Design and Implementation of a Smart Client for the Next Generation Mobile Classroom

by

Raj S. Dandage

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degrees of
Bachelor of Science in Computer Science and Engineering
and Masters of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

January 25, 2003

Copyright 2003 Raj S. Dandage. All rights reserved.

The author hereby grants to M.I.T. permission to reproduce and distribute publicly paper and electronic copies of this thesis and to grant others the right to do so.

Author

Department of Electrical Engineering and Computer Science

Certified By

Nishikant Sonwalkar
Thesis Supervisor

Accepted By

Arthur C. Smith
Chairman, Department Committee on Graduate Theses
Design and Implementation of a Smart Client for the Next Generation Mobile Classroom

by

Raj S. Dandage

Submitted to the Department of Electrical Engineering and Computer Science in partial fulfillment of the requirements for the degrees of Bachelor of Science in Computer Science and Engineering and Masters of Engineering in Electrical Engineering and Computer Science

Abstract
The Next Generation Mobile Classroom is a system of handheld PCs designed to promote interaction in large lecture halls. The system was designed to use MIT's existing wireless computing infrastructure and used in Fall 2002 in an MIT class. It consists of a server/web application, and a smart client that runs on handheld PCs and facilitates elaborate forms of lecturer/student communication. This thesis looks at the design and implementation of it, and focuses on the smart client. It explains the design requirements and pedagogical requirements, and how they led to the architecture and implementation that we chose. It describes the steps to building the software and environment for the system, and the technological advancements and results obtained from its use in class.

Thesis Supervisor: Nishikant Sonwalkar
Title: Principal Educational Architect, AMPS
Acknowledgements

This thesis is the product of years of toil and help from several people. I would like to thank them for their help and recognize the many contributions made to this work.

First of all, I would like to thank all of the members of the NGMC project at MIT, especially Mark Tompkins, John Cogliano, and Manuel Centeio. They have all spent many hours in order to make this project a success.

I would also like to give many thanks to Dr. Nishikant Sonwalkar, for advising me on this thesis and all of the other help he has provided me over the past five years. He provided the pedagogical ideas for this thesis and came up with the educational framework that we followed in building it. We also hope he will do an analysis of it once the bugs have been worked out. His papers were the inspiration for the ideas of NGMC.

One person whom we would never have been able to do this project without is Professor Ed Barrett. Ed was the one who was willing to take a chance on the project when other professors were not. His willingness to look into the future and explore was an inspiration to us all. He opened his classroom to us and was very open to new ideas.

I would also like to thank many other people who have made an influence on my life and who I could not be here without:

My parents, Subhash and Kanchan, who put up with me for the last 23 years.

Dr. W. John Dickerson, M.D., whose wisdom is surpassed only by his lankiness.

Microsoft and the iCampus Project, which was generous enough to provide us with $60,000 of funding to create open-source educational systems.

I am going to Miss MIT.
# Contents

Chapter 1. Introduction .................................................................................................................... 8
   1.1. The Roots ............................................................................................................................... 8
   1.2. Pedagogical Theory ............................................................................................................... 11
   1.3. To Build a Better Lecture ..................................................................................................... 14
   1.4. In the Trenches .................................................................................................................... 15
   1.5. A Viable Technological Solution ......................................................................................... 17
   1.6. And Onward.......................................................................................................................... 18

Chapter 2. Related work .................................................................................................................... 19
   2.1. Flash Cards .......................................................................................................................... 19
   2.2. EduCue PRS ........................................................................................................................ 21
   2.3. Classroom Communicator .................................................................................................... 22
   2.4. UCSD ActiveClass ............................................................................................................... 24

Chapter 3. General System Architecture .......................................................................................... 26
   3.1. Overall Architecture ........................................................................................................... 27
   3.2. Back-end Architecture ........................................................................................................ 29
       3.2.1. Database Design ........................................................................................................ 30
       3.2.2. Web Application Design ........................................................................................... 32
   3.3. Communication Protocol .................................................................................................... 33

Chapter 4. Smart Client Architecture ............................................................................................... 34
   4.1. Executive Layer ................................................................................................................... 36
   4.2. Message Handlers ............................................................................................................... 37
   4.3. Alert Handlers ..................................................................................................................... 39
   4.4. Plug-ins.................................................................................................................................. 40
       4.4.1. An Example Plug-in: Elevator Speech Recorder .......................................................... 41
   4.5. File Upload Manager .......................................................................................................... 42

Chapter 5. Security and Privacy ......................................................................................................... 45
   5.1. A Basic Scenario ................................................................................................................... 45
   5.2. IR Login ................................................................................................................................ 48
       5.2.1. Login System Design Choices .................................................................................... 48
       5.2.2. IR Login Implementation Details .................................................................................. 50

Chapter 6. Message/Alert Handlers .................................................................................................. 52
   6.1. Message Handler: Scheduler ............................................................................................... 52
   6.2. Message Handler: Contact Information .............................................................................. 54
   6.3. Alert Handler: Quiz Alert ...................................................................................................... 56

Chapter 7. Implementation Guide ..................................................................................................... 57
   7.1. Building from Source ......................................................................................................... 57
       7.1.1. Development Source Tree .......................................................................................... 57
       7.1.2. Pre-compile Constants ............................................................................................... 60
   7.2. Installation and Setup .......................................................................................................... 61
   7.3. Basic Usage .......................................................................................................................... 62

Chapter 8. Implementation and Results ........................................................................................... 63
   8.1. Usage Patterns ..................................................................................................................... 64
       8.1.1. Usage by Feature ......................................................................................................... 64
       8.1.2. Usage by Client Type .................................................................................................... 65
       8.1.3. Usage by Time .............................................................................................................. 66
8.2. Survey Results .................................................................................................................. 67
8.2.1. Usability of Login ......................................................................................................... 70
8.2.2. Usability of Smart Client Shortcut Functions .............................................................. 71
8.2.3. Usability of the File Upload System ............................................................................ 73
8.2.4. Usability of the Ask Question Facility ........................................................................ 74
8.2.5. Usability of Quizzes, Surveys and Mini-quizzes ............................................................ 75
8.2.6. Usability of the Calendar/Events System .................................................................... 76
8.2.7. Usability of the Contact Information Feature ............................................................. 76
8.3. Overall User Opinion ....................................................................................................... 77
8.4. Further Directions for Analysis ..................................................................................... 81
Chapter 9. Future Work ........................................................................................................ 82
9.1. Complete Pedagogical Analysis .................................................................................... 83
9.2. Enhanced Security .......................................................................................................... 84
9.3. Tablet PCs ....................................................................................................................... 85
9.4. Pervasive Classroom ....................................................................................................... 86
9.4.1. Classroom Hardware Interface ................................................................................ 86
9.4.2. Distribution of Devices .............................................................................................. 87
9.4.3. Plug-ins ......................................................................................................................... 87
9.4.4. Out of Class Tools ....................................................................................................... 88
9.5. Other Future Work .......................................................................................................... 88
Chapter 10. Conclusion .......................................................................................................... 90
Bibliography ........................................................................................................................................ 92
Appendix A ....................................................................................................................................... 95
Appendix B ....................................................................................................................................... 97
Appendix C ....................................................................................................................................... 101
List of Figures

Figure 2 The Learning Cube (used with permission from Dr. Sonwalkar).........................12
Figure 6 The table diagram for the NGMC backend database........................................31
Figure 7 The smart client user interface........................................................................34
Figure 8 The smart client module hierarchy diagram....................................................35
Figure 10 The NM_HTMLVIEW structure used by message handlers..........................37
Figure 11 Pseudo-code for the alert check mechanism of the executive.......................39
Figure 12 The generic FUM user interface......................................................................43
Figure 13 The SID transfer timeline................................................................................47
Figure 15 The POOM-accessible calendar included in PPC 2002.................................53
Figure 16 The contacts interface in PPC 2002 with information entered via NGMC........55
Figure 17 The smart client project open in EMVC on Windows XP..............................59
Figure 19 The smart client user interface with buttons highlighted..............................62
Figure 20 The percentage of total usage by feature......................................................65
Figure 21 The percentage if total use by the client type (smart client or other)..............66
Figure 22 Graph of number of page requests per hour on class days.........................67
Figure 23 Overall ease of use histogram (1 is least difficult, 5 is most difficult).............69
Figure 24 Overall usability histogram (1 is most usable, 5 is least usable).....................69
Figure 25 IR login ease of use histogram (1 is least difficult, 5 is most difficult)..........70
Figure 26 Manual login ease of use histogram (1 is least difficult, 5 is most difficult).....71
Figure 27 Smart client button ease of use histogram (1 is least difficult, 5 is most difficult).72
Figure 28 Quick menu ease of use histogram (1 is least difficult, 5 is most difficult)......72
Figure 29 File upload ease of use histogram (1 is least difficult, 5 is most difficult).......73
Figure 30 Ask question ease of use histogram (1 is least difficult, 5 is most difficult)......74
Figure 31 Survey, mini-quiz ease of use histogram (1 is least difficult, 5 is most difficult)....75
Figure 32 Events access ease of use histogram (1 is least difficult, 5 is most)..............76
Figure 33 Contact information ease of use histogram (1 is least difficult, 5 is most)......77
Figure 34 Answer to “would you like to use this system again” histogram (1 is definitely, 5 is definitely not).............................................................................................78
Figure 35 Answer to “would you like to see this in other classes” histogram (1 is definitely, 5 is definitely not)........................................................................................................79
Figure 36 Answer to “would you like to use the PDAs more in class” histogram (1 is definitely, 5 is definitely not)........................................................................................................80
Figure 37 Answer to “would you like to use the PDAs more outside class” histogram (1 is definitely, 5 is definitely not)........................................................................................................81
List of Tables

Table 1 The lifecycle of a smart client plug-in................................................................. 41
Table 2 Parameters for ngmc-sched message (in position order)....................................... 54
Table 3 Parameters for ngmc-contact message (in position order)..................................... 55
Table 4 The smart client directory structure..................................................................... 58
Table 5 Smart client pre-compile constants....................................................................... 60
Chapter 1. Introduction

When we began the Next Generation Mobile Classroom project, we had one goal: to simulate the experience of a small classroom in a large lecture hall. Countless studies have shown the benefits of small classrooms, from increased attention from professors to better interaction among students. Unfortunately, the university system is plagued with large lecture classes, often with 500 or more students. To solve this problem, many universities have tried to cut back on class sizes, or place professors in "lab" settings where they can work more directly with students. However, there is still an enormous number of large lecture classes, and they do not seem to be going away any time soon.

1.1. The Roots

As MIT students, we experienced the ills of large lecture classes first hand. In freshman and sophomore classes, we would often find ourselves among two hundred, three hundred, or more students in a lecture hall the size of a football stadium. Aside from being lost in the crowd, we were often seated too far from the lecturer to hear what was going on—much less pay attention. Far too often the result was to put our heads down and fall asleep. We may not have learned much about Vector Calculus from these experiences, but we did learn a great deal about lecture hall education. And from what we learned, we could see the flaws of the large lecture class and how to use technology to overcome them.

The first step in creating what we hoped would be an innovative new type of lecture was to take a critical look at the current state of lecture education from the point of view of students and professors. We created a small focus group of MIT students, and one of MIT professors. For each one we chose several people to come together as a sounding board and
discuss their feelings on how well lecture halls work, what materials or tools are available to enhance the experience, and how (if at all) these tools may be worked into a real class. What we found was that both students and professors were not satisfied with the large lecture experience, yet they had different ideas on how to fix it.

The students welcomed the idea of introducing technology into the lecture hall. Several already used laptop computers in lecture and saw no difficulty in using technology to communicate with the instructor as well. They agreed that it would help them pay better attention in class. And they noted that it could also help the lecturer pay more attention to them by signaling to him/her whether the students understand the material or are confused—especially if this could be done anonymously. Often, professors explain material too quickly or spend too much time on more trivial concepts. A lecture communication system could allow students to “flag” the professor and tell him/her to adjust the pace of lecture anonymously. It could also allow students to ask questions anonymously, without disrupting the professor.

Professors, however, were more concerned technology in lecture. While they agreed that in principle students could benefit greatly from being able to signal their understanding of material, they feared that giving students a piece of technology during lecture could cause distraction. One professor said that students would be tempted to use instant messaging or browse the web during class. Others were concerned that simply entering a question into a
device or answering a question with it would make them miss crucial seconds of material. The professors we spoke to were also against the idea of being assessed during class, even if the form of this assessment simply an “I’m lost” signal. As may be expected, this sort of assessment could be taken as criticism of the professor’s style, which is much better seen in private.

We also found that without a complete system to promote interaction in lecture classes, professors have come up with creative solution to the problems that they face in large lectures. A mathematics professor at MIT that we spoke with, Haynes Miller, told us that he uses “flash cards” to allow students to respond in his courses [Mil01]. With approximately 500 students, most of whom are sleep-deprived freshmen, he realized he needed some form of interaction to keep students engaged—or at least awake. So he gave each student three cards: one red, one yellow, and one green. During lecture he poses questions, such as simple problems to demonstrate the current concept, to the class, which the students answer by holding up one of the cards.

