University of Nebraska-Lincoln

College of Engineering
Computer and Electronics Engineering Department

Nimbus

By:
Chet Cyr
Jeff Lind
Steven Baker
Louis Sather

May 30, 2011

Abstract

Schools are constantly striving to increase the technology in classrooms while reducing cost. Traditionally, collections of laptops are used to create a mobile lab. However, these have a large upfront cost and require constant maintenance. Newer computer labs consist of many thin clients that connect to a central network server. Thin clients require an Ethernet connection, monitor, and mouse, which render the devices immobile.

The Nimbus will be able to fulfill this need because it is a wireless mobile computing device that connects to a server. Since the server handles the processing for each device, the Nimbus will not require the more complex and costly hardware that is required by laptops. Each device will act as a remote terminal to a desktop session. To increase the educational atmosphere of the product, a screen sharing mode has been implemented. This allows for a master device's screen to be displayed on each slave device. The screen sharing mode enables a teacher to display the same image to each student's Nimbus directly, creating a simulated one on one learning experience.

This project has designed and created a Nimbus device that met these objectives. This paper goes into detail about how the members of the Nimbus project team met the goals stated and more specifically, how the team met the IEEE standards 1233-1996, 829-2008, and 1012-2004. These standards relate to the project itself and the methods employed during the development cycle of the Nimbus.

Introduction

The Problem

The problem that the Nimbus project addresses is the need for increased interactivity between students and teachers in classrooms. Funding has always been a problem for schools, and with cutbacks from the government, schools look for ways to save money. As time goes on, the demands of schools increases, teachers are expected to provide individual time for each student and ensure the formation of well-educated and well-rounded individuals. With this pressure to do more with less, the members of team Reconstruction have noticed the need for a cheap, interactive, technological solution to assist teachers and students alike.

The Objective

The goal of the Nimbus project is to create a better group dynamic by allowing a teacher, presenter, or lecturer to use a device to facilitate communication. The Nimbus project set out to create a handheld computing device that would be cheap, simple and familiar. In order to be implemented in a typical school setting, the team devised a list of objectives. To maintain familiarity, the Nimbus device would need to be compatible with the currently dominant


computing environment. The most commonly used environment for schools is a Windows based infrastructure with wireless connectivity in the form of 802.11b/g Wi-Fi™. Simplicity means an intuitive and responsive user interface. The Nimbus project must bring functionality to the device simply. In addition to the compatibility with Windows based infrastructure, an intuitive screen sharing mode should exist. This would allow teachers to share their information, instructions, or lessons to their students and providing a means to involve students in the actual presentation of the information, not just in the reception of information.

The Standards

The standards chosen by the project team to review in this paper were IEEE standards 1233-1996, 829-2008, and 1012-2004. 1233-1996 is the Guide for Developing System Requirements Specifications. This was the first standard used by the team, chronologically and was used to develop requirements for the project. The second standard, 829-2008, discusses the testing schedule and standards used during the testing portion of the project. Finally, the 1012-2004 standard was used to document, verify and validate the software created in the project.

System Overview

The Nimbus is a mobile computing device. It was developed in order to bring teachers and students closer together in a classroom, improving the learning of students. When the Nimbus first starts, it asks the user to select if they would like to enter desktop or screen sharing mode. The desktop mode was designed for, but was not part of the project. This mode would allow a Nimbus to connect to a server through a terminal session, allowing for the Nimbus to create a Windows environment. This would allow students to individually work on papers, research, or other homework that can be done on a normal desktop computer. This was further developed by the use of automatically saving log in time, log out time, and other important usage statistics into an SQL database. An administrator can view this database through a password protected HTML/PHP website.

The second mode of the Nimbus is the screen sharing mode. The screen sharing mode allows for one master user, such as a teacher, to display their screen onto each of the slave devices. This is done through use of the 915MHz frequency with Frequency Shift Keying modulation. The master device transmits to all devices in range the master's screen data so each slave device can display the image on the LCD screen. To compress the data being transmitted, Run Length Encoding was chosen to send data over the network because it allows for compression of continuous data.

In order to make this device usable to students in a classroom setting, there are necessities that the Nimbus must have on top of the two operational modes. The first is that the device must be portable, so the student can move from classroom to classroom without having a tangle of cords. This requires a battery to supply power to the system, along with a way to charge the

batteries. There should also be preliminary status indicators to allow a user to perform basic troubleshooting of the Nimbus. No sound should be present for the system that would distract the students from learning. This will prevent students from watching online videos that are a distraction from class.

