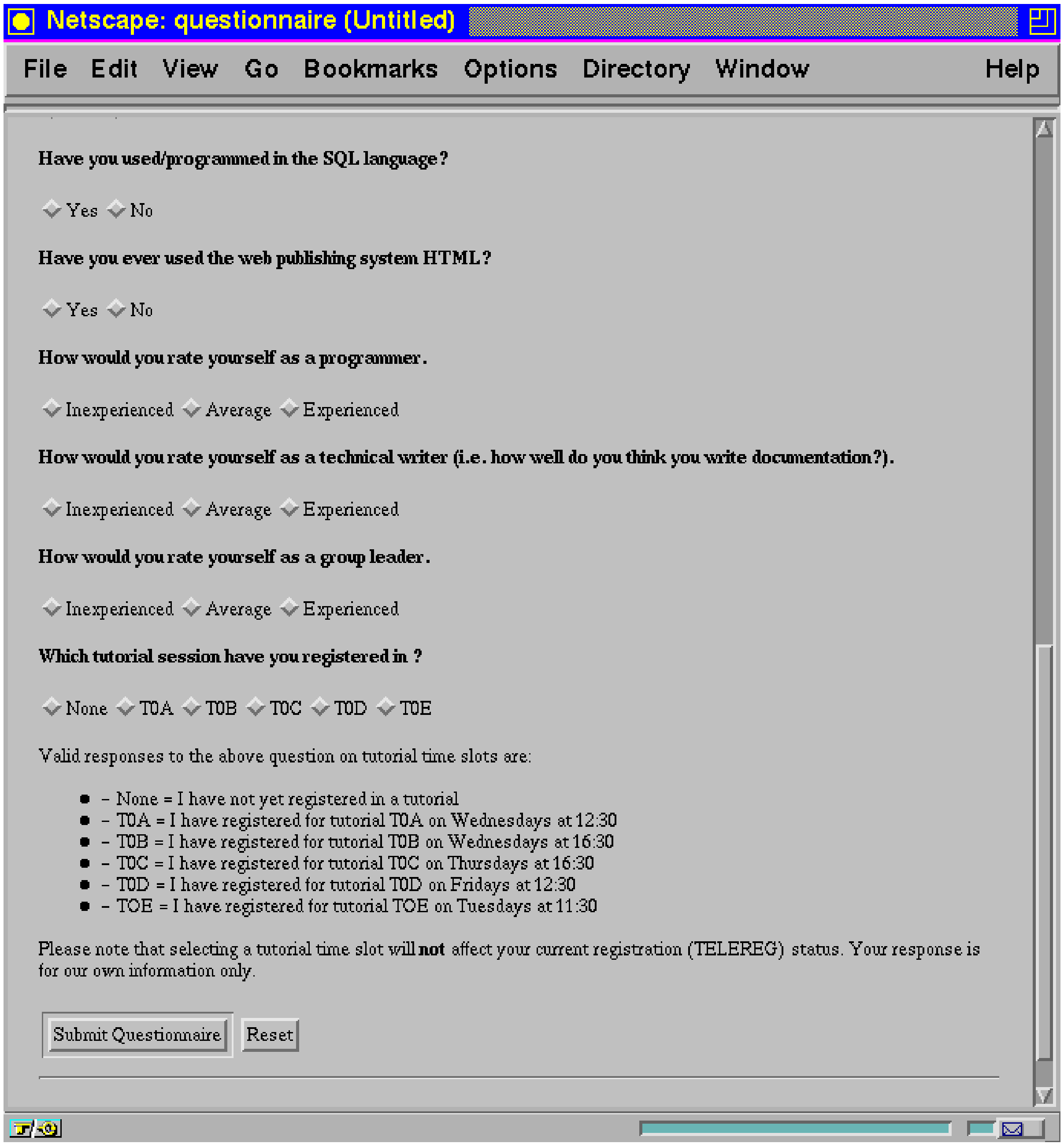
Have you programmed a graphical user interface (GUI)?

Yes No

Have you ever used a relational database system ?

Yes No

Have you used/programmed in the SQL language?


Netscape: questionnaire (Untitled)
File Edit View Go Bookmarks Options Directory Window Help

Have you used/programmed in the SQL language?

◇ Yes ◇ No

Have you ever used the web publishing system HTML?

◇ Yes ◇ No

How would you rate yourself as a programmer.

◇ Inexperienced ◇ Average ◇ Experienced

How would you rate yourself as a technical writer (i.e. how well do you think you write documentation?).

◇ Inexperienced ◇ Average ◇ Experienced

How would you rate yourself as a group leader.

◇ Inexperienced ◇ Average ◇ Experienced

Which tutorial session have you registered in ?