From this idea grew more complex in-lecture systems with similar purposes. The PRS system, distributed by EduCue, is one of these (we will look at the PRS system and its advantages and disadvantages in Section 2.2).1 The PRS system is made up of a bunch of IR transmitters and a receiver that is mounted in the lecture hall. Students each get a transmitter, and when asked a question, they push one of the 10 buttons on the transmitter. A screen at the front of class then shows which button was pressed. Unfortunately, the PRS system is only one-way, and there is no way for students to ask questions or express anything more complicated than a simple multiple choice answer.

---

1 Complete information on PRS is available at the EduCue website at http://www.educue.com.
Along this path, other lecture hall technologies came as well. At MIT, Eric Brittain developed the Classroom Communicator. This project used WAP-enabled cell phones to allow students to interact while sitting in lecture. It went beyond PRS by allowing students to ask questions and provided two-way communication between professors and students. Yet, because of cost and compatibility issues, this project proved infeasible in a real classroom environment.

The most promising of lecture hall technologies—and one that we will look at in depth later in this paper—was the UCSD ActiveClass system. ActiveClass, as we will see, is similar to NGMC in architecture. Each student receives a handheld computer with wireless interaction. In class, students can ask questions and respond to lecturers using these. But there were many issues with large lecture classes that we felt the ActiveClass system did not solve, such as integration with students’ daily activities and work schedules. These issues, we felt, merited a new design that would build on the ideas of ActiveClass while making the system more practical and useful.

1.2. Pedagogical Theory
To form a pedagogical framework for our design, we turned to Dr. Nishikant Sonwalkar, renowned MIT researcher and inventor of the “Learning Style” model of instruction which essentially states that people learn in different ways, and in order to maximize their understanding of material, one must present the material in a manner that matches their way of learning, or “Learning Style” [Son98].

---

2 Compete information on the Classroom Communicator project is available at the MIT Educational Technologies Group website at http://web.mit.edu/etg/www/  
3 Details on ActiveClass are available from http://www.ucsd.edu.
Dr. Sonwalkar explained to us that different forms of education can be depicted graphically using a three-dimensional figure called a Learning Cube (see Figure 2).

The y-dimension (horizontal) of the map shows the different Learning Styles, \( L_1 \) through \( L_5 \). The z-dimension (vertical) of the map shows the different media formats in which educational material can be delivered. And the x-dimension (depth) is the modality—teacher-centric, student-centric, or somewhere in between. The Learning Styles, referenced by \( L_1 \) through \( L_5 \), are “apprenticeship,” “incidental,” “inductive,” “deductive,” and “discovery” [Son01].

The problem with a typical large lecture is that it essentially flattens the Learning Map cube into one-dimension. In a large lecture, professors can provide different media formats—audio, graphics, etc. But, the modality automatically becomes teacher-centric, and therefore they are locked into using one, and only one, Learning Style. In a small class, students can give direct feedback to the professor and thus tailor the lecture to his particular Learning Style. This empowers the student, and moves the modality toward student-centric. However, in a large class, students have no method of selecting a Learning Style. This
means they are at a significant disadvantage if their preferred Learning Style does not match the one in which the course is taught.

Dr. Sonwalkar proposed that with the right technology, it is possible to give students access to the entire Learning Cube. For example, in his research [Son01B], he created web-based courseware in which students could view material in multiple Learning Styles and media formats, allowing for extremely student-centric education. Assessments provided during a course help the system decide the best combination for each individual student. He explained that NGMC could do the same thing for a large lecture class by providing a similar sort of assessment. If a lecturer asks a question at the end of each concept that he explains, and each student can respond, the lecturer can tell how many understood the concept and how many did not. Furthermore, by adding the ability for students to ask questions without disturbing the flow of lecture, the lecture can be even more tailored to the specific needs of the students. In essence, this allows students to learn in their Learning Style while in a large lecture.

Dr. Sonwalkar’s research provided us with a strong pedagogical basis for NGMC, making us much more confident that a system such as this could meet its objectives. His work also provided a way of measuring the effectiveness of NGMC called the Pedagogical Effectiveness Index, or PEI. Although we do not do complete pedagogical analysis using the PEI in this thesis, it is an important step in determining whether the system can truly enhance a lecture. Once NGMC’s bugs and usability issues are solved, we plan on doing a complete analysis and using PEI data to quantitatively measure the pedagogical value of NGMC.
1.3. To Build a Better Lecture

After doing this research, we felt ready to design a new system to address the drawbacks of current ones. We had decided early on to use some sort of wireless network with handheld computers as a foundation for the system. But everything else was to be designed from the ground up given the input we had. The first step that we took was to create what we called a "pedagogical requirements" list, which would delineate all of the requirements that the students and professors made up for us. Along with pedagogical requirements, we created a list of "technical requirements," which were mostly requirements set forth by the state of technology, but included requirements due to cost and space restrictions. The requirements that we came up with will be explained in detail in Chapter 2.

With this list and generous funding from the Microsoft iCampus initiative at MIT, we set out to design and build the system. Because there were to be multiple people working on it, the system was essentially split into two main parts: the NGMC web application and the NGMC "smart client." The web application was to contain all of the server-side software and architecture. The smart client was to be all of the software and tuning required to make the NGMC system run on handhelds (see Figure 1). We chose to do all of the infrastructure design and polishing at the end as a team.

It is worth noting here that as the title indicates, this paper focuses on the smart client and the general infrastructure issues. For detailed information on the web application and server-side components, see [Cog03]. Nevertheless, it is necessary to use some sections of this paper as a more general overview of the system, since much of the work that went into it cannot easily be classified into the categories above. A general understanding of the system makes the smart client parts more comprehensible and provide a way to understand how the smart client fits in with the system as a whole.
1.4. In the Trenches

Once design and construction were complete, the NGMC system had to be field tested. We had some trouble in the beginning getting professors to sign on to use debut version. Most of the professors—including those whom we initially talked to—had classes that were too big to fit our test budget. At approximately $1000 per handheld (including all of the support hardware, software, etc.), a 500 student class was not feasible. Other professors reiterated the concerns above: that handhelds in lecture would be too distracting or would take away from their lecture. Fortunately, Professor Ed Barrett, one of our former humanities professors who has a history of early adoption of new classroom technologies, agreed to work with us.

In Fall 2002, the NGMC system finally saw the light of day. We set the system up in Professor Barrett’s “Communicating in Cyberspace” course. This course had just the right number of students to allow us to make a fair evaluation of NGMC while still being able to afford the cost. We purchased a Toshiba e740 handheld PC\(^4\) for each of the students because of their built-in wireless capability and installed our software. We also set up a login station where students could swipe their handhelds to quickly login to the class (see Section 5.2). In the lecture hall itself, we set up a projector and connected it to a computer connected to the NGMC server. We also gave all of the faculty and support staff Toshiba e740s. With everything together, the professor and guest lecturers were free to use the system while teaching.

\(^4\) The Toshiba e740 is available from most computer stores. For more information see http://www.toshiba.com
The class got off to a somewhat rocky start using NGMC, but was soon able to use it to take full advantage of its features. The first few meetings of the class were spent getting the bugs out of the system. At first, the wireless network was not cooperating. We found that this was due to a variety of factors, including technical issues with the hardware and political issues at MIT. Once the network was fixed, the Toshiba handhelds began to act strangely—we later learned that this was due to a fatal flaw in the hardware, for which the Toshiba e740s were eventually recalled. Finally, however, the class was up and running with the system. Each meeting, lecturers would use it to ask questions and pose surveys to the class. The professor would use it to allow students to rate each other on in-class presentations and to provide an anonymous method of commenting on class material. And the students would use it to track group contact information and to schedule meetings, presentations, and conferences with the professor.

Throughout the term students used the NGMC system and the system collected data on their usage patterns. From the data, we were able to identify which functions of the system users liked and disliked, which were easy-to-use or more difficult, which could be modified or expanded, and which should be removed. We also surveyed the users and asked them specific questions of this nature. We discuss the results in Chapter 8.

Beyond basic collection and plotting of data, this thesis does not do pedagogical analysis of NGMC. It is very important to do this sort of analysis before the system is actually put into place. However, due to the small class size and the fact that the system is still in its early version of development, we leave this as future work (see Section 9.1). Dr. Sonwalkar's research defines a Pedagogical Effectiveness Index, or PEI, which can be used to form a qualitative measure of the educational performance of NGMC. This index can also be used to compare NGMC with other methods of teaching, such as standard lecture, as well as web-
based. Once the bugs and usability issues in NGMC have been solved, we plan to compute the PEI score of the system for a larger class with more realistic lecture conditions.

1.5. A Viable Technological Solution
The main technical contribution of NGMC is that it makes several advancements in handheld-server messaging via a “smart client” (which will be described in detail later). Because of the “smart client,” NGMC is able to solve many technical problems of communication and synchronization between handhelds and the Internet which had not previously been solved.

One important outcome of this is the NGMC messaging and alert system. NGMC implements a new messaging system that allows secure and anonymous communication over heterogeneous networks. This means that handheld productivity applications, such as calendars or address books, can directly integrate information from a data source thousands of miles away. The NGMC messaging protocol handles all of the complexity of routing these messages in real-time. This messaging system has applications in many different fields, and is not simply limited to in-lecture communication.

This thesis is intended to catalog and describe the technical contributions of the NGMC system. It explains how the “smart client,” by looking beyond simple question-and-answer systems, forms the basis of a pervasive educational environment, which not only addresses in-lecture issues but also addresses the issues of class-to-class transitions and student organization for large lecture classes.
1.6. And Onward

The remainder of this paper looks at the design, construction and evaluation of the NGMC system with special focus on the smart client portion. Chapter 2 describes the related work in the field of lecture enhancement technology and looks at NGMC's pedagogical design, explaining the requirements that we had in building it. Chapter 7 is a user's guide for deploying, using, and maintaining NGMC, with special attention paid to the smart client and the infrastructure needed to support it. Chapter 3 discusses the overall system architecture for NGMC. Chapter 4, Chapter 5, and Chapter 6 focus on the smart client, with specific details about architecture, security and privacy, and message structure. Chapter 8 provides an evaluation of the system with a look at the field results. Finally, Chapter 9 and Chapter 10 conclude the paper and look at future directions for the project.
Chapter 2. Related work

As noted in the introduction, there are a few different related projects that have been in the works at other universities and companies. While these projects bring technology—sometimes handhelds—into the classroom, they are not quite as advanced as NGMC in features and portability. In this chapter, we will take a look at the different projects, and discuss their good and bad points. We will also explain the conclusions that we drew from learning about these projects and how these conclusions affected the design requirements for NGMC. This chapter looks at four representative examples, flash cards used by Professor Miller, the PRS system by EduCue Corp., the Classroom Communicator at MIT, and the ActiveClass system at UCSD.

2.1. Flash Cards

Flash cards may not seem like an appropriate subject of discussion in a technical thesis like this; however, studying how students used them provided us with invaluable information on how to design NGMC. Professor Miller, the math professor who introduced us to this idea, explained to us that it provided a simple, inexpensive way in which students could interact during lecture [Mil01].

Each student was provided with three flash cards, one red, one yellow and one green. Whenever the lecturer wished to test the students' understanding of a concept, he would write a multiple choice mini-quiz on the board—usually containing only one question—where each answer is represented by one of the three colors. He would give the students a minute or two to work out the problem or problems, and then tell them to raise the card that they believed represented the correct answer.
This technique actually worked very well. It caused little or no disruption in class, while still providing valuable feedback to the professor. Moreover, it was very inexpensive, and required little of the students, except that they come to class equipped with the cards. Furthermore, flash cards cost nothing to use in another class or to use with a different subject type.

Still, there were problems with flash cards. They are, of course, easy to lose. Many students would lose them and simply stop participating in lecture. Or they would forget to take them to class. Those who managed to hold on to their flash cards and take them to class did not always participate either. Professor Miller estimated that less than 50% of students in class actually used the flash cards when asked a question. Without any form of accountability, it was easy for students to simply “blow off” the mini-quizzes. There was also the problem with counting them and determining a majority answer. Although most of the time, there was a large majority who held up a certain color, sometimes it was impossible to tell who was right and who was wrong.

More importantly, however, flash cards had very limited functionality, and they addressed very few of the problems with large lectures. For example, there was still no way for a student to ask a question to the professor, or to signal that he is lost. When a student did attempt to do one of these things, he not only had to be bold enough to speak out among several hundred students, but he also had to disrupt the lecture. The flash cards provided no privacy, so anyone could tell who was asking the questions or who answered the mini-quiz incorrectly.

This taught us several things that would help us design NGMC. First, the system must be simple and easy to use if students are going to use it. If some students are unwilling to raise a flash card, they will definitely not be willing to set up a complicated device and
navigate through menus when asked to answer a question in class. But it also taught us how important the issues of privacy and security are, and that allowing students to communicate privately with the professor in lecture would make them much more comfortable. It also left a large number of other issues unresolved.

