# Proceedings of the 7th Winona Computer Science Undergraduate Research Symposium

April 19, 2007

# Table of Contents

# Network File Distribution with the BitTorrent Protocol

Lincoln Scully

Saint Mary's University of Minnesota
700 Terrace Heights #1605
Winona, MN 55987

lascul02@smumn.edu

## ABSTRACT

The peer-to-peer BitTorrent protocol is presented as a means for distributing content internally over a network, rather than relying on the traditional client-server protocols. Instead of establishing a single one-way stream of information, BitTorrent makes several connections to other clients that contain at least part of the desired information. This information is then simultaneously downloaded from and uploaded to the other clients in pieces. Using a server and several clients on a network, BitTorrent is clocked alongside two traditional protocols to determine which takes the least amount of time. Analysis of the various protocols' performance suggests that only three or four clients are needed for BitTorrent to complete the file transfers more quickly, the trend being that the time saved increases with the number of clients and the size of the content being transferred.

## Categories and Subject Descriptors

C.2.2 [**Network Protocols**]: Applications; C.2.5 [**Computer-Communication Networks**]: Local and Wide-Area Networks – Internet

## General Terms

Measurement, Performance

## Keywords

Peer-to-peer, Client-Server, Networks, BitTorrent, Content mirroring, File backup

## 1. INTRODUCTION

BitTorrent is a peer-to-peer networking protocol based on the idea of several sources downloading and uploading to and from each other concurrently [1]. Coded in 2001 by Bram Cohen, it was originally intended and utilized for distributing large files among end users, namely Linux images [1, 2]. Since then it has become a popular method for sharing illegal digital copies of movies [1, 2] and is reportedly responsible for one-third of all Internet traffic today [3]. Because of this, the creator has formed BitTorrent Inc. and negotiated with the Motion Picture Association of America and raised capital to make www.bittorrent.com into a store that sells online video content [3].

The BitTorrent protocol is of unique design in that it employs both a "choking" algorithm and a "rarest first" algorithm. The choking algorithm promotes uploading by curbing the download speed when not sharing to encourage the communal experience of peer-to-peer file sharing. Being that BitTorrent breaks files up into several pieces to optimize downloading [4], the rarest first algorithm looks for the rarest piece of the volume to be transferred among all the uploaders and downloads it first. These are two of the reasons why BitTorrent is so efficient and reliable [5].

The great majority of BitTorrent use has been among home users, exchanging legal Linux distributions and illegal media files. Besides Bram Cohen's venture that is attempting to build a business on the technology he created, the only business entity outside of the Linux community to utilize BitTorrent is Blizzard Entertainment. They incorporated a client in their gaming software to facilitate the downloading of updates to their World of Warcraft game [6].

A potential use for BitTorrent in addition to file sharing and software/update distribution would be internal file transfers of commercial and non-profit organizations; examples of which are website/content mirroring and data backup. Perhaps the fact that BitTorrent has taken off as a piracy tool has discouraged legitimate organizations from experimenting and seeing how they might benefit from this technology. This idea of utilizing BitTorrent for internal file transfers is unexplored and is the purpose for this research.

Traditionally, files are transferred using either File Transfer Protocol (FTP) or HyperText Transfer Protocol (HTTP). FTP is the oldest, having originated in 1985 and was intended for transferring files over a network. HTTP was initially developed as a means of retrieving web pages but has since become a general purpose protocol for Internet and network communications. Both of these protocols are based on a 1-to-1 client-server ratio.

Because of the single source characteristic of HTTP and FTP, it was hypothesized that BitTorrent would take less time when mirroring content to several other machines simultaneously because of the sharing amongst all the clients. In this situation, the other protocols cannot compete with BitTorrent's "near-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

*Proceedings of the 7th Winona Computer Science Undergraduate Research Seminar,* April 19, 2007, Winona, MN, US.

optimal performance" [9] in terms of uplink bandwidth utilization and download time [10]. As the number of servers receiving the content increases, so would BitTorrent's efficiency relative to HTTP and FTP.