◇ None ◇ T0A ◇ T0B ◇ T0C ◇ T0D ◇ T0E

Valid responses to the above question on tutorial time slots are:

- – None = I have not yet registered in a tutorial
- – T0A = I have registered for tutorial T0A on Wednesdays at 12:30
- – T0B = I have registered for tutorial T0B on Wednesdays at 16:30
- – T0C = I have registered for tutorial T0C on Thursdays at 16:30
- – T0D = I have registered for tutorial T0D on Fridays at 12:30
- – T0E = I have registered for tutorial T0E on Tuesdays at 11:30

Please note that selecting a tutorial time slot will **not** affect your current registration (TELEREG) status. Your response is for our own information only.

Appendix B -- Student Deliverable Samples

Deliverable: Sample Thread

DeleteFlight Thread

Overview

This thread deletes a flight from the system. Only a connection belonging to a system administrator can use this function. An error occurs if the flight does not exist or if the flight described has any existing passengers with reservations (a flight cannot be deleted from the system if an existing passenger refers to the flight).

Stimulus

1) The ARS shall satisfy the requirements of this thread upon the receipt of a **[delete flight information]** of the form:

   a. **<flight_integer>**

Responses

1) The ARS shall return a **[delete flight acceptance]** as follows:

   a. **<delete_flight_accept_message>** to the user
   b. **<flight_record>** is deleted from the RMS.

2) The ARS shall return a **[delete flight privilege rejection]** as follows:

   a. **<user_not_admin_message>** to the user

3) The ARS shall return a **[delete flight rejection]** as follows:

   a. **<bad_flight_id_message>** to the user

4) The ARS shall return a **[delete flight active rejection]** as follows:

   a. **<delete_flight_is_active_message>** to the user

5) The ARS shall return a **[delete flight incomplete rejection]** as follows:

   a. **<delete_flight_incomplete_message>** to the user
   b. Prompt for **[delete flight information]**

Requirements

1) Upon receipt of a **[delete flight information]** the ARS shall return **[delete flight acceptance]** if all of the following conditions are true:

a. **<flight_integer>** is known to the RMS
b. **<flight_integer>** does not have any reservations on it
c. The user who initiated the thread has administrator privileges.

2) Upon receipt of a **[delete flight information]** the ARS shall return **[delete flight privilege rejection]** if all of the following conditions are true:

a. The user who initiated this thread does not have administrator privileges.

3) Upon receipt of a **[delete flight information]** the ARS shall return **[delete flight rejection]** if all of the following conditions are true:

a. **<flight_integer>** is not known to the RMS.
b. The user who initiated this thread has administrator privileges.

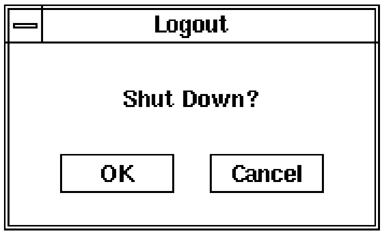
4) Upon receipt of a **[delete flight information]** the ARS shall return **[delete flight active rejection]** if all of the following conditions are true:

a. **<flight_integer>** does have reservations associated with it.
b. The user who initiated this thread has administrator privileges

5) Upon receipt of a **[delete flight information]** the ARS shall return **[delete flight incomplete rejection]** if all of the following conditions are true:

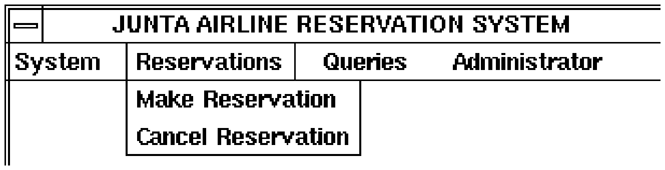
a. **<flight_integer>** is not submitted by the user
b. The user who initiated this thread has administrator privileges

Deliverable: Sample Layout Design

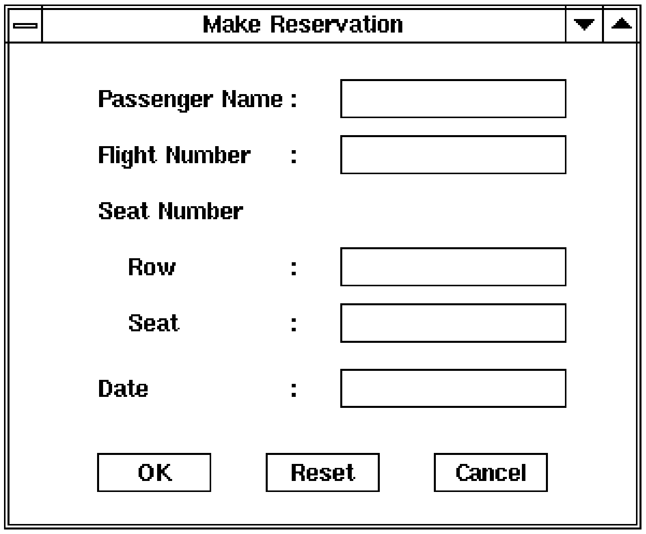


User can either cancel this operation or proceed with logout.