2.2. EduCue PRS
A more technological system designed to do a similar thing is the PRS system by EduCue. The PRS system is basically the high-tech version of the flash card idea, with a few extra advantages [Edu03].

PRS, or Personal Response System, is a system made up of a set of small infrared transmitter devices and a receiver. The transmitters each have several buttons, and are distributed to students at the beginning of class. When a professor wants to allow the students to interact, he may present a mini-quiz on the board, with multiple choice answers corresponding to buttons on the PRS transmitter. The students must then answer and the PRS receiver records the answers and sends them to a computer.

The advantage of this over flash cards is that a professor can tell who answered what, and thus can also tell if people have not answered at all. Therefore responses are much more strongly encouraged than they are with the flash cards. It is also easier to tell whether the majority answered one way or another. Finally, the PRS system is capable of having an “I am lost” button, which anonymously signals to the lecturer that one or more students does not understand the material, and the lecturer should move more slowly.

There are many drawbacks to PRS as well. First of all, at around $50 per transmitter, it is much more expensive than flash cards—especially given the modest benefit. More than that, however, it is not at all extensible or versatile. It can only handle multiple choice
questions, and it does not offer any additional functionality to address other problems with large lectures. It is also somewhat insecure, since IR devices given out each class are extremely susceptible to technical and non-technical hacking. Furthermore, it requires its own infrastructure of receivers placed in the classroom, so it is not easily portable.

Studying PRS, we determined several additional requirements for the NGMC system. First, it needs to be versatile, so that it can be used in any classroom for any subject. This means that the system must be technically portable, so that it can be moved around from room to room. Thus, it cannot rely on much special hardware in class, and it should try to take advantage of the existing MIT infrastructure as much as possible. It also means that the software must easily adapt to any type of subject, regardless of special requirements. Therefore, it should have some sort of plug-in mechanism and should have a comprehensive user structure that can handle classes, students, professors, administrators, TAs, user groups, student groups, recitation or lab groups, etc. Second, it needs to allow not only multiple choice questions, but more complicated interactions, including students and professors, as well as TAs, administrators, and support staff. Third, from student comments about PRS, we realized that NGMC would need to provide some level of privacy and anonymity, so that professors cannot constantly watch over everything students do.

2.3. Classroom Communicator
The classroom communicator was the first major project of this kind to take a different approach to enhancing large lectures. It used more advanced devices than simple multiple choice systems in order to provide a higher level of interaction than previous systems [Bri02].
Designed to work with WAP-enabled cell phones [WAP01] and the wireless web (see references on WAP at the end of this paper), the classroom communicator provided the basic ideas upon which the NGMC system were based. Each student must bring one of these cell phones to class and have it running as the class progresses. If a lecturer wishes to ask students a multiple choice question in the form of a mini-quiz, he can use a Java interface to enable the question, whereby each student can access it with his cell phone and answer it using the numerical keypad. But it expands on this idea by allowing students to enter questions for the TAs and lecturers using the alphanumeric entry, and providing other features to promote interaction.

The benefit to this approach is that it provides a much more interactive and adaptable environment, where students can go beyond simply answering multiple choice questions. It can also be expanded easily to provide more features, or specific features for different subjects or classes. This feature was not possible with flash cards or systems like PRS. Moreover, WAP cell phone service is available almost anywhere, and does not require any special infrastructure in class to work. Furthermore, because of the strict security precautions taken by wireless web provides, this was the first major classroom project to provide sufficient security and privacy.

However, there are several drawbacks. Wireless web access with cell phones is still extremely expensive, since almost all plans are charged by the minute. Also, there are no wireless web standards for cell phones (Sprint, AT&T, etc. all have different protocols), so students must have the same plans and phones, making the cost of the system even higher. The functionality of a phone is another disadvantage, since it is very difficult to write questions using an alphanumeric keypad like those on a typical phone, and the extremely small screen makes any elaborate display impractical.
These qualities led us to several conclusions about NGMC. First, the system should not cost too much. We realized that this statement was somewhat vague, but it basically means that the system must use only standard hardware, preferably for which price is dropping. It also should not include any per-use costs, such as network fees. Therefore, it should only require existing MIT networking infrastructure to run. Second, the system needs to be built using devices and protocols that are standardized: We thought about making it 100% web/HTML-based, but soon realized that there are many special features that cannot be created using these basic technologies. What we concluded was that the system should be completely compatible and usable with standard web browsers, but that it can also take advantage of special “smart clients” that enable additional functionality. Finally, the Classroom Communicator showed us the importance of security and privacy in a system like this, which became a very important goal in the final design of NGMC.

2.4. UCSD ActiveClass
The UCSD ActiveClass system is the newest of the ones mentioned here, and it is the most advanced. Based on Pocket PCs and an 802.11b wireless network, this system allows students to interact in a much less restraining manner than the other technologies, since it provides complete two-way communication. Students each receive a handheld with wireless access, and log in to the ActiveClass web site on their handheld. Using the system students can answer mini-quizzes or surveys, ask questions and vote on them. Also, unlike previous systems, ActiveClass allows students to access lecture outlines and messages of the day before class and perform a few other tasks [TGR02].

The system has many good points. The feature set is fairly complete, and the user interface makes it easy to use. It is not limited to a small feature set, since it uses Pocket
PCs, which function much like larger computers. It does not provide any revolutionary features beyond those in Classroom Communicator, but it does make all of the features usable.

The use of existing wireless architecture was also a good design choice, since it allowed the ActiveClass system to work easily and cheaply. ActiveClass uses 802.11b [IEEE96], which is a standard for providing Ethernet-compatible connectivity over radio waves. What this means is that with the appropriate infrastructure, any device that is within the range of the radio waves can have Internet access. Since many schools have already built a complete 802.11b infrastructure, systems using 802.11b can work anywhere with no additional cost other than the fixed cost of the infrastructure hardware.

ActiveClass essentially fixed all of the problems of earlier systems; however, because of its simple web interface, it was not able to provide a strong interactive experience. For example, the way in which it determines if there is a new question is that it reloads the main page every 30 seconds. Not only is this a big load on the server; it also means that a quiz could appear up to 30 seconds after shown. Since most of these quizzes are only a minute or less, valuable time is lost. Also, there is no integration with the handheld software. Many people currently use handhelds, such as Palm computers, to store contact information, events, and other important information. Because ActiveClass is completely browser-based, it is incapable of integrating with these features and thus awkward to use. Also, because students must enter their usernames and passwords to login at each class, it is a burden to begin using the system. And important time is wasted when students could be doing things such as reading about lecture. Finally, the browser restriction makes ActiveClass difficult to expand and specialize for specific classes.
This system influenced the design of NGMC in several ways. We very much liked the idea of using Pocket PC handhelds and 802.11b for wireless access. These provide a very extensible platform that does not have any hidden or recurring costs. We also liked how the features were implemented cleanly and were easy to use. This was especially true for the ask question feature, which allowed students to rate other students' questions to show urgency. But we also saw a few flaws with the feature, such as the fact that it disturbs professors while they are lecturing. We figured that it would make more sense to allow intermediaries such as TAs to view the questions and pass the urgent ones to the professor. In addition, the idea to base the system on HTML with a web browser was good to provide platform compatibility; however, it was also clear to us that the system sacrificed features because of this restriction. We decided that our "smart client" idea, which provides additional features to a regular web-based client, was the best compromise. ActiveClass also showed that it was important to integrate the smart client with the Pocket PC productivity applications, and make it easy for users to log in when they enter a class.

Chapter 3. General System Architecture

Software architecture is a very important part of a system design. In building the NGMC system, we had to make several design choices. There was no existing software infrastructure, so we were free to make whatever design choices we felt were necessary. The software architecture was to a small degree specified by the hardware that was available and the networking capabilities, but these constraints did not prove to be a major issue. What was more of an issue in the design was ensuring that all of the educational and technical requirements were met, and that the system ran at optimal speed with very few bugs. This
section will examine the software design of NGMC and will explain all of the decision choices that we made in the design process.

3.1. Overall Architecture
The NGMC software has a three-tier architecture, consisting of a database, a web server running a specialized web application, and a “smart client” front end.

In choosing this overall architecture, we had many options. Originally, we proposed a simple client-server scheme (see Figure 3). This scheme would have placed much of the intelligence in the client, and would use the server as a simple data repository and message relay. This would have allowed for high-speed automatic updates, and made it easy to closely integrate the NGMC client with the handheld OS. However, there were several flaws to this scheme. First, with this scheme, users would be locked in to the client device that we chose. That means that users could not use laptops or other devices to access the system. Second, it would require us to build a transaction server from scratch, with the ability to securely and reliably store data. Not only is this difficult; it is also very prone to bugs and concurrency issues.
We then began to look into the option of giving the system a basic three-tier architecture. This is shown in Figure 4. This would entail a back-end database, with a web server and web application on the middle tier, and a basic web browser on the front end. This design choice had several advantages over the previous one. First, it made the client platform and device independent. Thus, any device with a basic web browser would be capable of connecting to NGMC. This includes Palm devices, cell phones, and workstations, in addition to the handhelds and laptops originally chosen. It also made it easy to support and update the system, since all the business logic would be contained on the server. In addition, there would be no need for users to download and install software on their devices. However, like the others, this architecture had important drawbacks. Because of the limited functionality of the web browser, there were many functions of the system that would have to be left out. Pop-up quizzes, for example, require some sort of server-push mechanism to work. A typical web browser does not allow this, so students would be forced to click to access quizzes. In addition, with a typical web browser, it is impossible to integrate with the address book and calendar on the handheld. Without this capability, there would be no use for the group or scheduling functions of the NGMC system.

The architecture that we finally chose was a mix of the two above. By combining the three-tier architecture with a smart client that was capable of providing specialized services on the client side, we were able to take advantage of the benefits of both architectures (see Figure 5). Like the three-tier

Figure 5 The smart-client architecture option
architecture, at the back end, the NGMC system used a relational database. The database we chose to use was the open-source database MySQL. We had originally chosen to go with Microsoft SQL Server; however, we later decided that the open-source route would better serve us. We did, however, use only ANSI standard database code, so NGMC could easily be ported to another database if necessary. For the middle tier, we chose Apache with PHP on Windows 2000. Apache and PHP are open-source, and they work seamlessly with MySQL. On top of Apache/PHP we built a web application for serving most of the NGMC functionality. Finally, on the client side, we offered a choice. The client could either use a standard web browser, in which case some of the specialized functionality would be disabled, or it could use our "smart client" program. The server-side application is capable of automatically detecting the client platform, as well as whether the client is using a browser, and then formatting the data accordingly. We developed the smart client for the Pocket PC 2002 platform using Microsoft Embedded Visual C++. The smart client has many specialized features that are not available when using a standard web browser, but it is not easily portable to other operating systems or devices.

3.2. **Back-end Architecture**
The following sections provide a basic overview of the NGMC back-end architecture. This overview is intended to provide only the information necessary to understand the smart client, which is the focus of this thesis. For more information on this portion of the project, see [Cog03].
3.2.1. Database Design

As noted earlier, the database backend of the NGMC system runs on MySQL. We chose MySQL because it is open-source as well as easy to set up and use. Microsoft SQL, our original choice, proved to be a heavier load on our server hardware than we had hoped, and we found that MySQL ran a lot faster. Unfortunately, MySQL does not support enterprise features such as load balancing and per-transaction logging; these would be very important when scaling the system to a large number of users and using it for mission-critical applications, such as grading.

The database design is typical for a web application. It consists of 16 tables to store user and group information, media metadata, and questions. The structure of the database is shown in Figure 6.
Figure 6: The table diagram for the NGMC backend database.
It is out of the scope of this thesis to do an in-depth analysis of the database structure, since this is described in detail in [Cog03].

3.2.2. Web Application Design

The NGMC web application is designed to perform most of the “business logic” for the NGMC system. All of the functions available on the system use data that is processed by this web application. The main design goal in building the web application is to provide all of the features listed in the previous chapter while running as smoothly and quickly as possible. The NGMC web application runs on an Apache web server running a PHP interpreter. We chose PHP because it is simple and open, unlike competing languages such as ASP or JSP. This sub-section gives a brief overview of the design of the NGMC web application.

The NGMC web application consists of a core module, which maintains session, user, and event information, as well as several plug-in modules that provide functionality to the system. In building the web application, much care needed to be taken to ensure that it was usable on any client device. Therefore, the core module contains special code that detects a client device and operating system and then renders the page accordingly. User and group management is done by a complex user manager, which monitors permissions on all objects in the system. Users and groups can be members of groups, recursively, and classes are simply groups of students with the professor as a head. Finally, plug-in modules are used for all of the special functionality of the NGMC system, such as quizzing, message-of-the-day, and calendaring.

The quiz module works by storing all of the quiz questions in the database, then sending a quiz message to the client, which opens the quiz URL. A user of the system—
typically a professor or a TA—uses the quiz administration tool to create a quiz. Once this is done, the user sets the permissions of the quiz so that it visible to the students. This forces the web application to send a goto:quiz message to the client, which in turn loads the quiz in its browser. This is explained in detail in the next chapter.