Recently a network simulator was developed called the General Peer-to-Peer Simulator (GPS) that can quite accurately model BitTorrent network activity [7]. However, since real network tests were possible and feasible they were performed because they are tangible and more acceptable, even to the BitTorrent creator [8].

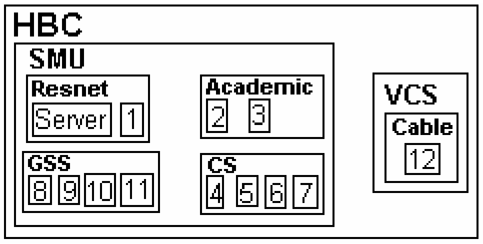
## 2. METHODOLOGY

### 2.1 Test Bed

Comparing the performance of the BitTorrent protocol versus the standard Hypertext (HTTP) and File (FTP) transfer protocols involved setting up several client PCs on different subnets of the Saint Mary's University (SMU) network. These included two on SMU's residential subnet, two on the academic subnet, four in the Computer Science laboratory, and four in the GeoSpatial Services cubicles, as well as one cable modem connection off campus at Valley Computer Solutions (VCS), from Saint Mary's' service provider, HBC. The PCs were desktops that varied from 333Mhz Pentium 2 machines to 1Ghz Pentium 4 machines with anywhere from 192MB up to 1GB of RAM, as system bottlenecks would affect each test equally. All of these clients had 10Mb ethernet connections or were on switches that had a 10Mb feed.

The tests were performed during an academic break while university was officially closed with almost all students off campus. This provided a low-traffic network for which repetitive tests could be run with no significant anomalies or inconsistent data.

**Figure 1. Test Bed Diagram.**



### 2.2 Software Utilized

All PCs involved in the tests had legal copies of Windows 2000 Professional or Windows XP Professional installed, with all the latest service packs and updates, as well as the Java 2 Platform Standard Edition Runtime Environment 6.0 (JRE 1.6.0). For the BitTorrent tests the host server utilized the free and open-source Java-based Azureus 2.5.0.4 program running on port 4950 which served as both the initial server and kept track of the torrents. For the HTTP and FTP tests the server was running Microsoft's Internet Information Services 6.0, the website running on port 4951 and the FTP site on 4952. The clients all used the official BitTorrent client as well as Java HTTP and FTP download classes to initiate and transfer the test files utilizing URLConnections and binary BufferedInputStreams. Automation was achieved by means of a Java socket listener, running on port 4953, that instantiated DOS batch files that called the desired test program with the right parameters. After several tests it was determined that the HTTP and FTP transfers would be performed one at a time to simplify timing because the difference between that and simultaneous transfers offered no statistically significant difference in performance. Whereas, the BitTorrent tests were fully automated by sending a command to the Java sockets of several machines at once to initiate the file transfers simultaneously and then Azureus log files were analyzed to determine the time it took to successfully transfer the content to all machines, from the start of the first incoming connection to the last end of stream socket exception. Great care was taken in setting up both the BitTorrent clients and the Azureus seeder/tracker to ensure that the exchanges among them would not be hindered by transfer rate limits nor unnecessary traffic, such as Azureus' default Distributed Database tracker or multi-Azureus client connectivity.

### 2.3 Test Plan

The tests were performed incrementally, the first test starting out as the host server transferring to one other machine. Each subsequent test then added another server as time and resources allowed with the end result being that every server contained a copy of the files being transferred to all the servers.

The transfer times would then indicate in which situations each protocol works best, and also provide a metric for comparison, either supporting or disproving the hypothesis.

The tests consisted of transferring random zip files of approximately 10MB, 100MB, and 1GB. This was to illustrate whether or not this made a difference in the protocols' performance.

## 3. RESULTS AND ANALYSIS

As hypothesized and predicted, BitTorrent took longer than HTTP in the first couple of tests, but BitTorrent overcame that difference and gradually widened the gap in performance as the testing went on with more and more machines being added. The biggest gap in results came from the 1GB test as is shown in Table 1. The complete results are shown in Appendix 8.3.