Reservations Menu

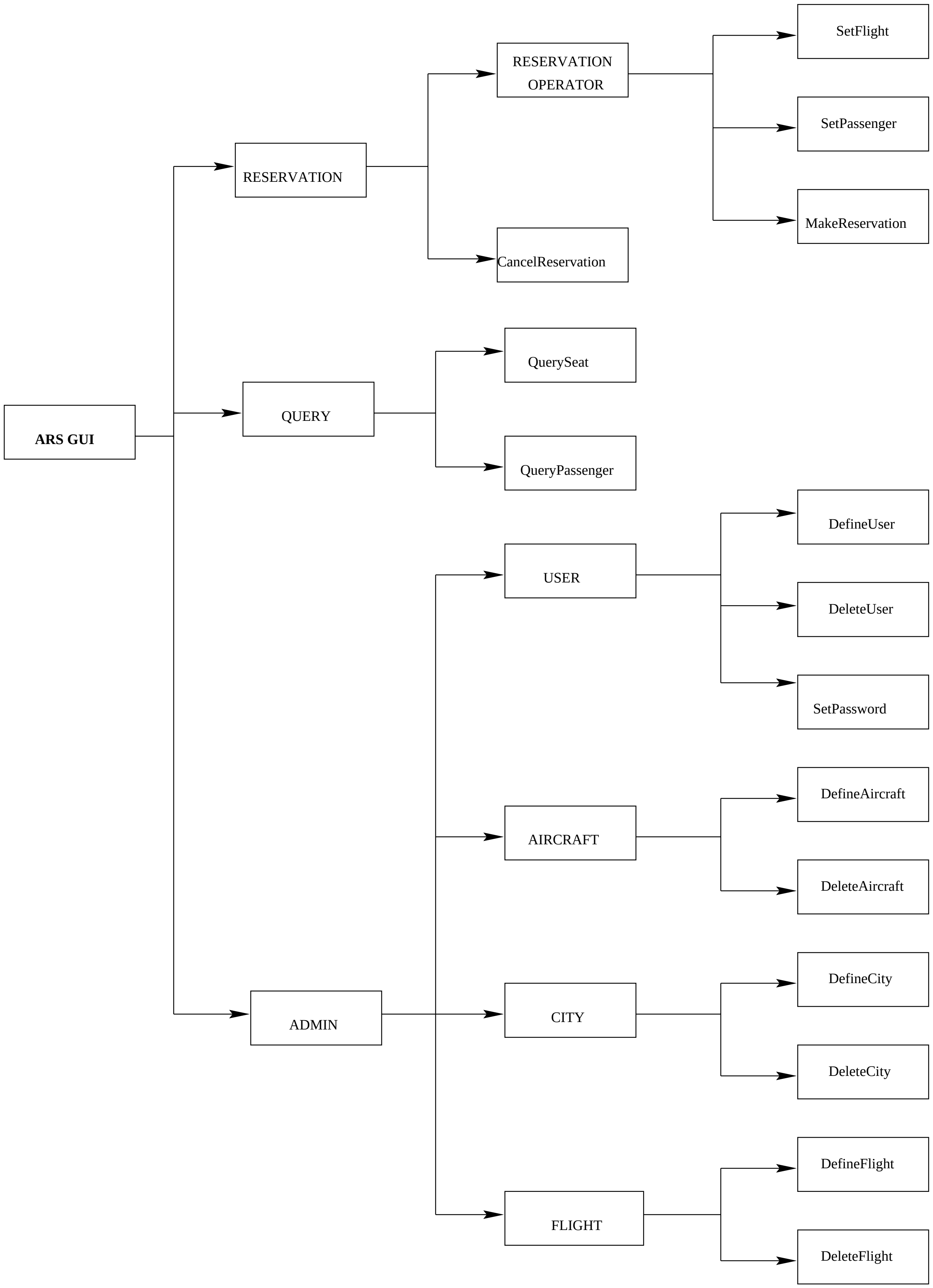


Make Reservation



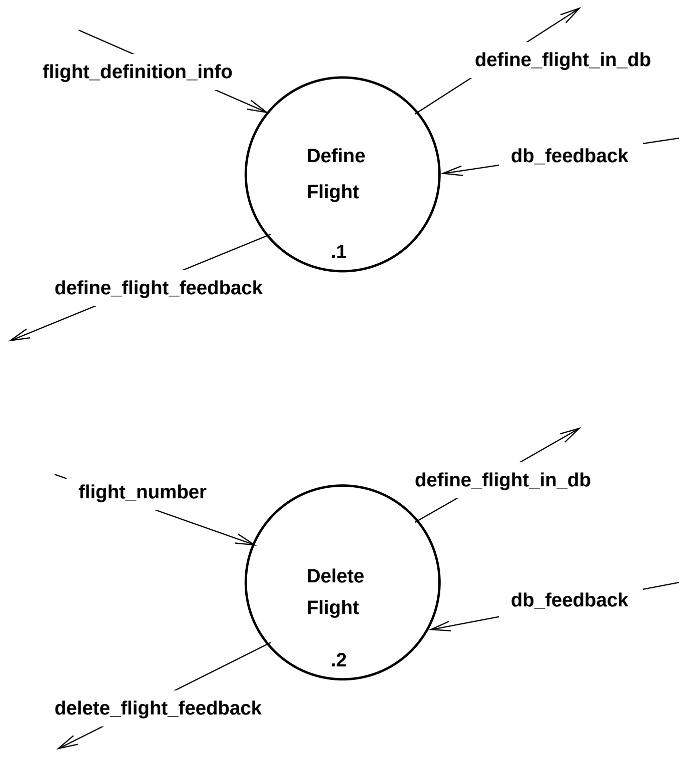
If any field contains incorrect, unavailable, or unresolvable information, the corresponding error message is brought up.
Successful execution lets window disappear.

Deliverable: Sample Is-Composed-Of Diagram



4.2 DFD 2.2: Flight Functions

2.2;1
Flight Functions



4.2.1 P-Spec 2.2.1: Define Flight

```
TITLE:
Define Flight

INPUT/OUTPUT:
db_feedback : data_in
flight_definition_info : data_in
define_flight_in_db : data_out
define_flight_feedback : data_out

BODY:
Package the flight_definition_info data into a format appropriate for the
database.

Send a command and the packaged data (define_flight_in_db) to the database to
```

```
add the new flight record to the database.

Receive feedback from the database (db_feedback) and transform it into
appropriate feedback (define_flight_feedback) for the GUI.
```

4.1.2 P-Spec 2.2.2: Delete Flight

```
TITLE:
Delete Flight

INPUT/OUTPUT:
flight_number : data_in
db_feedback : data_in
delete_flight_in_db : data_out
delete_flight_feedback : data_out

BODY:
Package the flight_number data into a format appropriate for the database.

Send a command and the packaged data (delete_flight_in_db) to delete the flight
record from the database.

Receive feedback from the database (db_feedback) and transform it into
appropriate feedback (delete_flight_feedback) for the GUI.
```

3.7 Module: Users

3.7.1 Abstract Specification

The Users module is responsible for adding and deleting user id's, and setting passwords for users of the system.

3.7.2 Externally Visible Attributes

Constants: none

Data Types: none

Functions:

- SetPassword(connectID: connectType, userid: integer, Password: char*) returns integer