The calendar and group information components allow users to enter information via web forms. The data is stored in the database until someone requests to distribute it. When this happens, a ngmc-sched (for calendar) or ngmc-contact (for address book) message is sent to the client, and the client parses the information that is sent with the message and incorporates it into the handheld's internal software. These messages (and how they are processed on the client side) are explained in the next chapter.

Finally, the student to teacher questions, the MOTD, and the other modules use simple web forms as input and store the information to the database for later use. These modules use no specialized functionality on the client side.

3.3. Communication Protocol
The NGMC system uses the 802.11b wireless protocol for all data transfer. This is a standard protocol, which has been very thoroughly tested and perfected over several years. The 802.11b network allows Ethernet packets to flow between wireless and wired networks seamlessly [Ava01]. For link and data communication, NGMC uses TCP/IP over 802.11b, which is well supported and provides all of the connection services that the system needs. It is possible to also use UDP or some other protocol over 802.11b, but this is unnecessary with our current requirements. Also, TCP/IP provides reliable and persistent connections, something that is very important because 802.11b is inherently not reliable by itself [HK01].
Chapter 4. Smart Client Architecture

As explained earlier, the NGMC smart client is intended to provide special functionality that would not be available with a standard web browser. The smart client replaces the standard browser on platforms where this option is available (see Figure 7).

Figure 7 The smart client user interface

With it, features like pop-up quizzing, calendar and address book synchronization, and file upload are enabled, thereby making courses more interactive and making the system easier to use. This chapter discusses the design of the smart client in detail, describing all of the components and how they interact.

The smart client program is built on Microsoft's Pocket PC (PPC) 2002 platform using Embedded Visual C++. We had originally wanted to use Java or Visual Basic; however, neither is capable of integrating with the low-level operating system services that
the smart client needed. Embedded Visual C++ with the Embedded Visual Tools platform API was the only language/API combination that provided access to these services. We chose to use Microsoft Foundation Classes (MFC) with Embedded Visual C++ to reduce the amount of UI and system coding necessary. By using these development tools, we were able to compile the smart client to run on any device using PPC 2002—including Pocket PCs, Handheld PCs, and even some tablet PCs—with the Intel ARM or X Scale processor.

The basic architecture of the smart client program is a module hierarchy, as shown in Figure 8.

![Figure 8 The smart client module hierarchy diagram](image)

At the base of the smart client is the program layer. The program layer does all of the standard call routing. Because the smart client uses MFC, much of this program layer is standard code. Above this layer are the file upload manager, the executive layer, and the OS htmlCtrl service. The htmlCtrl service is a service provided by the PPC 2002 operating system. It provides an HTML renderer based on Internet Explorer 5. It also provides hooks to notify programs of several key page events, including: page loaded, page complete,
meta tag encountered, and link followed. The executive layer listens for these events and passes the relevant ones to the appropriate message handler. It also acts as an intermediary between the plug-in modules and the rest of the program. In addition, when the IR login feature is enabled, it manages user login via the NGMC IR login component. Finally, the file upload manager implements the FTP protocol to upload files to the server.

4.1. Executive Layer

The executive layer of the smart client mediates communication between the other components. When the smart client is loaded on a device, it is responsible for telling the htmlCtrl to log in. For smart clients with the IR login function enabled, this is done by using the IR module to get the appropriate information, and sending it to the server (the login function, including IR, is explained in greater detail later in this chapter). Once the user is logged in, htmlCtrl an enhanced web browser.

For the functions that do not require special client side components, htmlCtrl follows the links sent by the NGMC web application. However, for special functions, the NGMC browser sends messages in HTML meta tags. When it encounters these meta tags, htmlCtrl notifies the executive layer by sending a WM_NOTIFY message to it. The Executive parses the parameters of the WM_NOTIFY message, and it passes the message to the appropriate message handler (message handlers are also discussed later in this chapter). The message handler, then, is responsible for performing whatever actions are necessary. The executive is also responsible for opening its own TCP connection to the server and checking for alerts. The alert check runs periodically, and important information is forwarded to the appropriate components (see Section 6.3).
4.2. **Message Handlers**

Message handlers are pluggable components that the smart client uses to process a message from the NGMC server. When a message handler instance is created, it registers itself with the executive. The executive maintains a list of message handlers and their "targets", which it uses to find the appropriate handler for a given message.

Upon a page request, the NGMC server sends a message in the body of a returned HTML page as a meta tag (see Figure 9). When htmlCtrl encounters a meta tag, it notifies the executive layer, which finds the appropriate message handler (if one exists) and forwards the message to the handler. When a message handler receives a message, the message is in the form of a NM_HTMLVIEW structure. This structure contains all of the information necessary to identify the event that spurred the message (in this case, the event is encountering a meta tag). It also contains target and data fields, which store contents of the meta tag. This structure is shown in Figure 10.

![Figure 9 The syntax of the meta tag read by a message handler](image)

![Figure 10 The NM_HTMLVIEW structure used by message handlers](image)
The `http-equiv` parameter of the meta tag is placed in the `target` field, and the `content` parameter of the meta tag is placed in the `data` field (see meta tag syntax, Figure 9). The message handler first looks at the value of the `target` parameter to ensure that the message was routed correctly. If the `target` is correct, it parses the value of the `data` field. The structure of the value of the `data` field depends on the message handler used. Specific message handlers that were implemented for this version of the smart client are discussed later in this chapter.

Message handlers have access to all the resources available to the executive, including the OS GUI and network APIs as well as htmlCtrl, the file upload manager, and any smart client plug-ins.
4.3. Alert Handlers

An alert handler is also a pluggable component which connects to the executive; it differs from a message handler, however, in that it may be called anytime, and not only when the htmlCtrl is activated. An alert handler is instantiated just like a message handler. It specifies a "target", which the executive uses to determine which alert handler to call upon receiving a given alert. The executive then begins a timer thread which it uses to

```
procedure check_alert() begin
  h := open_http(server, alert_script_url);
  while (line := http_read(h)) begin
    (target, parameter) := parse(line);
    procedure handler := table_lookup(ALERT_HANDLERS, target);
    handler(parameter);
  end while
end procedure

  t := timer_create( 5_SECOND_TICK_INTERVAL );
set_tick_procedure(t, procedure check_alert);
```

Figure 11 Pseudo-code for the alert check mechanism of the executive

look for new alerts from the server. The timer is set to tick at intervals of about five seconds, and on each tick it connects over HTTP to a CGI script on the NGMC server and requests a list of new alerts. Alerts are returned by the server in the form of an alert target and parameter. The smart client executive parses the alert and uses the target to determine the appropriate alert handler to execute from its list of alert handlers. It passes the parameter to the alert handler, which the handler uses to perform the appropriate task. Once the handler performs its task, control is returned to the executive, which returns to its timer loop.
The five second tick interval was chosen in order to ensure that for a class with approximately 100 simultaneous users, the number of requests would not overflow the server. This may be adjusted in practice, or set to a random interval in order to better spread requests over time. Note that unlike ActiveClass, which uses basic HTML and thus must reload the entire page each time, NGMC only has to get the alerts, which are much shorter and thus can be downloaded more frequently using the same bandwidth. ActiveClass, because it requires constant reloads of a lot of data, is a very heavy load on the network and is not very scalable. NGMC differs in this respect, since it does not need to get a large amount of data, so it does not clog the network, and thus is extremely scalable.

4.4. Plug-ins
Smart client plug-ins connect directly to the executive layer. Because they may need to access any components of the OS or the of client itself, they are given direct handles to the program layer (from which they have access to all modules in the client). They also have direct access to the OS.

The plug-in API is fairly straightforward. Each plug-in goes through a five step life cycle. This is shown in Table 1. When a smart client is started, the executive loads all of the available plug-ins. The plugin_init event is triggered on each of them to tell them to initialize. Once the user has logged in, the plugin_start event is triggered. The plug-in is passed user information, including session ID. When the user logs off, the plugin_stop event is triggered, so that the plug-in can remove all session information. Finally, when the smart client application completes execution, the plugin_unload event is triggered, and the plug-in can then perform cleanup. This is normally used to de-allocate resources, close connections, or free devices.
Table 1 The lifecycle of a smart client plug-in

<table>
<thead>
<tr>
<th>Event</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>plugin_init</td>
<td>Called when the smart client program begins execution. This is normally used to load initialization data, to connect to a system device, or to log in to some service on the network.</td>
</tr>
<tr>
<td>plugin_start</td>
<td>Called once the smart client program is running and the executive has successfully logged the user in.</td>
</tr>
<tr>
<td>plugin_alert</td>
<td>(Optional) Called when the executive receives an alert. Alerts are sent to all plug-ins for processing before they are processed by the executive itself.</td>
</tr>
<tr>
<td>plugin_stop</td>
<td>Called when the smart client is running but the user has logged off (see the security section of this chapter for more information).</td>
</tr>
<tr>
<td>plugin_unload</td>
<td>Called right before the smart client program completes execution. This is normally used to de-allocate resources, close connections, or free devices.</td>
</tr>
</tbody>
</table>

There is one additional important part of the plug-in life cycle: the plugin_alert event. This event may be triggered at any time between the plugin_start event and the plugin_stop event; it may be triggered any number of times (including zero). As soon as the executive receives an alert (alerts are explained in depth later in this chapter), it forwards the alert to all of the plug-ins for processing. If all plug-ins decline to process the alert, the executive processes it and continues.

Note that the plugin_alert event is not triggered when a meta tag message is received. If the plug-in wishes to be notified of meta tag messages, it must register a corresponding message handler with the executive. This message handler is responsible for notifying its plug-in whenever a pertinent message is received.

4.4.1. An Example Plug-in: Elevator Speech Recorder
The Elevator speech recorder is a plug-in that was constructed for the first field test of the NGMC system. The purpose of it is to save time in class by allowing students to do one of
the class projects, an oral "elevator" speech, outside of class. Using their handheld and the smart client, they would be able to record their 30-second speech and then automatically upload it for grading by the professor.

This feature was implemented in the smart client as a plug-in. On `plugin_init`, it allocates necessary resources and connects to the WAV audio in service provided by the PPC 2002 operating system. Because almost all PPC 2002 handhelds are built with microphones, no special hardware was required. The plug-in sits idle until it receives an alert from the server, telling it that the student requested to begin recording. It then goes into record mode, recording the speech until the user clicks stop. Once the user is finished, the FUM (see Section 4.5) is used to upload the WAV file to the server over FTP, where the professor can then listen to it and grade the speech.

### 4.5. File Upload Manager

The file upload manager (FUM) provides a service that can be used by any module to upload a file. For security reasons, it is only allowed to upload to the NGMC to which the executive is connected in this version of the smart client. This means that an FTP server must be installed on the same machine as the web server/NGMC web application. We plan to change this in future releases, but for now, this configuration provides the best combination of simplicity and security.
The FUM uses a simplified version of the FTP protocol to perform uploads. We originally implemented the FUM using only active FTP. We quickly learned, however, that this would not be viable in many situations (mostly because of NATs and firewalls, but also because some FTP servers simply do not implement active FTP). We therefore implemented both active and passive FTP, but restricted the commands to those that were necessary for minimal upload functionality.

When a plug-in or message handler wishes to upload a file (the executive or htmlCtrl never need to upload files on their own behalf), it makes a request to the FUM directly. The FUM can either be passed a file to upload, or the user can be given a file dialog box to select the file to upload. The FUM uploads in binary mode only, so any files that are uploaded will be exactly the same on the server as they are on the client. Because of this, the FUM is not recommended for uploading text files.
The FUM returns an error code when it exits. If the upload was successful, it returns zero (for no error). If not, an error code describing the particular error is returned. All of the error code constants are defined at the top of the FUM header file. In either case, control is returned to the calling module, which can then determine how to continue. The file path of the uploaded file is returned to the caller as well so that it can, if necessary, notify the server that the file was uploaded and deliver any other relevant information.
Chapter 5. Security and Privacy

The smart client takes elaborate measures in order to ensure security and privacy while remaining easy to use. A user is required to log in to the NGMC system whenever he/she wants to use it. Since the system is wireless, it is relatively simple to sniff the network traffic. Also, the system is designed to use the MIT network, which is inherently insecure. Therefore, we found it necessary to implement an encryption policy which would certify that any data that may be sensitive is not sent as clear text. The privacy protection policy, meanwhile, had to be able to certify that no personal data and no information identifying a user's location ever leaves the user's device (unless the user chooses to distribute it).

Since there are two ways in which a user may log in—with or without the infrared (IR) login enhancement—this section is split in two. The first part looks at how security and privacy are maintained in a basic scenario, without the IR login enhancement. The second part explains the adjustments to security and privacy functionality due to the IR login enhancement [SW96].

5.1. A Basic Scenario

In a basic (non IR) scenario, security and privacy are mostly maintained by the standard HTTP security protocols built into the web server and htmlCtrl. When a user begins an NGMC session in this scenario, the user starts the smart client and is taken to the login page. The login page does an HTTPS post to a secure script on the server, so once the user enters login information (username and password) and clicks to submit, the secure connection is established. All subsequent transactions are via HTTPS, a standard security protocol for
web-based systems that is implemented in htmlCtrl and most web servers (including Apache, the server that the current version of the NGMC web application runs on).