IEEE Standards

1233-1996 - IEEE Guide for Developing System Requirements Specifications

In order to make a project fully understood by both the technical community and the clients, it is necessary to develop system requirements that spell out the goal of the project.[3] To build well-formed System Requirement Specifications (SyRS), requirements must be taken from a variety of sources, including, but not limited to customers, environment, political impact, and the technical community. These different sources can identify the requirements, upon which, an analysis will condense and synthesis the requirements into the SyRS, which should be abstract, unambiguous, traceable, and possible to validate. These developed SyRS must be presented to the technical community and customers to verify the SyRS meet the initial presented requirements.

During the project, the team first identified the sources of requirements, which were the team, the Senior Project Officer, and teachers. The team developed requirements by having a brainstorming session where all ideas were presented. When talking to a teacher from a local high school, many of the basic requirements for the system overlapped. The Senior Project Officer had requirements that were necessary for the class. This identified list of requirements is listed below:

- The Nimbus will have an easy to use user interface.
- The project will be able to connect to a server wirelessly.
- The Nimbus will be portable.
- The Project will be able to display to a projector.
- The Nimbus will be able to communicate to other Nimbus.
- The Nimbus will have a built-in keyboard, touch screen, and USB port.
- Test individual sections of the project for functionality.
- Provide simulations of appropriate circuits for analysis.
- Provide an efficient design for the project.
- A budget of $1000 for total parts must be maintained (include bad or unused parts).
- Use Skill Teams and peers for assistance when a problem is encountered.
- Apply for and use an IEEE mini-grant.
- Request sample parts from manufactures when available.
- Use information from datasheets, catalogues, and the library.
- Consult Faculty members on relevant topics.
- Meet and complete project Milestones.

- Have weekly agendas, meetings, and minutes.
- Use log books and proper documentation.

From this list of brainstormed ideas and requirements from the teacher, the ideas were condensed, removed, and synthesized into ten project requirements. These were presented to the Senior Project Officer. He presented feedback on these SyRS, which lead to a redefinition and final revision:

1. Read keyboard and touch screen input data with at least 95% accuracy determined by display output with at least 640x480 resolutions at 30Hz. {3, 2, 2, Input/Output}
2. Design a broadcasting peer-to-peer output display network containing at least two simulated slave nodes and one master node at 915MHz. {2, 1, 4, Growth capacity}
3. Compress transmission data using Run Length Encoding over a wireless network preventing unauthorized access. {3, 1, 5, Security}
4. Develop and implement an SQL database to automatically log usage statistics such as user and device IDs, log in time and log out time. {1, 1, 5, Input}
5. Using information stored in an SQL database, create a password protected HTML report that will be generated using PHP for a Windows Server to be accessed by a Network Administrator via web browser. {2, 2, 5, Output}
6. Create a bill of materials and order/sample all parts needed for the design (BOM) {1, 1, 5, Documentation}
7. Develop a complete, accurate, readable schematic of the design along with {1, 1, 5, Documentation}

a. Interface Loading Analysis
b. Interface Timing Analysis

8. Complete a layout and etch a printed circuit board (PCB). {1, 1, 5, Performance}
9. Populate and debug the design on a custom printed circuit board. {1, 1, 5, Performance}
10. Package the finished product {1, 1, 5, Performance}
11. Demonstrate its functionality

Each of these was seen as abstract, unambiguous, traceable, and possible to validate. The first five were identified by the team as necessary to complete the project as desired for a learning environment. The last five were presented by the Senior Project Officer as requirements for the class.

The SyRS had to be categorized in order to achieve a structure that would lead engineers to the development of the Nimbus. Listed above, the SyRS are in descending priority. Each is also given Criticality, Feasibility, Risk, and Type (Where 1 is most critical, feasible, or risky and 5 is least critical, feasible, or risky).