- DefineUser(connectID: connectType, userid: integer) returns integer
- DeleteUser(connectID: connectType, userid: integer) returns integer

3.7.3 Externally Visible Behaviour

SetPassword is responsible for setting the password for a specified user. Note that only the connection of a supervisor can change the password of another userid.

The DefineUser function allows for the definition of a new user, with a given authority class. Note that only someone with a supervisor connection can use this function.

The DeleteUser function allows the given userid to be deleted from the system. Only a supervisor can delete another user from the system, and that user's connections are closed before deleting him/her from the database.

3.7.4 Imports

- DbAddUser(UserId: integer, Class : integer) from the database interaction module.
- DbDeleteUser(UserId: integer) from the database interaction module.
- DbQueryUser(UserId: integer) from the database interaction module.
- DbQuerySupervisor(UserId: integer) from the database interaction module.
- DbSetPassword(UserId: integer, Password: *char) from the database interaction module.
- Connection_IsValid(connectionType) from the connection module.
- Connection_CloseUserConnections(userType) from the connection module.

Appendix C -- Communications Questionnaire

Department of Computer Science, The University of British Columbia
319 Communication Questionnaire

This questionnaire is for students taking CPSC 319. Its purpose is to obtain insights into what online communication tools students actually used, how often, and what other sorts of tools would prove to be beneficial, useful, or just plain interesting. Please answer this questionnaire as accurately as possible.