**Table 1. The 1GB tests (ms).**

| Ratio | 1GB BT | 1GB HTTP | 1GB FTP |
|---|---|---|---|
| **1-to-1** | 1073493 | 999343 | 1018997 |
| **1-to-2** | 2029849 | 1988906 | 2011463 |
| **1-to-3** | 2689217 | 2968875 | 3003322 |
| **1-to-4** | 2908993 | 3949655 | 3991422 |
| **1-to-5** | 2021227 | 4935430 | 4978921 |
| **1-to-6** | 2620776 | 5911176 | 5967121 |
| **1-to-7** | 2591917 | 6893301 | 6958753 |
| **1-to-8** | 3238437 | 7910424 | 7982288 |
| **1-to-9** | 3509326 | 8912535 | 8983352 |
| **1-to-10** | 3501215 | 9924880 | 9992839 |
| **1-to-11** | 3657008 | 10958656 | 11023011 |

In Table 1, notice that both the HTTP and FTP file transfers took less time than the BitTorrent when the server ratio was 1-to-1, meaning that one machine held the complete data and it was being sent to only one machine. But when the ratio is 1-to-3, the BitTorrent file transfer takes the least time by a slight margin.

Finally, when the ratio is 1-to-11, the BitTorrent transfer takes roughly one-third as much time as the HTTP and FTP transfers.

The few BitTorrent tests that were executed with 10MB file transfers show that BitTorrent works best over a period of time, which requires either a slow connection or an exceptionally large file. For example, notice that sending even a smaller file like 10MB to twelve clients—including one on a slower cable modem connection—resulted in a time savings of more than 60% when compared to HTTP. This is consistent with the 1GB tests and similar to the 100MB tests, which saw about a 48% time savings with seven clients, and roughly a 60% time savings with eleven.

One unforeseen result came about in the 1GB test on the slowest connection. In this case FTP was quicker than HTTP, albeit by a small amount. However, an inquisitive mind would wonder if this is a trend that would continue if further tested. Regardless, that is outside the scope of this research project.

To put a perspective on the time saved instead of just a percentage, the longest, largest BitTorrent transfer took 7301648 milliseconds less time than the corresponding HTTP transfer, which is roughly two hours, which is a very significant difference.

A closer look at the data reveals that although the HTTP and FTP transfer times increased linearly, BitTorrent's performance fluctuated slightly and followed more of a curve. A graph illustrating this is shown in Appendix 8.2. This is suspected to be due to multiple clients residing on the same subnet facilitating quick exchanges between them, as well as a hint of randomness consisting of exactly which clients connected first to the tracker and which clients found each other first and committed bandwidth to each other first. Also, such connections are probably due to which pieces of the torrent are contained on what machines at any given time i.e. the rarest-first algorithm.

## 4. CONCLUSIONS

BitTorrent takes less time than HTTP and FTP when backing up/transferring large files to several machines at one time. In these tests, the performance gap widened to where two-thirds of the time was saved when using the BitTorrent protocol when eleven clients were involved. This shows that BitTorrent can be used effectively by an organization to save time when sending content out to several machines, the time saved increasing with the number of machines involved.

## 5. FURTHER RESEARCH

An ideal extension of this research would be to develop a site mirroring or network backup utility that automated file replication and distribution using the BitTorrent protocol. This would have to automate several of the steps in sharing BitTorrent content, but would actualize and validate this research. Essentially a server piece would have to be developed residing on the server to be mirrored that would send its data via a built-in BitTorrent agent to the client piece residing on the mirror servers with a built-in BitTorrent client to receive and share the data with the others while keeping a complete copy for itself. BitTorrent would be the means for quickly backing up data on several servers over a network or the Internet to prevent data loss.

## 6. ACKNOWLEDGEMENTS

Many thanks to all those who helped make this project happen, either implicitly or explicitly, including but not limited to God, my supervisor Dr. Gerald W. Cichanowski, Ann C. Smith, Dr. Joan M. Francioni, David E. Hajoglou, Nathan Lloyd & the SMU IT Department, Mark Krinke & Digicom, Inc., Jim Bloedorn & GeoSpatial Services, Mom & Dad, my fellow CS495 classmates, my roommates for putting up with the server in the living room, and my advisor, Carol Shields, for putting up with me all these years.