Throughout the NGMC session, the smart client maintains a session ID, or SID, which it uses to identify the user and session to the server (see Figure 13). The SID is a nonce for each session; that is, once a user logs out, the SID is never used again by the same user or any other user. The smart client begins execution with the value of the SID set to null. Once the user enters his/her information on the login page, each subsequent page received from the NGMC web application on the server contains a meta tag message with the SID. For this message, target value is “sid” and the data value is the SID of this session. The SID should never change during a user's interaction with the NGMC system. If it does, either an error has occurred or security was breached, so the client suspends execution.

The SID must remain private and secure, since it is basically used as both a username and a password for the duration of a session. For the most part, the SID is only needed inside the htmlCtrl, so security is ensured: the server sends the SID to the client as a field in a HTTPS encrypted page, and the htmlCtrl sends requests to the server as HTTPS POSTs. However, when the additional functionality of the smart client is used—such as the shortcut buttons at the bottom of the window, the alerts, or the class plug-ins—the smart client must connect to the server independent of htmlCtrl.
In designing for these cases, we had two options. First, we could have implemented HTTPS in the executive layer. Although this would have been the most secure method, time and resource constraints made it infeasible. Instead, we chose to take a much simpler route. Since the server has a list of active SIDs at all times, it is possible to use a basic `crypt()` scheme: The smart client runs the `crypt()` algorithm on the SID that it plans to send; call this SID₀. SID₀ is then sent over the insecure network to the server, which recognizes it as an encrypted SID. It then looks in its table of active SIDs for one that matches SID₀. If it finds one, the one that it finds is used. This scheme is not as secure as HTTPS and may, in an extremely unlikely case, introduce bugs. It is also not very scalable, since the server must check all of its SIDs to find the correct one. Nevertheless, it works decently for this version of the smart client. In future versions, we plan to move to HTTPS.

Finally, when a user logs out, the SID is destroyed. If the user clicks the logout link or uses the logout shortcut button in the smart client, the smart client notifies the server and the server clears the SID. It is also possible, however, for the session to expire on the server because of inactivity. In this case, the smart client is able to detect the expiration because it will receive an alert that the SID is no longer valid. Upon receiving this message, the smart client destroys the SID that it has cached and clears the internal state to prepare for a different login.
5.2. **IR Login**

The IR login system allows users to login to an NGMC classroom quickly and easily, minimizing the entry of text using the cumbersome stylus. In designing an automated login system, we had several options; still, the only really practical one was using IR with another device. Unfortunately, IR login has some drawbacks and adds a great deal of complexity to the system. This section will look at the design choices that we made in building the IR login system, and how the system modifies the security architecture of NGMC as a whole. For more information on IR and the IrDA specification, see [Tan01].

5.2.1. **Login System Design Choices**

The IR login system was the result of numerous attempts to provide do location mapping that gave users privacy and security while being usable and compatible with the hardware/software on the handheld devices.

Originally, we planned to use MIT’s “cricket” devices to provide the handhelds with the necessary information [Pri01]. These devices use electromagnetic and sonic ranging to determine the closest “beacon” (or transmitter) to the “cricket.” The beacons, without gaining any information about individual crickets, are able to beam their unique identifiers to the crickets near them. In our scheme, each handheld would be attached to a cricket, and beacons would be placed in each of the participating class’ classrooms. When a user entered a classroom with his/her handheld device, the cricket attached to it would automatically recognize which classroom it is in use that to log in.

The problem with using crickets, however, was that crickets are unreliable, bulky, and expensive pieces of hardware that require extremely specialized software to run. It
proved impractical to have users carry crickets around with their handhelds, and to force users to debug them every time they did not function correctly. Moreover, the software to use the crickets was proprietary and used a Java interface. We did not want to force students to load Java and all its supporting libraries onto their handhelds, as this is a difficult process and is prone to error. We therefore chose to abandon the crickets entirely.

We then thought about doing location mapping by using the wireless infrastructure that was already deployed at MIT. Each wireless access point has a distinct MAC address, the handheld must know this MAC address in order to communicate with the access point. We figured that we could write software so for the handheld to determine the access point’s MAC address. This software would then use some sort of look up table that translated a MAC address to a classroom (the table would have had to be entered manually by an administrator). By using only the MAC addresses, we could avoid using specialized hardware, and location information could be attained over any standard 802.11b setup.

Unfortunately, this too was not feasible, since finding any information about the MAC address of an access point through software requires hardware-dependent code. That is, every different network card requires different code to return this information. Moreover, the code would require direct access to the hardware, which would have meant writing PPC 2002 device drivers for each card. This would be too difficult, especially because many network card vendors do not publish detailed enough specifications to do this.

We then decided to try another avenue: use the IR transceiver that is built into PPC 2002 handhelds, and make users swipe their handheld to log in. More specifically, when a user enters a classroom, he points his handheld at the login beacon at the entrance to class. The login beacon sends a class identifier to the handheld, which beeps upon receipt. The
handheld then automatically logs the user into the class on the NGMC system, and the user walks in to the classroom. Since IR and the IrDA API are pre-built into PPC 2002, coding the software would be relatively easy. Also, users would need no additional hardware, and the smart client software could be integrated with the IR software so that users only need to install one program.

5.2.2. IR Login Implementation Details
An IR login requires a handheld and a transmitter, or beacon. The beacon is designed to continually broadcast an identifier so that handhelds can be swiped against it (see Figure 14). We had originally hoped to have a specialized piece of hardware to do this; a small IrDA capable device that just broadcasts a sequence of bytes that we could install in the doorway of each participating classroom. However, we did not have anyone on the team who was capable of building the hardware. Instead, we used an extra handheld device and programmed it as the beacon. The handheld device is placed at the entryway to a classroom and users slide their handhelds in front of it to log in.

When the beacon is installed, it is put into a connect loop. This loop initializes the IrDA device and looks for a connection. The user’s handheld, meanwhile, is constantly in a similar loop, which runs somewhat slower to conserve power. Once the user’s handheld

![Figure 14 The workflow of the IR login](image)
comes within IR sight of the beacon, the beacon performs a handshake and then sends the pre-programmed classroom ID over the IrDA connection. The user's handheld checks the data and returns an acknowledgement to the beacon. It also beeps and flashes its LED to signal that the classroom ID has been received. Note that since the beacon is not connected to any network, this exchange is done without any information being sent to the server.

Once the user’s handheld knows what classroom ID to use on login, the executive attempts to log in and jump to the classroom with the given ID. If the username and password were saved from the last login, these are used, so the user does not have to enter them. If the login was successful, the executive simply jumps to the classroom page specified by the given ID, so the server cannot tell whether the user went to this page manually or automatically. The SID is embedded in the meta tag of the returned page, at which point the usage scenario continues as in the basic case.
Chapter 6. Message/Alert Handlers

As explained earlier, the smart client is capable of receiving and processing messages and alerts from the NGMC server. A message is a piece of information that is embedded in an NGMC page as a meta tag. An alert, meanwhile is a piece of information that the NGMC server sends to a smart client by polling an HTTP script. We have already discussed messages and alerts and how they work in the NGMC system. This section will look at two examples of messages implemented in this version of the smart client: scheduling and contact; in addition it will look at an example alert: the quiz alert.

6.1. Message Handler: Scheduler

The first example of a message handler is the scheduling component of the smart client. The scheduling message handler integrates with the operating system scheduling service for Pocket PC 2002 with the smart client. By doing this, the Pocket Outlook calendar, which has many extended features is directly accessible.

We chose to implement the scheduling message handler this way because we felt that it was necessary that the scheduler be directly integrated with the calendar. This would make it convenient for students to use it, hopefully resulting in its being commonly used. The scheduling message handler is built upon the Pocket Outlook Object Model (POOM) API. With POOM, the smart client has access to all of the features that a user would normally use. For example, the calendar shown in Figure 15. POOM provides C++ interfaces that allow the smart client to connect into these features and send them data or get data from them.
If a user wishes to place an event in the calendar, he uses the NGMC web application and enters the time, date, duration, and other information. He then selects the group of users to whom the event is to be broadcast. Once the event is entered, the next time any one of these users accesses the NGMC application using the smart client, the HTML page returned will contain a message for the smart client to enact the scheduling message handler, which then incorporates the event into the user’s calendar.

![Calendar Image](image)

**Figure 15** The POOM-accessible calendar included in PPC 2002

The scheduling message handler for the smart client is enacted when executive layer encounters an *ngmc-sched* message. As explained earlier, this message would be sent by the NGMC web application by embedding a meta tag in the HTML that is returned when a page is requested (see Figure 9 for the syntax; examples are in the following sections). The parameters to the message are shown in Table 2. Note that parameters are passed in HTML by position, separated by semi-colons.
Table 2 Parameters for ngmc-sched message (in position order)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>subject</td>
<td>The text that appears in the subject line in the Calendar view of Pocket Outlook (in Figure 15, this is “6.001 Exam”)</td>
</tr>
<tr>
<td>cat</td>
<td>The category that this event appears under and can be sorted by</td>
</tr>
<tr>
<td>date_mo</td>
<td>The month when this event is to take place (0-indexed number)</td>
</tr>
<tr>
<td>date_day</td>
<td>The day when this event is to take place (0-indexed number)</td>
</tr>
<tr>
<td>date_year</td>
<td>The year when this event is to take place (4-digit number)</td>
</tr>
<tr>
<td>body</td>
<td>The text that appears as the complete description of the event in the Pocket Outlook detail window.</td>
</tr>
</tbody>
</table>

The scheduling message handler takes these parameters and uses the POOM calendar interface to enter them into the user’s calendar on the handheld.

6.2. Message Handler: Contact Information

The contact information message handler works in a manner similar to the scheduler above. It also uses the POOM API to connect into the PPC user interface so that users can access information from the NGMC system directly. When the smart client receives contact information it enters it into the Contacts list in Pocket Outlook so that the information is accessible to users with the push of a button. An example of the PPC interface to the contact information entered by the smart client is in Figure 16. The contacts interface in PPC 2002.
Upon registration, every user can choose whether to make his contact information public. If he does, the information is sent to other users the next time they use the NGMC system with the smart client. It is then automatically incorporated into the Pocket Outlook Contacts application by the contact information message handler of the smart client.

The contact information message handler for the smart client is enacted when executive layer encounters an `ngmc-contact` message. As explained earlier, this message would be sent by the NGMC web application by embedding a meta tag in the HTML that is returned when a page is requested. The parameters to the message are shown in Table 3. Note that parameters are passed in HTML by position, separated by semi-colons.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>fname</code></td>
<td>The first name of the user</td>
</tr>
<tr>
<td><strong>iname</strong></td>
<td>The last name of the user</td>
</tr>
<tr>
<td><strong>email</strong></td>
<td>The email address of the user</td>
</tr>
<tr>
<td><strong>homephone</strong></td>
<td>The home phone number of the user</td>
</tr>
<tr>
<td><strong>busphone</strong></td>
<td>The business phone number of the user</td>
</tr>
<tr>
<td><strong>cellphone</strong></td>
<td>The cell phone number of the user</td>
</tr>
<tr>
<td><strong>addrstreet</strong></td>
<td>The street address of the user</td>
</tr>
<tr>
<td><strong>addrcity</strong></td>
<td>The city of the user</td>
</tr>
<tr>
<td><strong>addrstate</strong></td>
<td>The state that the user is in</td>
</tr>
<tr>
<td><strong>addrzip</strong></td>
<td>The zip code of the user</td>
</tr>
</tbody>
</table>

The contact information message handler takes these parameters and uses the POOM contacts interface to enter them on the handheld.

### 6.3. Alert Handler: Quiz Alert
We now look at a very simple example of an alert handler. Unlike the message handler, an alert handler can be activated by the smart client executive at any time, and is not limited to being called when an HTML page is loaded. When the server sends an alert, the smart client uses the appropriate alert handler to process it. The example alert that this section will focus on is the quiz alert.

When the smart client executive receives a quiz alert—that is, an alert with the target "goto:quiz" (see the section on alert handlers)—it executes the quiz alert handler. The parameter of the goto:quiz alert is the URL of the quiz. When the alert is received, the quiz alert handler looks at the URL. It then tells htmlCtrl to display that URL. The user can then take the quiz.
Chapter 7. Implementation Guide
Depending on the level of customization necessary for a particular class, the NGMC system can require varying amounts of setup and support work. For organizations simply adding a class to a current installation, very little setup work is necessary, and most of this work is done on the server side. For organizations creating an entirely new installation of NGMC, there is work required on the server side, but there is also work required to prepare the smart clients for students, as well as infrastructure work to make sure the system functions smoothly. This section is a user's guide for setup, use, and support of the smart client. A similar guide for the server is out of the scope of this paper, but is available in [Cog03].

7.1. Building from Source
The smart client source is located on our main server: ngrcserv.mit.edu. Most of the client side software for NGMC is written using Microsoft Embedded Visual C++ 3.0 with the Pocket PC (PPC) 2002 SDK.\(^5\) To provide an open development environment, the source code is available via Microsoft Visual SourceSafe, as well as for download. This section describes how to compile the source for a specific network environment and prepare it for distribution.