Topic: **Frequency of Usage**

Please check the answer that **most closely describes** how often you did the following activities:

Statement: **How Often did you...**	**Never**	**Once a Month**	**Once a Week**	**Once a Day**	**Many times a day**	**Always**
1.perform 319 electronic communication on an off-campus computer						
2. check your e-mail account(s) for messages						▨
3. check the course newsgroup						▨
4. post to the course newsgroup						▨
5. access the 319 web server						
6. access VGroups in the Virtual University						▨
7. post in a VGroups conference						▨
8. use a Talk or Chat system for 319						

Topic: **Electronic Mail**

1. What e-mail package do you use most often?

____ ean or Xean	____ zmail	____ pine
____ eudora	____ pegasus	____ Netscape Mail
____ CC:Mail	____ MS-Mail	____ Other

2. How many 319-related e-mail messages do you SEND in an average week?

____ None	____ Between 1 and 5
____ Between 6 and 10	____ More than 10

3. When you compose a 319-related email message, how long is it, usually?
____ No text, just a subject line ____ Between 1 and 3 lines of text
____ Between 4 lines and a screenful of text ____ More than a screenful of text

4. Here is a list of possible purposes for sending electronic mail to your 319 team. Check all that apply to messages you have sent to one or more members of your team in the last month:

____ Announce a meeting	____ Announce a decision	____ Ask a task-related question
____ Ask administration question	____ Announce task completion	____ Ask for help from teammates
____ Complain about the course	____ Tell a joke	____ Complain about a teammate
____ Reply to task-related question	____ Relate some facts	____ Reply to an admin question
____ Discuss another course	____ State your opinion	____ Schedule non-319 meeting

5. How many 319-related e-mail messages do you RECEIVE in an average week?

____ None	____ Between 1 and 5
____ Between 6 and 10	____ More than 10

6. When you RECEIVE a 319-related email message, how long is it, usually?

____ No text, just a subject line	____ Between 1 and 3 lines of text
____ Between 4 lines and a screenful of text	____ More than a screenful of text

Topic: **The Course Newsgroup Ubc.courses.cpsc.319, and the course Web Server**

1. How many articles on the newsgroup do you read?

____ I Never read any articles	____ I only read those posted by Peter or the TAs
____ I read about 1 out of every 3 articles	____ I read about 2 out of every 3 articles
____ I read every article	

2. From memory, please check all the features THAT ARE available on the CPSC 319 web server

____ Instructor's office hours	____ List of members of every team
____ Links to each group's home page	____ Form to automate self-evaluation
____ Weekly lecture notes	____ Optional online quizzes
____ Oracle Tutorial	____ Tcl/TK Tutorial

Topic: **Talk or Chat Systems**

1. If you used such a system for 319, check ALL reasons that apply to WHY you used it:

____ To ask a question	____ To coordinate activity
____ To report your status	____ To report results
____ To save walking	____ To talk about non-319 topics with a team member
____ Other reasons	

Topic: **VGroups and the Virtual University**

VGroups was a conferencing system we provided, to give teams another tool to help them with their online communications. Most people stopped using it after Christmas. We are interested in your feedback:

1. Please check ALL reasons that explain specifically why you chose not to use VGroups:

____ Never heard of it	____ Chance the interface might change
____ It sounded too hard to learn	____ It duplicated existing comm. tools I use
____ It was not a UBC product	____ There were researchers doing studies on it
____ No reason!	____ It was one more source to check
____ Nobody else used it	____ It was too slow

____ It was only for communication, no other course material was there

____ Other reasons

2. If and When you posted, who was your intended audience?

____ I never posted any messages	____ Always an individual
____ usually an individual	____ Both Individuals and the whole team
____ usually the whole team	____ Always the whole team

3. If you have any comments, thoughts or stories about VGroups, please submit them on the back of this questionnaire, or email them to spage@cs.ubc.ca

Topic: **Express your Attitude**
Check number 1 for Strongly Disagree, 2 for Disagree, 3 for Neutral, 4 for Agree, or 5 for Strongly Agree

Question	**1**	**2**	**3**	**4**	**5**
1. “Most of my E-mail messages are sent to 1,2,or 3 team members.”					
2. “Most of my E-Mail messages are sent to my whole team.”					
3. “Face-to-face talk has a harsher 'tone of voice' than e-mail”					
4. The newsgroup provides course information in a timely fashion.					
5. “The newsgroup has been good for answering my questions.”					
6. The newsgroup has been useful for discussing course-related issues.					
7. The newsgroup is dominated in a bad way by a small number of students.					
8. Peter and the TAs listen to us when we raise a problem in the 319 newsgroup.					
9. The VGroups feature that allows each team to have their own message space or archive (like a bulletin board or newsgroup) is useful for team communication.					
10. The VGroups feature that allows users to create sub-conferences for different topics is useful for team communication.					
11. The VGroups feature that allows access to be granted or revoked for each user and each conference is useful for team communication.					