## 7. REFERENCES

[1] S. H. Kwok. File sharing activities over BT Networks: pirated movies. *Computers in Entertainment*, Vol. 2, Issue 2, Apr. 2004, page 11.

[2] C. Thompson. The BitTorrent Effect. Wired Magazine Issue 13.01, Jan. 2005. http://www.wired.com/wired/archive/13.01/bittorrent.html (8 Feb 2007).

[3] S. Levy. No, It's Not The New Napster. Newsweek Vol. 146, Issue 22, page E4. Nov. 28, 2005

[4] M. Ceballos, J. Gorricho. P2P File Sharing Analysis For a Better Performance. *Proceedings of the 28th International Conference on Software Engineering,* pages 941-944, 2006.

[5] A. Legout, G. Urvoy-Keller, P. Michiardi. Rarest first and choke algorithms are enough. In *Proceedings of the 6th ACM SIGCOMM on internet Measurement* (Rio de Janeriro, Brazil, October 25 - 27, 2006). IMC '06. ACM Press, New York, NY, pages 203-216.

[6] World of Warcraft – Frequently Asked Questions. Blizzard Entertainment, 2005. http://www.blizzard.co.uk/wow/faq/bittorrent.shtml (8 Feb 2007).

[7] W Yang, N. Abu-Ghazaleh. GPS: A General Peer-to-Peer Simulator and its use for Modeling BitTorrent. In *Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, 2005*, pages 425-432.

[8] L. Goad. BitTorrent Creator Dismisses Microsoft P2P Project. eWeek News, June 21, 2005. http://www.eweek.com/article2/0,1895,1830206,00.asp. (28 Feb 2007)

[9] A. R. Bharambe, C. Herley, V. N. Padmanabhan. Analyzing and Improving a BitTorrent Network's Performance Mechanisms. *IEEE Conference on Computer Communications*, 2005.

[10] A. R. Bharambe, C. Herley, V. N. Padmanabhan. Some Observations on BitTorrent Performance. In *Proceedings of the 2005 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, pages 398-399, Banff, Alberta, Canada, 2005.