7.1.1. Development Source Tree
The smart client software distribution is organized into the following directory structure:

Table 4 The smart client directory structure

<table>
<thead>
<tr>
<th>Directory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ngmc_client/</td>
<td>The base NGMC directory. Contains all of the client side software and source code.</td>
</tr>
<tr>
<td>ngmc_client/smart_client/</td>
<td>The directory for all components of the smart client. Includes all of the C++ files, header files, EMVC project files, and workspace files.</td>
</tr>
<tr>
<td>ngmc_client/smart_client/ngmc/</td>
<td>The HTML resources (including start page) that are embedded in the smart client.</td>
</tr>
<tr>
<td>ngmc_client/smart_client/res/</td>
<td>VC++ build resources for the smart client.</td>
</tr>
<tr>
<td>ngmc_client/ngmc_ir_transmitter/</td>
<td>The directory for all components of the IR beacon used for automated IR login in class.</td>
</tr>
</tbody>
</table>

The project files are in the appropriate directories. This directory structure is also intact on the VSS tree which can be connected to on ngmcserv.mit.edu.

To build the smart client, open the workspace file `ngmcbrowser.vcw`. This will open the environment shown below in EMVC:
Although EMVC allows the project to be built for Windows CE 2.x as well as Pocket PC 2002, some of dynamic link libraries required for it are not available in CE 2.x. Make sure the build target is PPC 2002. Also, as of the writing of this document, the only applicable target processor is ARM (XScale is ARM compatible). Make sure that the target is set to ARM. Also, before building, several libraries must be available: commctrl.lib, coredll.lib, aygshell.lib, htmlview.lib, winsock.lib, ceshell.lib, osptk.lib, shlwapi.lib, cemapi.lib.
7.1.2. Pre-compile Constants

In order to make the smart client executable more self-contained and secure, all of the HTML and other files that it needs are compiled into the file. In addition, the server addresses and paths are compiled in as headers instead of using a configuration file. Therefore, if creating a new install of the NGMC system, set the following constants as needed:

Table 5 Smart client pre-compile constants

<table>
<thead>
<tr>
<th>Pre-compile Constant</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NGMC_LOGIN_URL</td>
<td>The URL of the NGMC web application. This is the page that is first shown when a user starts the smart client.</td>
</tr>
<tr>
<td>NGMC_IRLOGIN_URL</td>
<td>The URL to post login information when logging in via IR login.</td>
</tr>
<tr>
<td>NGMC_UPDATE_CHECK_URL</td>
<td>The URL to which the smart client connects when looking for new alerts.</td>
</tr>
<tr>
<td>NGMC_FTP_SERVER</td>
<td>The IP address or hostname of the FTP server that the smart client connects to for doing FTP uploads.</td>
</tr>
</tbody>
</table>

Once these are set, the project can be built. The output will be called ngmcbrowser.exe. The next section will show how to distribute and install this application on handhelds.

To build the IR transmitter for use with the IR login system, open the project in ngmc_ir_transmitter/ and compile it. It can be compiled for PPC2002 or Windows CE 2.x on any supported processor. Once it is built, install it on a separate device (we used a different handheld—an HP Jornada—with CE 2.x) and place it at the classroom entrance. Make sure that it is connected to a constant source of power; batteries will not last long enough for more than a few days of operation.
7.2. Installation and Setup

Once the smart client is built and an executable file is available, it needs to be placed on appropriate devices and distributed to class members. The first step to doing this is to select an appropriate PPC 2002 handheld. We used the Toshiba e740 because it was fairly inexpensive and had integrated 802.11b; however, the quality of the Toshibas was at best, sub-par. They have since been recalled.

Good handhelds to use are Compaq iPAQs with a BackPAQ (although these are bulky, see Figure 18), and HP Jornadas with Symbol CompactFlash 802.11b cards.

When setting up smart clients for the classroom, a common problem that users experience is network failure. The MIT 802.11b network requires MAC addresses to be registered with MIT Information Systems before wireless is activated. This caused a number of problems for users, who were confused about this process. Even when everyone was properly registered, however, users still had intermittent network problems, especially with DNS resolution and DHCP connection. Because proper network operation was vital for testing the system, we chose to place a separate 802.11b access point and NAT in the classroom. With this in place, all users could connect without trouble. Adding a separate access point is an extreme solution: these problems can be avoided by pre-configuring the handhelds and testing them before distribution.

Figure 18 An early version of the NGMC system on a Compaq iPAQ
There are several options for distributing the software. For new installations of the NGMC system, the best way to do the install is using the EMVC built-in installer and to install and test each copy before distributing the handhelds. This is the safest and most straightforward way, but depending on the number of users, this way may be impractical. Another way—the way we chose—is to place the executable on the web and have users download and install the software themselves. We experienced some problems with users not being able to install the executable in the right place, or not being able to find it once installed.

7.3. Basic Usage
The following is a very brief guide on how to use the smart client. This guide does not explain how to use the features of NGMC, as that is out of the scope of this document. For a complete guide for students or professor, see [Cog03].

Figure 19 The smart client user interface with buttons highlighted
Figure 19 shows the smart client user interface. The central pane is where most of the interaction happens. The links and what they do are fairly self-explanatory. The bottom buttons provide quick access to the most common features. The Home button takes the user to the main user home page—the one shown in Figure 19. The Ask a Question button jumps to the page on which a user can enter a question. The Upload File button shows a file dialog box from which the user can select a file to upload. The Jump to Menu provides a dynamic list of features that the user can go to. This list is dependent on the current state of the user and the system.

Chapter 8. Implementation and Results

When using NGMC in the classroom, we had interesting results. We found that students and professors were definitely intrigued by the features available in the prototype. Moreover, they were happy to try the system and use it in different ways inside and out of class. However, we also found that there is room for improvement, and that some of the edges need to be smoothed out before the system can be used reliably in a large class. Over the course of the term, we recorded a large amount of data using usage logs. These logs recorded which functions of the system were used the most often, which type of clients were preferred, when the system was used. These logs provided a good basis for understanding how users reacted to the system. In addition, we gave a survey to users at the end of the term to see what their thoughts were. This chapter looks at the results that we obtained.

Because this is an EECS thesis, it does very little pedagogical analysis. The data for this is available to anyone who wishes to analyze the pedagogical effectiveness of the system.
in the future; however, it is not the main goal of the thesis. We hope that when the system is used in the future in larger classes, there will be more data for analysis, and true statistical analysis can be done, including any statistically significance testing. Here, however, we mostly look at raw data to get a general picture of the outcome.

8.1. Usage Patterns
Throughout the duration of the class, we recorded several aspects of a user’s session with NGMC. With this information we were able to notice several patterns in several different categories. This section will look at these patterns; the following section, which looks at the user survey, will attempt to correlate the results here with user responses in the survey.

8.1.1. Usage by Feature
One very useful data point we discovered was usage by feature. That is, the percent of time spent on NGMC using each of the major features. For this analysis, we grouped the features of NGMC into the following: Quiz means any mini-quizzes or surveys given by the lecturer to the students. Course means any course functions, such as message of the day and course information. Session means any session information, such as management and distribution of contacts or events. Media means any upload or recording of media, such as audio speeches. Question means student questions sent to the lecturer or TA. And portal means any other general NGMC functions, such as login.

Given these groupings, we ran a script on the server logs to classify each transaction made by a user (the script is supplied in Appendix A). The results generated by the script are plotted in Figure 20:
As the chart shows, the most commonly used feature—by far—was the quiz. This was apparent in class as well, as the professor offered surveys or mini-quizzes on most of the lecture material. Surprisingly, students did not ask too many questions using the NGMC system. This may have resulted from the small size of the class, which made it unnecessary to use the question feature. The rest of the features were used as expected. The portal and session features got a lot of use because they are essential to login and use of the system. Media was used very little, since only one assignment required it. And course was used only a small amount for MOTD and other minor features.

8.1.2. Usage by Client Type
Another pattern that we looked at was usage by client type. That is, since NGMC can be used by the smart client on the Pocket PC or by a normal web browser on any platform, it is
useful—especially for the purposes of this thesis—to determine which was more used. We found that the smart client on the Pocket PC was overwhelmingly more often used than a web browser on any platform, as shown in Figure 21.

![Usage by Client Type](image)

**Figure 21** The percentage if total use by the client type (smart client or other)

This finding is worthy of note because it shows that the smart client proved useful for users in class. While the results may be skewed because students were specifically shown how to use the smart client, they had the option of not using it and using the web browser on the Pocket PC in class.

### 8.1.3. Usage by Time

Another pattern that appeared in the data was that there was a great spike in usage during the class hour (3:00 PM – 4:30 PM). This is not surprising, since NGMC is intended to be used in class. What is surprising, however, is that there were a noticeable number of people using the system at other times as well. A graph of usage by hour is shown in Figure 22.
The usage patterns shown resemble common usage patterns for an MIT Internet resource [Pen02]. There is a large number of hits late at night, a smaller number during the day, and an even smaller number early in the morning. This indicates that students use the system outside of class, most likely for contact information, schedules, and lecture outlines.

8.2. Survey Results
As explained earlier, another part of the evaluation process was to have users fill out a survey about their experience with NGMC. The survey contained questions divided into three categories: ease of use, usefulness, and whether users would like to see something in the future. The survey questions were all on a 1-5 scale, with a space for comments as well (the complete survey is in Appendix B). We found that overall, users found the system useful and relatively easy to use; however, there were some flaws that they noted and changes that could be made.
There were several external issues that may have affected the results of the survey, and these issues should be taken into account when interpreting the results:

First, there were many problems with the connection to MIT’s wireless network. MIT requires that special procedures be taken when connecting to its wireless DHCP server. These procedures are not always easy to follow, and they require a fair amount of legwork and understanding on the part of the students. As we realized later in the term, the best way around this issue was to bring a wireless NAT to class and use that instead of the MIT network; however, this occurred too late, and students had many problems connecting to the network.

Another external issue that may have affected the results was the instability of the Toshiba e740 handhelds. We bought these machines in their first iteration, and they turned out to function poorly. We experienced countless problems with the network cards on these devices, which made them work intermittently. Worse, many simply did not work out of the box, or stopped working soon after we purchased them. We returned these under warranty, but this may have caused frustration to students. We found out later that the devices were recalled for numerous defects.

Taking these issues into account, we may now look at the survey results and analyze them. The most general results of interest are the overall ratings that users gave to the system. The graphs in Figure 23 and Figure 24 provide a look at users’ overall opinions of the system. Figure 23 shows a histogram of the overall ease of use rating that users gave NGMC on a 1-5 scale, where 1 is very easy and 5 is very hard. The average rating that users gave was a 2.8. This is a fairly average result, showing that students found some things that were easy and others that were difficult. In the next several subsections we will look at which were which in detail.
Figure 23 Overall ease of use histogram (1 is least difficult, 5 is most difficult)

Users gave a similar rating for usefulness, as shown in Figure 24. The average rating was a 2.4 out of 5, where 1 is most useful and 5 is least useful.

Figure 24 Overall usability histogram (1 is most usable, 5 is least usable)

Users who left comments about the usefulness of the system mostly complained about the difficulty that they had in connecting to the wireless network and using the handhelds in
general, which made NGMC less practical for them. They also noted that it was difficult to use the handhelds to write text, which is required to use the ask question feature and other features. Finally, many of them commented that although they could easily see the benefit of NGMC in a larger classroom, 21W.785, having fewer than 20 students, was not large enough to necessitate it. For instance, one person remarked, “it seems very useful, and I could definitely use it in a class like 6.001, but this class just has too few students to make this really worthwhile.”

8.2.1. Usability of Login
IR login was met with mixed reviews by users. While they found it useful when it worked on the first try, it often required several swipes before the data was successfully transferred and they could use the system. As Figure 25 demonstrates, student gave it a difficulty rating of average, or 2.9 out of 5.

Figure 25 IR login ease of use histogram (1 is least difficult, 5 is most difficult)
Most users who commented about IR login said that it was a great idea, but the IR function needed to work more reliably before it really helped. We are still trying to determine whether the IR problems are a result of the Toshiba hardware or the controller software.

Users found the manual login on the smart client more usable, giving it an average rating of 2.1 out of 5 (Figure 26). However, several commented that it was more cumbersome than the IR login, especially since it involved writing passwords on the handheld, which can be very error prone.

![Manual login with smart client](image)

*Figure 26 Manual login ease of use histogram (1 is least difficult, 5 is most difficult)*

### 8.2.2. Usability of Smart Client Shortcut Functions

The shortcut features of the smart client were not a key feature, but they were intended to make navigating the system a lot easier, by allowing users quick access to the most used functions. The smart client buttons—the buttons on the bottom left of the smart client window—provide quick access to the user home, the ask question facility, and the file upload tool. Users found these buttons useful, giving them an average rating of 2.1 out of 5 (see Figure 27). Users had some complaints that early in the term, the icons on the buttons
were not very intuitive, and although this problem was fixed, it caused them to avoid using the buttons later in the term.