If you used a **talk** system or a **chat** system in 319, please answer the following 3 questions using the above ratings. If you did not use such a system, please leave these questions blank.

12. Talk or Chat systems saved time in my communications.					
13. Talk or Chat systems made it easy to contact someone.					
14. Talk or Chat systems helped me convey messages of immediate importance.					

Topic: **Other Useful Features of OnLine Communication**
This section describes some features that are available in the above and other systems that provide on-line communication. Please answer with how useful you find these features.

Please check the box that matches your attitude for the following aspects of communication:
1 for Very UNimportant, 2 for A Little UNimportant, 3 for Undecided, 4 for A Little Important, 5 for Very Important, 6 for NO Opinion

Other Potential Communication Features	**1.**	**2.**	**3.**	**4.**	**5.**	**6.**
1. Team asynchronous communication tools (newsgroups or mailing lists just for your team, that don’t require the reader to be online when you send the message)						
2. Team synchronous communication tools (Talk or Chat tools just for your team’s use; synchronous means ALL participants must be online to be able to use the tool)						
3. Pre-structured message templates for memos, bug reports, announcements, etc...						
4. All course-related material and tools in one integrated place						
5. Ability to include images in my messages to team members						
6. Ability to work in the same room as a teammate doing a similar or related task						
7. Ability to link to web sites, messages, and resources, within the body of my message						
8. Ability to separate the 319-related messages from those messages not related directly to the course						
9. Ability to send e-mail from within the integrated online course system						
10. Ability to use documentation tools from within an integrated online course system.						

Send any questions or additional comments about this questionnaire to spage@cs.ubc.ca

Appendix D -- On-Line Discussion

The following is an excerpt from an important exchange in the CPSC 319 newsgroup during the 1996-97 academic year. It has been edited for brevity, with some of the repetitive or peripheral posts omitted. It illustrates the emphasis of some students on their grades, while others used the active and collaborative nature of the newsgroup to come to terms with a non-trivial issue. The conversation is taken verbatim from the newsgroup; all grammatical and spelling errors are [sic]. My annotations are in italics in curly braces.

Instructor:
I want to clarify CSAir's requirements for allowing multiple connections and multiple users.
. . . It can be seen from code.h that you should be allowed more than one connection at a time (that's the reason we pass a connection ID to every function). However, the [sample] program only allows one connection at a time, so this requirement was perhaps a little vague.
. . . Also, a user is allowed to be logged on more than once (on more than one connection). This hasn't been explicitly stated in any documentation, but the existing RMS allows it to happen, and if you didn't know about it, you do now.
{ *These requirements were stated explicitly during lecture in the second week of the course, and not mentioned again. An ambiguous few lines in the code.h document made reference to them* }

Student 1:
Requirement is not important. I would say it is a bad requirement. Most security concious apps only allow one login. Ever tried logging into more than one x-term?
. . . Yes, I'm upset. Why didn't you say anything when we handed in our second deliverable (Module implementation guide)? Don't tell me you couldn't see we would have a problem after one look at our dbSchema! Let me assure you, something like this would NOT be tolerated in the real world. If the customer wants something changed, first there is a feasibility analysis, then a timeline analysis. If the change will not fit in the current milestones, we tell the customer they will have to wait for the next release or respin. . . . I will rework the design if I have to, but not without a fight. Are you trying to mess up our precariously balanced schedule on purpose?
{*This begins two common themes: 1- is it a change or a missed requirement (who is at fault?) and would it happen this way in the "real world" or not (what arguments can be made against it?)* }

Student 2:
I second that motion. If we are to develop the ARS modeled according to the real world (with all the pitfalls and such), then we should receive appropriate compensation for our additional time and effort beyond what was agreed upon our original contract. Thus, I feel that this should be an *optional* requirement. eg. We get paid an extra $5000 for this requirement. (which translates to bonus marks) Otherwise, we should get marked as orginally.
{*I will omit most other references to the grades portion of the issue. This picks up the 'change' theme from the previous poster*}

Instructor:
After negotiations with various people, I've decided to re-express the requirement about connections and users in the RMS. It was never my intention that this requirement should be sprung on you to make life miserable, but instead it just turned out that it hadn't been clearly stated and some teams had taken a different approach.
CSAir's requirements will stay as they are (that is, multiple connections and multiple users), and if its not too late, then you're encouraged to follow those requirements.
. . . If you choose not to conform, then you must have a good explanation of why you decided not to, and a brief explanation of how you would go about changing your design in the future.