## 8. APPENDIX

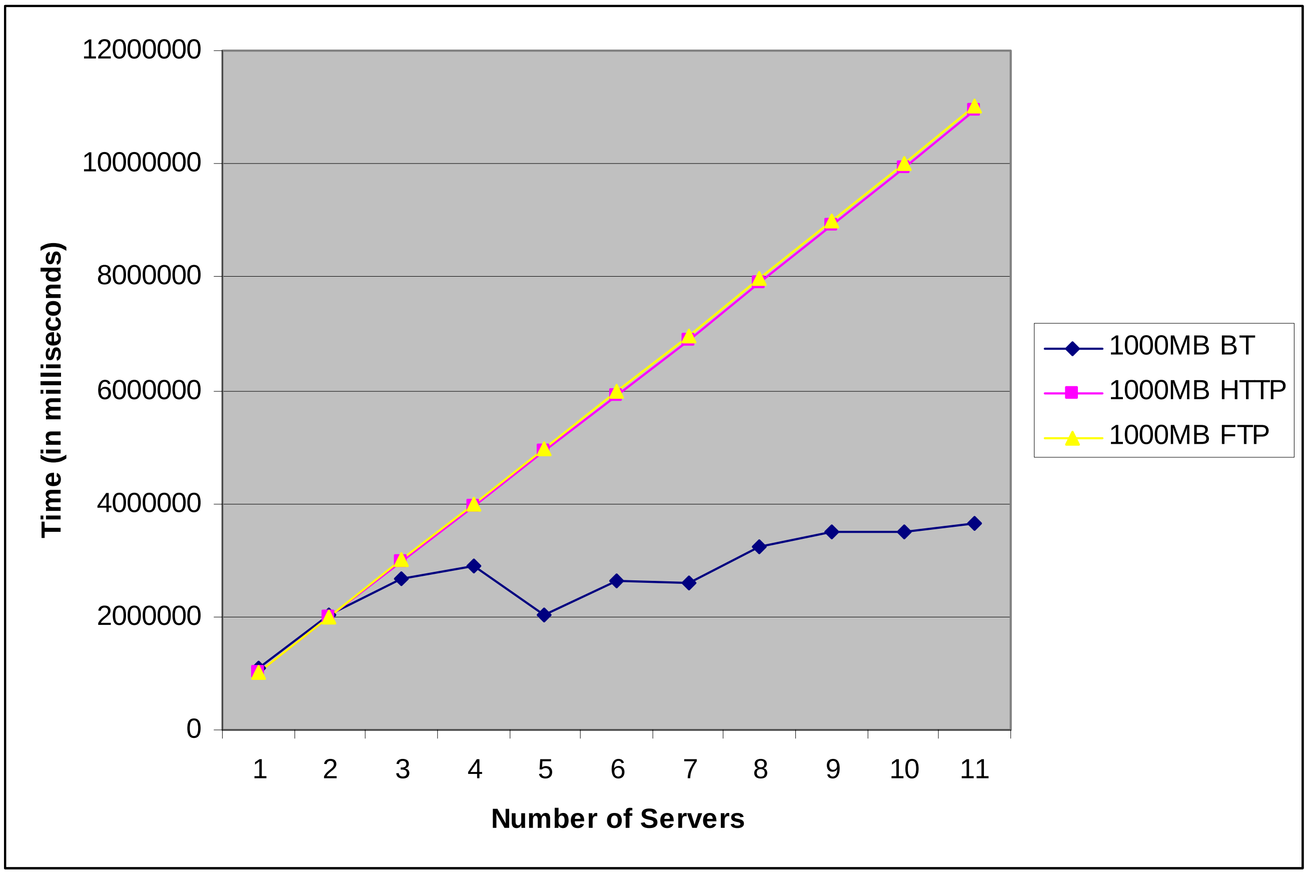
### 8.1 Glossary

**(.)torrent** - A text file that points to machines seeding the desired content as well as other machines in the swarm
**Leechers** - People who download from others without sharing any files on their own computers
**Seed** or **seeder** - A computer with a complete copy of the BitTorrent content (At least one seed computer is necessary for a BitTorrent download to operate)

**Swarm** – The totality of all computers simultaneously downloading and uploading the same torrent content

**Tracker** - A server that manages the BitTorrent file-transfer process

## 8.2 Graph showing the length of time relative to the number of servers involved.



## 8.3 File transfer times in ms.

| Ratio | 10MB BT | 10MB HTTP | 10MB FTP | 100MB BT | 100MB HTTP | 100MB FTP | 1000MB BT | 1000MB HTTP | 1000MB FTP |
|---|---|---|---|---|---|---|---|---|---|
| 1-to-1 | 11727 | 9598 | 30724 | 117252 | 98874 | 106278 | 1073493 | 999343 | 1018997 |
| 1-to-2 | 24485 | 19550 | 61223 | 256509 | 196144 | 221933 | 2029849 | 1988906 | 2011463 |
| 1-to-3 | 35130 | 29077 | 91641 | 317146 | 291453 | 336948 | 2689217 | 2968875 | 3003322 |
| 1-to-4 |  | 38628 | 122114 | 345357 | 387502 | 452407 | 2908993 | 3949655 | 3991422 |
| 1-to-5 |  | 48027 | 152556 | 391072 | 485318 | 568715 | 2021227 | 4935430 | 4978921 |
| 1-to-6 |  | 57492 | 183025 |  | 581045 | 684325 | 2620776 | 5911176 | 5967121 |
| 1-to-7 | 78213 | 66972 | 213560 | 355231 | 677151 | 799599 | 2591917 | 6893301 | 6958753 |
| 1-to-8 |  | 76806 | 244201 |  | 775349 | 917248 | 3238437 | 7910424 | 7982288 |
| 1-to-9 |  | 86447 | 274858 |  | 872761 | 1040453 | 3509326 | 8912535 | 8983352 |
| 1-to-10 |  | 96338 | 305450 |  | 971501 | 1156918 | 3501215 | 9924880 | 9992839 |
| 1-to-11 |  | 106255 | 336036 | 339959 | 1070140 | 1279332 | 3657008 | 10958656 | 11023011 |
| 1-to-12 | 98001 | 303786 | 553278 |  | 2522562 | 3454656 |  | 31415212 | 29987551 |