![Smart client buttons histogram](image)

**Figure 27** Smart client button ease of use histogram (1 is least difficult, 5 is most difficult)

Students found the quick menu, the menu which contains drop-down access to most common NGMC features, very useful. This is evident from the histogram in Figure 28:

![Smart client quick menu histogram](image)

**Figure 28** Quick menu ease of use histogram (1 is least difficult, 5 is most difficult)
Users gave the smart client an average rating of 1.8 out of 5, and commenting that the menu was a very helpful quick navigation tool, and it made using the handheld while listening to lecture much easier.

8.2.3. Usability of the File Upload System

The file upload system tended to be one of the most difficult parts of the system to use. Due to a variety of problems, users seemed to be frustrated with it and thus avoided using it as much as possible. As Figure 29 demonstrates, users gave it a rating of 3.2 of 5—the lowest rating given.

In their comments, users explained that file uploads often did not work for them. In fact, during the term we got several questions from users trying to upload files. Although we tried to make this process as easy as possible, there were several obstacles. The wireless adapters (the hardware on the Toshiba handheld) were less than reliable, often losing the network connection during transfer, or right as users were trying to start the upload. The
NGMC server and the FTP server running on it were also somewhat to blame, since the server often dropped FTP connections. Finally, there were some bugs in our FTP upload implementation on the smart client, which made transfers slower than they would be using a standard FTP program. This is one of the features that we plan to overhaul in the next release of the software.

8.2.4. Usability of the Ask Question Facility

Users found the ask question facility very easy to use, giving it an average rating of 1.6 out of 5 (see Figure 30). Most commented that they found it very easy and intuitive. Some, however, noted that it is difficult to enter a question on the handheld because the handwriting recognition in Pocket PC 2002 is not very accurate. They added that this made it somewhat difficult for them to pay attention while asking a question.

![Ask a question facility](image)

Figure 30 Ask question ease of use histogram (1 is least difficult, 5 is most difficult)
Also, because of the small size of 21W.785, some commented that it was easier to simply raise their hands instead of using the handheld. We expect the results to be different in a larger class.

8.2.5. **Usability of Quizzes, Surveys and Mini-quizzes**

The quiz feature is another one of the most important. Users found this feature easy to use on the smart client, giving it an average rating of 1.8 out of 5, as Figure 31 demonstrates. They especially found the fact that quizzes automatically pop up when the lecturer launches them to be useful.

![Answer surveys/mini-quizzes](image)

*Figure 31 Survey, mini-quiz ease of use histogram (1 is least difficult, 5 is most difficult)*

They found it very easy to follow along with lecture, and use the quiz when it pops up, so that they do not need to navigate through a bunch of pages to get to the quiz that they need to take. Quiz creation was found to be somewhat more difficult, but that is out of the scope of this thesis, since it does not involve the smart client.
8.2.6. **Usability of the Calendar/Events System**

Users also found the calendar and events system relatively easy to use. They had little trouble accessing event information and incorporating it into their schedules. They seemed to enjoy this feature a lot, and they were able to make good use of it. As such, they gave it an average ease of use rating of 2.1 out of 5 (see Figure 32).

![Events access ease of use histogram](image)

*Figure 32 Events access ease of use histogram (1 is least difficult, 5 is most)*

Ease of creating an event is out of the scope of this document, since it is mostly done in the NGMC web application on the server.

8.2.7. **Usability of the Contact Information Feature**

The contact information feature was also very easy to use according to the survey. Users had no trouble entering their information and marking it to be made public. They also had no trouble accessing others' information, which was automatically incorporated into their address book when they used the feature. As one can see from the histogram in Figure 33, gave it an average rating of 1.8 out of 5.
Some did note, however, that they were reluctant to make their information available. They did not thoroughly understand the permissions system, and thus they were unsure what information would be displayed and who could see it. Once we explained to them that information would only be available to their group members, and they need only enter the information that they wanted to make public, they were more willing to use this feature.

8.3. Overall User Opinion

Overall, users looked at NGMC favorably, and most said that they would use it again in other classes if they could. They found the system usable and saw that it could be very useful in larger classes (see Section 8.2).

When asked whether they would use the system again, users gave NGMC a rating of 2.5 out of 5, where 1 is definitely and 5 is definitely not. They said that they liked the way that it encouraged in-class discussion and participation. But they also had some criticisms, as noted in the previous sections. A histogram that displays the users’ ratings appears in Figure 34.
Even more users felt that they would like to see NGMC used in other classes. The histogram in Figure 35 shows that users gave NGMC a rating of 2.4 out of 5 on this topic. Some commented that it would be better suited for a larger class and that they would rather use it in a class like MIT's 8.01 or 18.01, which both contain several hundred students. Since 21W.785 was a fairly small class, the system was not as well suited to it and it would be more worthwhile when used in a class where students could not participate aloud during class.
There were mixed opinions about whether users would like to see the PDAs used more in class. Several thought it was a good idea, while others thought it was a very bad idea. Those who thought the latter cited some of the issues mentioned in the last section, especially that writing on the handheld during class is often distracting. Users gave this proposition a rating of 2.7 out of 5, but the results were more spread out than others, as seen in Figure 36.
The results were also mixed when users were asked whether they would like to see the handhelds used more outside of class. Some gave a resounding yes, saying that it made assignments easier to keep track of, and allowed them to do more fun projects, such as the speeches that were assigned in 21W.785. Those who did not want to use the handhelds outside of class said that they made that choice mostly because they did not want additional work to do.
Figure 37 Answer to “would you like to use the PDAs more outside class” histogram (1 is definitely, 5 is definitely not)

The users gave the idea of using the handhelds outside of class a rating of 3.4 out of 5, meaning that they tended to not like it, but the results were spread out with some strongly advocating it and others against it. This is visible in the histogram in Figure 37.

### 8.4. Further Directions for Analysis

As noted at the beginning of this section, this thesis does not do pedagogical analysis of the data in detail. It is very important to do this sort of analysis before the system is actually put into place. However, due to the small class size and the fact that the system is still in its early version of development, we leave this as future work (see Section 9.1).
Chapter 9. Future Work

We designed the NGMC project as a multi-year project to build a new type of educational experience. What we have created takes a step in that direction but is nowhere near complete in achieving the goal. The overarching goal of the project was something that encompasses the whole classroom and takes advantage of all the technology available there.

This is evident from the publicity materials that we created for the iCampus committee:

Imagine a classroom where the professor's lecture slides show up on screen as soon as she enters. Where students' homework grades display in the palm of their hand. Where everyone participates in lecture by taking mini-quizzes; and where the answers are tallied and feedback appears right before their eyes. Imagine a classroom where assignment dates are automatically integrated into students' calendars. And faculty contact information is entered directly into their address books. Imagine a classroom where study groups are organized at the touch of a button. Where lecture text is synchronized with slides for hearing impaired students. And where students can ask questions without disturbing the flow of lecture. This is the vision of the Next Generation Mobile Classroom.6

The smart client that we developed is basically a starting point from which we hope to build an entire system of this type, which we hope to be able to use in all classes at MIT, so students need only carry their handhelds to class and be able to do everything that they do today while having all the benefits explained above.

In this chapter, we will take a brief look at the possible future directions of the NGMC project. Because we already have plans to use it in another MIT course in Spring 2003, some of this work is currently underway.

---

9.1. Complete Pedagogical Analysis

Being an EECS thesis, this thesis did not go into much detail in analyzing the usage of the system. The data for this is available to anyone who wishes to analyze the pedagogical effectiveness of the system in the future; however, it is not the main goal of the thesis. We hope that when the system is used in the future in larger classes, there will be more data for analysis, and true statistical analysis can be done, including any statistically significance testing.

It is also very important to study how the Learning Cube and different modes of learning play a role in the workings of NGMC. To do this, one can use a Pedagogical Effectiveness Index, or PEI, which was developed by Dr. Sonwalkar [Son02] to test a system such as this. PEI essentially looks at several properties of a method of education and comes up with a score which can be used to measure its effectiveness. Since Dr. Sonwalkar has done a great deal of analysis using PEI on in-lecture education and web-based education, with the PEI, it will be possible to compare NGMC with a traditional lecture and have a quantitative measure of the results. This analysis can provide invaluable data when complete [Son02C].

For this iteration of NGMC, however, we chose not to do pedagogical analysis for two reasons: First, we felt that we did not have enough data or a large enough sample size to do a study such as this. There would need to be more students, and we would have to do a more through pedagogical analysis of how the students learned. Being EECS students, felt that we were not qualified enough to do this, and would defer an analysis like this to someone with expertise in that field, such as Dr. Sonwalkar. Second, the current version of NGMC has bugs and usability issues, as described in the previous section. It is important
that these be addressed before doing such a study, so that the pedagogical analysis is not biased by these external factors.

9.2. Enhanced Security
One of our original design goals, as mentioned earlier, was to make the system secure enough to allow grades and other personal information to be distributed over it. In building it, we paid a lot of attention to protecting anonymity, so that students do not feel that the system is spying on them. However, we left much of the security issues to be done in the future.

In order to enhance the security of NGMC, the first thing we would do is make it run on an HTTPS web application server. Currently, since NGMC is a wireless application that uses all standard protocols over HTTP, a potential hacker has several opportunities for breaking reading personal information by sniffing the network. The hacker could perform what is called “war driving,” sniffing the TCP packets as they are sent over the air. He could also sniff the packets on the MIT network, since this network is not inherently secure. Running NGMC over HTTPS would address this problem. It would be a simple modification, since it does not affect the actual NGMC software at all. And doing it would provide NGMC with a much higher level of security, since it would no longer be possible for a hacker to sniff NGMC data over the air or over the MIT network.

Another security enhancement of importance is to provide a trusted form of authentication for users. The most likely way to do this is to use certificates, as most current MIT web sites do. Currently anyone with a student’s username and password can access his account on NGMC. Certificates would make it more difficult for hackers to misrepresent
themselves as a user and access personal data. This would be a fairly simple fix, since most of the support for them is already built into the htmlCtrl component of NGMC.

9.3. Tablet PCs

Having debuted only a few months ago, Tablet PCs are probably the most hyped technology today. But very few people can find good uses for them. Many have suggested that they be used for education in some way, but there has been no software that would make this a reality—yet. We believe that a modified version of the NGMC smart client would be a perfect educational application for the Tablet PC.

Imagine students walking from class to class carrying only a device the size of a notebook. On it they have all of their notes, which they write using the notepad feature of the Tablet PC. But they also have lecture slides, access to lecture text, and the ability to ask questions and take mini-quizzes in class. With the NGMC software, the Tablet PC becomes a complete in-class tool. For example, while students are listening to lecture and taking notes, the lecturer can present a mini-quiz which would appear right on the tablet PC. This would eliminate the need for students to switch between the device and their notes, as they

Figure 38 A Tablet PC with Windows XP Tablet running
have to do when using a Pocket PC. With a larger screen and better handwriting recognition, they could easily write a question to send to the lecturer during class, and they could easily schedule and manage events.

We see a lot of promise in using these devices for education. Unfortunately, at present they are much too expensive to give to students. Still, we hope that as their price decreases, with software like NGMC on them, these devices could become an integral part of the educational experience.

9.4. **Pervasive Classroom**

Another goal of ours is to make the system ubiquitous, into what we call a “pervasive classroom,” as indicated in the vision above. We hope to see students using NGMC going between classes carrying their handheld or Tablet PC and using the system in and out of class. This entails integrating the system with all of the classes and classroom devices at MIT and adding hardware and software to make it worthwhile for students and teachers to use.

9.4.1. **Classroom Hardware Interface**

The first step toward this goal is to make the system work with any classroom. To do this, IR login devices would need to be installed in every classroom, and the 802.11b handoff would need to be so that students would not have to reboot their devices when they go from class to class. Also, the system would have to be integrated with the Athena hardware in each classroom, so that devices such as projectors, workstations, cameras, and other classroom devices are all integrated. This would require a lot of software development, and because of the heterogeneous computing environment at MIT will be very difficult to program. Moreover, there are political issues over the control of hardware which we would
have to deal with in order to make this work. But the result would be a system in which a lecturer can enter a classroom, and his slides are automatically projected on the screen, and as he goes through lecture, pre-created mini-quizzes are automatically triggered. And of course, as with the current system, students will automatically see the message of the day, and lecture outline when they enter, as well as mini-quizzes and other material that the professor distributes during lecture.

9.4.2. Distribution of Devices
The next step to making NGMC ubiquitous is to find a way to acquire and circulate handhelds or Tablet PCs to students. Because the price of these devices is so high, the system is currently infeasible in large classes. In the future, we hope to be able to find a way to either purchase devices for all of the students, or to get students to purchase the devices for themselves. In addition, once the students have obtained the devices, it needs to be easy for them to install and use NGMC. This means building a versatile software distribution system with which the NGMC software is automatically installed and configured as soon as the students turn on their device. Because the network and device hardware is not reliable, the distribution system would have to be able to account for hardware problems and handle them appropriately.