Having a surprise change to your plans is a good thing to have to deal with. In the real world, lots of these surprises are going to happen.
{*The instructor gives the students some good news: the requirement is as it always was, from Day 1, but for the sake of everyone's sanity, it will be turned into an optional part, with justification provided for its omission. Note a defence based on the "real world"*}

Student 3:
Hi, all! Interesting turn of events but I'm unclear as to a few points.
. . . Secondly, C [for those that are using C] is not a concurrency language. Our team actually thought about the problem of connection id's, etc. but through it out in a sec after realizing C's lack of concurrency.
. . . I suppose RELEASE could be added to every function and we hope that Oracle can handle concurrency.
{ *A technical debate revives the issue. This student is not interested in the grades debate, but rather in the real issue, of how a single program and database can permit multiple instances and logon connections.* }

Student 1:
Peter's suggestions would actually make life easier if one wasn't finished design and halfway through implementation. With his two suggestions, there really is no concurrency concern. It just happens as a result. Idea is: I can make a connection that doesn't matter on anybody else (connections). Multiple connections involve multiple instances of the RMS. The only thing shared is ORACLE. So one instance of the RMS can only handle one user, but you can have as many instances of it as you want.
{ *An important realization: knowing the correct requirements from the beginning would have reduced the number of technical difficulties they have faced. The debate begins about just what part should run multiple instances* }

Student 3:
I don't get this. If true, then this problem is solved, but if let's say you connect to it and I do, connection id's have to be created for both simultaneous use of the program. If your side gives you a connection id of "1", how will running the program for me know NOT to give me "1"? . . . if we're just talking about you logging in on computer X and leaving it after logout at the login screen and I sit down, then yes, the data structure would still be there and it could give me another connection id; but that's not what we're talking about, are we?
. . . Okay, the code is shared, but even if [it] assigned a connection id to users, it couldn't do it so that two simultaneous connections would have unique id's...
{ *Theories begin to form. At this point, Student 1 thinks it is solved, Student 3 remains unconvinced* }

Student 1:
Yes. Problem. Our team has a simple solution: User ids are stored in ORACLE, they are unique, and they are integers. Why not make your userid your connection id?
{ *Student 1 plays what he/she thinks is the ace card: userid = connection id. It is a reasonable assumption, for multiple instances, but does not account for multiple sessions by the same user* }

Instructor:
Interesting design, but could you imagine this working in real life? What if Canadian Airlines had 10000 travels agents using 10000 UIs connecting to 10000 copies of the RMS accessing one database. Each UI will be on a different (travel agent) computer, but the RMS must be on Canadian's own system (for security and database access reasons). Does it make sense to have 10000 RMS's on one machine, or one RMS (or perhaps a small cluster of them like with busy web servers) that can handle 10000 connections?
{ *At this point, the instructor intervenes, to offer a new twist to the issue* }

Student 1:
I would envision each travel agent with their own workstation with a copy of the RMS and GUI. When they start up a session, it just connects to the central database (ORACLE) Why wouldn't that work? I don't see any major security problems. It sure is a LOT easier to implement. :)
{ *Student 1 begins to struggle; maybe the earlier solution is not so strong after all...?* }

Instructor:

Don’t forget why the RMS exists in the first place. Apart from providing a user friendly interface to the database, it also restricts operations to authorized users. If all users ran their own RMS, then they’d would need almost complete access to the raw database, and there would be nothing to stop users from hacking the software, or writing their own RMS that would allow them to do illegal operations.
{ *The theme of system security is revisited* }

Student 1:

I see your point. Perhaps one could implement some kind of device for ORACLE that would allow only authentic RMSs to connect? (Augh.. Headache.) Since when are travel agents smart enough to hack a RMS? ;)
{ *The student is stumped, but by now is also very intrigued about the theoretical problem; marks have not been mentioned for several days* }

Instructor:

One of the concepts that is missing here was something that I covered in one of the first lectures, but haven’t really discussed since. The whole idea is that there should be one RMS (and database) and multiple UI’s connection to it. In previous years they managed to implement this feature, but I took it out due to the extra complexity.

Here’s a rough diagram:
```
 UI -\
      \
 UI --- RMS - DB
      /
 UI -/
```

The whole point of the connection ID is that each UI will call OpenConnection and receive a unique connection ID. For all the following transactions, the RMS uses the connection ID to work out who the request is from.

The RMS must deal with concurrency at the level of connections. You don’t need to worry about threads/semaphores since the RMS will ensure that the each request is handled sequentially. That is, if any procedure call (e.g. DefineFlight) is made by the first UI, then any other call by any other UI will have to wait until the first call is complete.