# Analysis of Microsoft Office 2007 User Interface Design

Catherine Beel
Saint Mary's University
700 Terrace Heights
Winona, MN 55987
cabeel03@smumn.edu

## ABSTRACT

Microsoft's recently released Office 2007 suite has a user interface that appears significantly different from any of its previous suite interfaces. The UI redesign represents a massive research and development effort and is intended to enable users to focus on the task at hand (what) rather than tool itself (how). Microsoft claims that new UI features, such as ribbons and contextual tabs, are easier to use and less distracting than the traditional menu-oriented features. We tested a number of these new UI features during formal usability experiments. Our results show that, on average, users did not perform operations any better in this new UI. The users found the arrangement of the new features to be confusing. Users experienced difficulties locating operations to perform various tasks.

## General Terms

Measurement, Design, Experimentation, Human Factors.

## Keywords

Usability study, HCI, Office 2007

## 1. INTRODUCTION

User Interfaces (UI) are extremely important in today's computing environments, The design and creation of the UI must be done according to UI design principals to ensure their ease of use. In 2007 Microsoft will release a radically revamped UI for its Office products [5]. For example, the traditional file menus have been replaced with the new Ribbon feature, shown in Figure 1. [1] We will provide a timely investigation of the effectiveness of this new UI design before Office 2007 is released to the public.

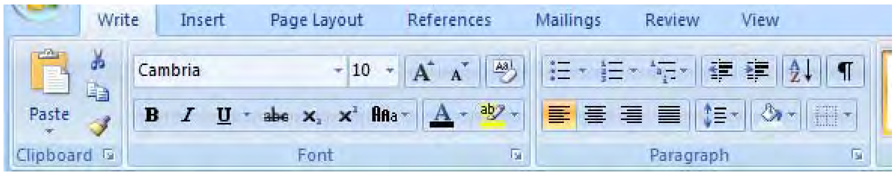


**Figure 1.** The Ribbon

Many of these UI design changes stem from well-researched Human Computer Interaction (HCI) principles. For example, Fitt's Law [7] motivates the change from the traditional icon sizes, which are shortcuts for the UI features, to the new icons found on the Ribbon. Now upon viewing the icons a user will notice the new icons are labeled with significantly larger clickable images. This will make them larger targets. With a higher screen resolution available on most current monitors the Ribbon can be larger and hold more icons [2]. The Mini Toolbar (Figure 2) was created and designed following the principals laid out by Fitt's Law. The developers implemented a simpler version of this concept in the old UI, called 'on-object UI'. [4] This basic version is the AutoCorrect feature found in the Office Applications. It is the small tab that appeared next to auto-corrected text in Word. With the Mini Toolbar appearing next to the mouse after an object is selected, the distance the mouse needs to travel to use an available feature is greatly reduced. The majority of the object relevant functions are available on this toolbar.

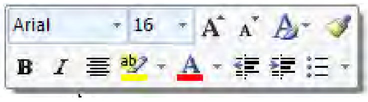


**Figure 2.** The Mini-Toolbar

The Contextual Tabs concept engineered the change from the traditional menu toolbars to the tabs the user sees when an object is active in the UI [3]. For example, if the user is using the drawing tool, the tab that is relevant to the drawing feature will appear next to the menu tabs on the ribbon. This newly appeared tab will contain the functions that are available for that object. When an object is no longer selected, the tab that appeared for the object will disappear because the functions that the tab provided are no longer applicable.