829-2008 - IEEE Standard for Software and System Test Documentation

IEEE Standard 829-2008 is titled "Standard for Software and System Test Documentation." The purpose of this standard is to provide a method to test portions of the project and for the tests to remain consistent between the different collaborators and associated parties of the project. This standard assists in communication between the parties, as well as reducing the overhead and complexity of the testing process.[2]

During the course of the project, specifically during the planning phase, a test plan was generated. According to the standard, a test plan involves a testing schedule and standardized method for testing the schedule. In order to produce a valid testing schedule, a list of the objects to be tested was generated, as well as the criteria for testing each object.
Each high level test corresponded to a project requirement. The final five tests that were performed were the integration tests, and tested the following:

- User Interface including Keyboard, Touch Screen, and LCD
- Wireless compression using Run Length Encoding
- Wireless networking via Frequency Shift Keying
- A Structured Query Language Database that stored logged data
- A PHP web page to display the Database data

Each test was broken down, according to the testing plan, and each assigned their own unique test identifier. Each test had listed the items to be tested, the specific features to be or not to be tested, and a list of pass fail criteria which included the method, or approach, to test the criteria. Due to the nature of the project, test suspension and resumption were mentioned in the Project Plan technical paper, which cited that if a test needed to be suspended, the test progress would be invalidated and the test must be completed again in its entirety.

The deliverables of the test were included in the Input/output sections of the test for each pass/fail criteria. The environmental needs section of the test, as required by the specification, was included and labeled as "Setup."

Each test was performed in its entirety by one person, specified at the beginning of the test document. The person chosen to perform the test would have the required skills to perform the test, and was chosen by the systems engineer to perform the test.

The risks and contingencies were listed in the Project Proposal, as well as the final report of the project. Specific risks for each test were included in the comments section of the test.
The testing procedure used was consistent across all tests, with all incidents recorded. If a test failed, the product was sent back into production using the results of the test to fix the problem.

1012-2004 - IEEE Standard for Software Verification and Validation

The IEEE Standard for Software Verification and Validation is a standard that develops a framework for software verification and validations tasks. These software verification and validation tasks require that all inputs and outputs be clearly defined for the system code.[1] The need arose in this project to conform to IEEE standard 1012-2004 because the verification and validation tasks help determine and correct software errors that may occur during the software development stage. The verification tasks also were needed to help ensure that the project software complied with the overall system requirements.

The software development phase of this project utilized this IEEE Standard. The software verification and validation tasks were completed by compiling the program code with the GNU C compiler available in KEIL. KEIL was the software environment that the software was developed in and it helped verify the program code that was created for this project. The software was validated successfully when it fully met the required system specifications and features required by the project objectives. All of the program functions were defined using documentation headers to clearly define the inputs and outputs of the program. The following is an example documentation header which assisted in helping to find problems in the software.

```
//Function Definition
//Name: Name of the Function
//Inputs: Inputs to the function
//Returns: Return type from function
//Outputs: Outputs from the function, to the LCD, Wi-Fi, or LEDs.
//Notes: Any notes related to the function.
```

In clearly defining these system inputs and outputs of every function of the software, it greatly aided the debugging process of the development cycle. The process that was followed to test the software helped validate that it met the system requirements and verify that it was working properly.

Conclusion/Results

With the current hardware configuration Nimbus using an ARM Cortex M3 it is limited on memory and speed. The use of an ARM 9 or equivalent microcontroller could allow for a complete remote desktop configuration and a refresh rate on the LCD screen. This set aside the Nimbus is capable of reading using input data with 95% accuracy and displaying at 640x480 resolution. Broadcasting to multiple Nimbus's is possible with a limited frame rate and run length encoding has been achieved through software implementation. An SQL data base has

been configured to record any device that connects to the network and the data is accessible through a HTML/PHP website.

The Nimbus provides a solution that is currently not available. It's a portable computing solution that uses inexpensive hardware components. It records user log data to a database so that each device can be tracked and accounted for. Using a separate transceiver it is able to send and receive screen data for presentation purposes. Along with this is the IEEE Standards have helped to clearly mapped out the requirements of the project, test the components of the project and create the framework of the software.

The documentation, structure and standards used in this project have set the standard of future projects for the members of this team. Working in a multidisciplinary environment is something to be expected in all projects and documentation must be there to effectively communicate with each other. The outlined IEEE standards created an outline of how to do this properly and helped to complete this project.

References

[1] 1012-2004 - IEEE Standard for Software Verification and Validation. Internet: http://standards.ieee.org/findstds/standard/1012-2004.html. [Accessed: January 20, 2011].

[2] 829-2008 - IEEE Standard for Software and System Test Documentation. Internet: http://standards.ieee.org/findstds/standard/829-2008.html. [Accessed: January 20, 2011]

[3] 1233-1996 - IEEE Guide for Developing System Requirements Specifications. Internet: http://standards.ieee.org/findstds/standard/1233-1996.html. [Accessed: January 20, 2011]