As far as your coding is concerned, you just need to write separate procedures, but pay close attention to the connection ID that is passed in. The connection ID refers you to that connection’s private data such as the user, the current flight and current passenger etc.

Oracle concurrency will only be useful if there is more than one program that is directly accessing the database. In our case, there is only one RMS per database. You could have more than one if you wanted, but that would complicate things.
{ *After a couple students gave the problem their best effort, the instructor reveals most of the solution. This is the theoretical solution, but implementing it requires some technical acumen as well. The specifics of method of implementation have been omitted by the instructor.* }

Student 3:

I see. That makes it clearer, but how would we do this? Maybe I’m still missing something, but when we use the makefile to create the executable "ars" isn’t the UI linked in with the RMS into one executable. To have let’s say three copies running as you’ve shown there, they’re 3 pairs of UI-RMS programs?
. . . What I could see is probably creating a table called "connectionid" that has one column called "id" that has its values added to and removed "on the fly" while the program is running, there-by dumping this problem on to good’ol Oracle. I suppose that would work. Was that what you had in mind? . . . And maybe keep a "count" starting at 0 and add one for each new user. And decrement it as each one logs off. And when it reaches 0 again, call a function called "DeInitializeRMS" in CloseConnection?

Thanks a lot! Your post was very helpful...but I'm wondering how easy it is to change our implmentation...
{ *This student identifies the remaining part of the theoretical solution, omitted by the instructor in the previous posting, namely how Oracle might handle the multiple sessions of the same agent. The student also returns to the implementation issues.* }

Instructor:
Yes, exactly. The one part of the system that was used in previous years, but I left out due to complexity, was called "The Multiplexor". It was a networking program that would allow multiple UIs to be connected to a single RMS.
{ *The technical details are then filled in, and the problem is solved at this point* }

Instructor:
If you refer to my previous posting you aren't required to support multiple connections, but you should justify why you haven't done it, and describe how you'd go about doing it if you wanted to (yes, there's heaps of flexibility and freedom in this course :-).
{ *Some more postings return to the issue of whether or not students REALLY need to implement this. The instructor assures them that he has already addressed this* }

Student 4:
As much as you want it to be, this is not a real life situation. I will be happy to re-design my software if the customer is willing to pay me more and claim the responsibities of breach of contract. I am pretty sure that there will be some kind of written agreements that stated any changes to the contract should be announced in advance. Another thing is, if we have to re-design my program, we are going to spend more time for sure. Why should I spend extra hours on a 3-credit course risking losing grades on my other courses worthy of 15 credits?
{ *A new student to the debate revives the change-vs-missed-requirement theme* }

Student 3:
I hate to break it to you, but it WAS specified and it ISN'T a change to the design. It wasn't mentioned to us in public, and perhaps that was the confusion, but on code.h, the connection id clearly states that it is a "unique" id. If you project is doing anything else, then [like our's], you probably ignored that comment beside the variable.

As much as you'd like to talk about "breach of contract" in the real world, this was clearly stated in code.h and if this WAS the real world, then our final proposal's wrong since this wasn't specified in it and would have never been accepted. You're talking about the real world, but in the real world, there wouldn't be a set deadline of late November for the proposal. We would have had to continue submitting it until the employer is happy with it.

Don't insult CS Air / TA's / Peter about real world. Face it...they know it isn't, cuz if they did, I'm sure they wouldn't have accepted over half of our proposals!
{ *One of the students who participated throughout the entire episode defends the course staff, saying that it was genuinely a missed requirement* }

Student 5:
If you mean:
typedef int connectType; /* unique integer ID for connection */
then I am curious as to how you can imply from that that a single user can have multiple connections? I checked OpenConnection too and it doesn't say anything either way as to whether or not you can or cannot have multiple connections for a single userID. The real problem is ambiguity in natural language writing, and it has always been my understanding that that is why we have the analysis stage.
{ *Another new student takes issue with the clarity of the requirement* }

Student 3:

As you pointed out, it really depends on how you interpret it. But, yes, this is the line that I meant. But, unfortunately, despite the size of code.h, this problem arose just by overlooking one variable or one line in the file.

In any case, who to blame is also hard to say. On one hand, you say the customer is since they are suppose to check our proposal over. On the other hand, if there was something ambigious, it is our responsibility to clarify it with the customer.

{ *At this point, the thread concludes. Student 3, and probably Student 1 have clearly grappled with the issues and gained a strong understanding. Unfortunately, there are also students who wish to deny any blame, or make sure the grades will not be adversely affected* }