Microsoft believes that its choice to align their design decisions with HCI principles will pay off quickly with user satisfaction in the way of a quick learning curve for established Office users. We hypothesize that the jump to this UI paradigm is too large and will cause immediate problems for established Office users. More formally, we hypothesize that despite the fact that Microsoft designed their new Office 2007 suite to adhere to currently accepted UI principles, users still face a usability hurdle due to the act that they are conditioned to operate in the previous Office UI paradigm.

## 2. METHODS

To properly conduct a usability study on the Office 2007 UI, we found 50 participants and anticipate that will be sufficient to generate significant results. The participants have varying levels of experience with previous Office versions. We gave a pre-test survey for the participants in order to gauge their current Office proficiency and they ware categorized as low, medium, or high performers.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

*Proceedings of the 5th Winona Computer Science Undergraduate Research Seminar,* April 19, 2007, Winona, MN, US.

We were able to recruit participants from a variety of different majors and ages of users that can be found on a University campus. These participants ranged from students to faculty, each who have different levels of comfort with currently available UI's.

## 2.1 Test Environment

For the experiment we used Morae software created by TechSmith [6]. The Morae package contains Morae Recorder, Morae Remote Viewer and the Morae Manager. The Recorder captured video of the screen, user (including audio) and system events, which are automatically synchronized and each action is recorded (such as keystrokes and mouse clicks). The Remote Viewer allowed us to view the user test while the user was performing the test. The Manager application allowed us to analyze the user recording and create markers and segments based on when the user began and finished each of the tasks of the test. This provided us with accurate times for each task.

Four computers have Morae installed that monitored the system and users as they attempted the given tasks. Figure 3 shows the lab set up. The Lab required video and audio setups and access to the Office 2007 and 2003 products. It was imperative that each computer be set up identically; therefore we have set up the four test machines with a second boot option that ran Windows XP and has only the necessary applications available (Morae, Office, Webcam, etc).

**Figure 3.** The Lab Setup

## 2.2 User Tests

The participants were given an introduction to the study they participated in and a brief run down of the different software packages that they would be using, prior to the test being conducted. Each participant was required to complete a pre-test survey (Appendix A) that provided us with their prior knowledge and comfort with the Word UI. Each user was given directions (Appendix C) and identical tasks (Appendix D) to perform. The directions walked them through the test, where they could find the documents and what they needed to open. The tasks described the actions the user was to perform on the test document that was provided for them. They were encouraged to verbally indicate what they were thinking and/or feeling throughout the test.

The participants were given five separate tasks that had them use different features of the office application. They were required to navigate the different features of the UI. Two of the tasks the users preformed were identical enough to each other that we disregarded one. We focused on the four tasks that were left. Task #1 required them to perform simple font alignments changes. Task #2 required them to access the spacing function in the paragraph features of the UI. Task #3 required the participant to insert bullets into pre indented text. Task #4 required them to change the format of an existing table.

The participants were given a post-test survey, containing open ended questions that allowed them to describe any successes or frustrations they may have experienced while performing the test tasks on the old and new UI's.

## 3. RESULTS

Upon analyzing the pre-test surveys from the 50 participants in the study, we found them to be ranging from the ages of 18 to 31 (49 undergraduate students and 1 faculty member). From those surveys we were able to conclude that all had used the previous versions of Microsoft Word 2003, 64% use the application 0-5 hours a week, 26% 5-10 hours a week, 6% 10-15 hours a week and 4% use the application 15 or more hours a week, so we have a range of those who are comfortable with it on a day to day basis.

The test asked the users to open the Word 2003 application and open a file. For the 2003 recordings we had a total of 94% (47) successful recordings, of those recordings 85% preformed the open function, the other 15% directly opened the file from their file folder. From those 85% the average time it took for them to click on the 'Open' function (either the icon on the toolbar or File → Open) was 4.06 seconds, with the high of 14.81 seconds and a low of 1.27 seconds. The times are shown in Table 1.

**Table 1.** Word 2003 Times (in seconds)

| Word 2003 | Open | Save | Save As |
|---|---|---|---|
| **Average** | **4.06** | **2.99** | **3.76** |
| Max | 14.81 | 5.02 | 11.41 |
| Min | 1.27 | 1.38 | 1.07 |
| Number of recordings performing the action | 40.00 | 5.00 | 46.00 |

The test then asked the users to perform some everyday tasks to manipulate the test document provided for them.