9.4.3. Plug-ins
The third step to making NGMC ubiquitous is building plug-ins, both general and class-specific. One important plug-in that we hope to build is especially geared toward hearing-impaired students. This plug-in would use current speech-to-text software, such as MIT’s Galaxy, to translate the lecture speech to text in real time. As the lecturer speaks, his slide
would appear on the handheld or Tablet PC screen, along with a transcript of what he is saying. This would allow students to read the text if they are unable to make out what the lecturer said. Also, it would allow students to refer to things said earlier in lecture, in case they missed something. The lecture text would also be archived, so that students could return to it and check something they are uncertain about. It would also make it easy to distribute the lecture as part of OpenCourseware or some similar initiative. Other plug-ins may be created for specific classes, such as simulations for physics or engineering courses or video capture for business courses.

9.4.4. Out of Class Tools
Another step toward making NGMC ubiquitous is to build tools to be used on NGMC devices outside of class. There are already some tools built into the system for this purpose—the event calendar and contact information features. However, there are many opportunities to create others. For example, a reading list could automatically be distributed 45 minutes before class, so that students at lunch could quickly skim the material and possibly understand lecture better. And facilities around campus could be linked to the system, so that students could do such things as reserve athletic courts, rent videos, or order food through it. Although the latter idea is somewhat broad, it could be a feasible goal once every student has a handheld and carries it with him everywhere.

9.5. Other Future Work
Other plans for future work include the obvious—fixing bugs, and preparing the system for class next term, as well as integrating it with other campus educational programs. We have been working on fixing the bugs that were found last term in the software and dealing with
the usability issues that were brought up. We hope to see an improvement in users’ reception in the class next term. We also are trying to find a campus group to take over development and maintenance of the software next year when we are gone.
Chapter 10. Conclusion
This thesis looked at how NGMC and the NGMC smart client can increase interactivity in the classroom by taking advantage of the latest technology. This project provides a unique opportunity to take advantage of the latest technology to revolutionize the classroom experience. For students, this means a new and exciting way to learn. For professors, it means a more active and involved lecture, with less preparation time and better feedback from students. Technically, this project integrates a variety of components for the first time, resulting in a pervasive educational environment. It builds an educational infrastructure that promotes interaction between students and professors and can be used as the foundation for future learning applications.

But the most important technical accomplishment of NGMC is the powerful messaging system that allows organizational features of a PDA to be directly integrated with a multi-user network environment, where information can be shared and accessed by anyone with the appropriate permissions. NGMC creates the first real handheld-server messaging protocol, which can be extended to allow any sort of data to be transmitted and viewed by users with a handheld wireless device. Messages in NGMC are real-time, and can be dynamically routed to any component of the handheld, including its productivity applications. In addition, the NGMC messaging system ensures message security and anonymity, over heterogeneous networks (such as the 802.11b and IrDA networks used).

This allows for a variety of interaction mechanisms that are rare if not impossible in traditional large classes. For example, a professor can give "mini-quizzes" during lecture to gauge students' grasp of key concepts. With the touch of a button, he/she can deliver a multiple choice question to all students' handheld devices. As soon as the student answers the question, results are tallied and the professor sees a histogram of results. With these
results he/she can easily decide whether to delve deeper into the current concept or move on.

Another way in which NGMC promotes interaction is by allowing students to ask questions in large lecture halls. Traditionally, in these environments, students tend to feel uncomfortable speaking out and interrupting the flow of lecture. However, with NGMC, students can enter the question on their handheld device. As the lecture progresses, TAs or other faculty collect these questions and select interesting ones to pose to the professor. The professor can then answer them aloud to the entire class.

Finally, NGMC helps students manage their time by automatically entering course information into their calendar and address book. As soon as a student enters the classroom, the due dates of the latest assignments are entered into the calendar on his/her handheld. Also, the contact information for TAs and course staff may be entered in the student's address book, so that the student has easy access to this information when he/she needs it.

This project provided us with a unique opportunity to take advantage of the latest technology to revolutionize the classroom experience. For students, this means a new and exciting way to learn. For professors, it means a more active and involved lecture, with less preparation time and better feedback from students. Technically, this project defines a "smart client" which provides several features, including a new message and alert system for handhelds and servers. It will build an educational infrastructure that promotes interaction between students and professors and can be used as the foundation for future learning applications.
Bibliography


Appendix A

The source code used to scan the web server log files for information on applications used by NGMC users.

```java
import java.io.*;
import java.net.*;
import java.util.*;

class Scan {
    public static void main(String[] args) throws Exception {
        int app_quiz = 0;
        int app_course = 0;
        int app_session = 0;
        int app_media = 0;
        int app_question = 0;
        int app_total = 0;
        int app_portal = 0;
        int num_kermit = 0;
        int num_indexviews = 0;

        BufferedReader br = new BufferedReader(new FileReader("c:\xaccess.log"));
        String s = "";
        while ((s = br.readLine()) != null) {
            int pos = s.indexOf("app=");
            if (pos >= 0) {
                int pos2 = s.indexOf('&', pos);
                if (pos2 < 0) pos2 = s.indexOf(';', pos);

                String typ = s.substring(pos+4, pos2);
                //System.out.println(typ);
                if ("quiz".equalsIgnoreCase(typ)) app_quiz++;
                else if ("course".equalsIgnoreCase(typ)) app_course++;
                else if ("session".equalsIgnoreCase(typ)) app_session++;
                else if ("media".equalsIgnoreCase(typ)) app_media++;
                else if ("question".equalsIgnoreCase(typ)) app_question++;
                else if ("portal".equalsIgnoreCase(typ)) app_portal++;
            }
            app_total++;
        }
        if (s.indexOf("index.php") >= 0) {
            num_indexviews ++;
            if (s.indexOf("sid") >= 0) num_kermit ++;
        }
    }
    System.out.println("Usage Percentages");
    System.out.println("-------");
    System.out.println("Quiz: " + app_quiz + "(" + (app_quiz/(double)app_total) + ")");
    System.out.println("Course: " + app_course + "(" + (app_course/(double)app_total) + ")");
    System.out.println("Session: " + app_session + "(" + (app_session/(double)app_total) + ")");
    System.out.println("Media: " + app_media + "(" + (app_media/(double)app_total) + ")");
    System.out.println("Question: " + app_question + "(" + (app_question/(double)app_total) + ")");
    System.out.println("Portal: " + app_portal + "(" + (app_portal/(double)app_total) + ")");
}
```

95
System.out.println("Total: " + app_total);
System.out.println("Index Views");
System.out.println("Number of Index Views with Kermits: " + num_kermits);
System.out.println("Number of Index Views: " + num_indexviews);
}

public static void main(String argU) {
	ry { process(); } catch (Exception e) { e.printStackTrace(); }
}
# Appendix B

The user survey for NGMC.

Next Generation Mobile Classroom

## User Survey

<table>
<thead>
<tr>
<th>Ease of Use</th>
<th>Rating</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Login with the automated login (IR)</td>
<td>1 2 3 4</td>
<td></td>
</tr>
<tr>
<td>Manual login with smart client</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Login on PC, laptop, etc.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Smart client buttons</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Smart client quick menu</td>
<td></td>
<td></td>
</tr>
<tr>
<td>File upload system</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Audio recording/upload tool</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Comments:

- H
- J
- J
<table>
<thead>
<tr>
<th>Feature</th>
<th>Rating</th>
<th>Usefulness</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ask a question facility</td>
<td>(very easy) 1 2 3 4</td>
<td>(very easy) 1 2 3 4</td>
</tr>
<tr>
<td>Answer surveys/mini-quizzes</td>
<td>(very easy) 1 2 3 4</td>
<td>(very easy) 1 2 3 4</td>
</tr>
<tr>
<td>See survey/mini-quiz results</td>
<td>(very easy) 1 2 3 4</td>
<td>(very easy) 1 2 3 4</td>
</tr>
<tr>
<td>Use the calendar to enter an event</td>
<td>(very easy) 1 2 3 4</td>
<td>(very easy) 1 2 3 4</td>
</tr>
<tr>
<td>Access events entered by other users</td>
<td>(very easy) 1 2 3 4</td>
<td>(very easy) 1 2 3 4</td>
</tr>
<tr>
<td>Make my contact information available</td>
<td>(very easy) 1 2 3 4</td>
<td>(very easy) 1 2 3 4</td>
</tr>
<tr>
<td>Get other users' contact information</td>
<td>(very easy) 1 2 3 4</td>
<td>(very easy) 1 2 3 4</td>
</tr>
<tr>
<td>Edit my user information</td>
<td>(very easy) 1 2 3 4</td>
<td>(very easy) 1 2 3 4</td>
</tr>
<tr>
<td>Overall</td>
<td>(very easy) 1 2 3 4</td>
<td>(very easy) 1 2 3 4</td>
</tr>
<tr>
<td><strong>Usefulness</strong></td>
<td></td>
<td>(very useful) 1 2 3 4</td>
</tr>
<tr>
<td><strong>Rating</strong></td>
<td></td>
<td>(very useful) 1 2 3 4</td>
</tr>
<tr>
<td><strong>Comments</strong></td>
<td></td>
<td>(very useful) 1 2 3 4</td>
</tr>
</tbody>
</table>
Surveys/mini-quizzes

Events and calendar

Contact information

Overall

General

Would you like to use this system again?

Would you like to see this in other classes?

Would you like to see the PDAs used more in class?

Would you like to see the PDAs used more outside class?

Would you like to have more assignments using the PDAs?

Submit
Thanks for filling out the survey!
Appendix C

Survey results for user survey:

<table>
<thead>
<tr>
<th>Ease-of-use</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Login with the automated login (IR)</td>
<td>3</td>
<td>3</td>
<td>7</td>
<td>5</td>
<td>1</td>
<td>2.894736842</td>
</tr>
<tr>
<td>Manual login with smart client</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>2.105263158</td>
<td></td>
</tr>
<tr>
<td>&quot;Login on PC, laptop, etc.&quot;</td>
<td>8</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2.157894737</td>
</tr>
<tr>
<td>Smart client buttons</td>
<td>6</td>
<td>8</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>2.105263158</td>
</tr>
<tr>
<td>Smart client quick menu</td>
<td>9</td>
<td>6</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>1.789473684</td>
</tr>
<tr>
<td>File upload system</td>
<td>1</td>
<td>4</td>
<td>6</td>
<td>6</td>
<td>2</td>
<td>3.210526316</td>
</tr>
<tr>
<td>Audio recording/upload tool</td>
<td>0</td>
<td>5</td>
<td>8</td>
<td>5</td>
<td>1</td>
<td>3.105263158</td>
</tr>
<tr>
<td>Ask a question facility</td>
<td>5</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1.631578947</td>
<td></td>
</tr>
<tr>
<td>Answer surveys/mini-quizzes</td>
<td>7</td>
<td>9</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1.842105263</td>
</tr>
<tr>
<td>See survey/mini-quiz results</td>
<td>7</td>
<td>8</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>1.894736842</td>
</tr>
<tr>
<td>Use the calendar to enter an event</td>
<td>5</td>
<td>8</td>
<td>1</td>
<td>1</td>
<td>2.473684211</td>
<td></td>
</tr>
<tr>
<td>Access events entered by other users</td>
<td>5</td>
<td>8</td>
<td>4</td>
<td>2</td>
<td>0</td>
<td>2.157894737</td>
</tr>
<tr>
<td>Make my contact information available</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1.736842105</td>
</tr>
<tr>
<td>Get other users' contact information</td>
<td>9</td>
<td>5</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2.052631579</td>
</tr>
<tr>
<td>Edit my user information</td>
<td>7</td>
<td>9</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1.842105263</td>
</tr>
<tr>
<td>Overall</td>
<td>2</td>
<td>7</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td>2.789473684</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Usefulness</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ask a question facility</td>
<td>7</td>
<td>8</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>1.894736842</td>
</tr>
<tr>
<td>Surveys/mini-quizzes</td>
<td>15</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1.263157895</td>
</tr>
<tr>
<td>Events and calendar</td>
<td>9</td>
<td>6</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>1.789473684</td>
</tr>
<tr>
<td>Contact information</td>
<td>10</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1.947368421</td>
</tr>
<tr>
<td>Overall</td>
<td>5</td>
<td>6</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>2.421052632</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>General</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Would you like to use this system again?</td>
<td>2</td>
<td>9</td>
<td>6</td>
<td>1</td>
<td>1</td>
<td>2.473684211</td>
</tr>
<tr>
<td>Would you like to see this in other classes?</td>
<td>4</td>
<td>7</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>2.421052632</td>
</tr>
<tr>
<td>Would you like to see the PDAs used more in class?</td>
<td>6</td>
<td>3</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>2.684210526</td>
</tr>
<tr>
<td>Would you like to see the PDAs used more outside class?</td>
<td>3</td>
<td>7</td>
<td>8</td>
<td>1</td>
<td>3.368421053</td>
<td></td>
</tr>
<tr>
<td>Would you like to have more assignments using the PDAs?</td>
<td>3</td>
<td>9</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>2.378947368</td>
</tr>
</tbody>
</table>