Once those tasks were complete they were then asked to save the file as "<userID>Test2003.doc", implying that they use the 'Save As' function. 10% of the 47 recordings used the save icon on the toolbar and 97% did a 'Save As'. There is some overlap. 8% of those who preformed a straight save, continued on to locate the 'Save As' option. Of the 10% it took an average of 2.99 seconds to find and select save. Of the 97% it took 3.76 seconds to find and select 'Save As', as seen in Table 1.

That concluded the first part of the test. Next the users were required to restart their test machine and choose the boot option that would load the operating system that had the 2007 version of Microsoft Office. Unfortunately, for reasons we couldn't define, the logon to the 2007 boot took approximately five to ten minutes. Fortunately the participants were willing to remain patient with the systems and the tests.

After evaluating the pre-test surveys, we discovered that 18% of the participants have used the 2007 version of Microsoft Word. All participants that have used the new UI indicated that they have used it less than 5 hours a week, though with less than five hours a week of use time. One participant had indicated that the Office 2007 suite is the available Suite on their computer, thus they anticipate using the new UI in an increasing amount in the future.

Of the 50 participants that took the test, 86% of the recordings of the participants test sessions were successful. Once again the users were asked to open the Word 2007 application first and then open the file. This created interesting results. 88% were able to find 'Open', although on average it took them 28.17 seconds, with a high of 115.90 seconds and a low of 4.63 seconds, as shown in Table 2. On one occasion, the recording showed the user spending up to 3 minutes and 52 seconds trying to find the open function, and at the 3:52 mark, the user gave up trying to find open thus concluding that particular participants test, leaving 42 recordings.

**Table 2.** Word 2007 Times (in seconds)

| Word 2007 | Open | Save | Save As |
|---|---|---|---|
| **Average** | **28.17** | **4.43** | **25.51** |
| Max | 115.90 | 15.32 | 138.32 |
| Min | 4.63 | 1.63 | 2.99 |
| Number of recordings performing the action | 38.00 | 20.00 | 41.00 |

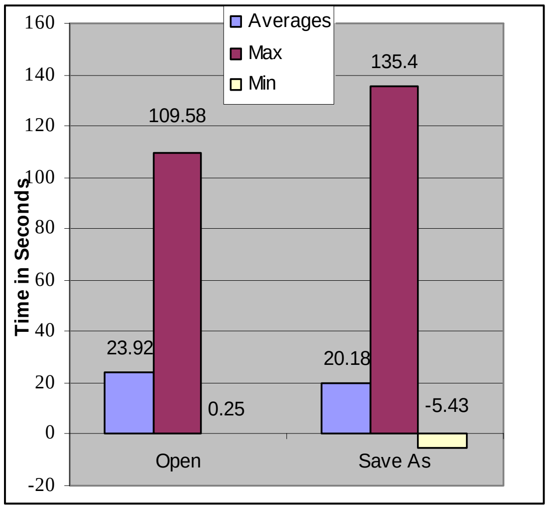


**Figure 4.** Time Differences

Figure 4 shows that in the new 2007 application it took the participants on average 23.92 seconds longer to find the 'Open' function, with a high of 109.58 seconds and a low of .25 seconds (disregarding the users who did not use the UI to open the file). Figure 4 also shows that the user continued to have difficulties finding 'Save As'. In the Word 2007 UI the 'Save As' and 'Open' icons are on the same menu, shown in Figure 5.

The user took, on average, 20.62 seconds longer to find 'Save As' on the new 2007 UI, with a high of 135.4 seconds, and -5.43 seconds as a low (this user preformed the save faster using the 2007 UI). There were a total of 2 users that performed the 'Save As' faster in the new UI. In the 2007 UI test there was a larger number of people who chose the save option that is located by default on the Quick Access Toolbar (QAT), The QAT's location is indicated in Figure 6.

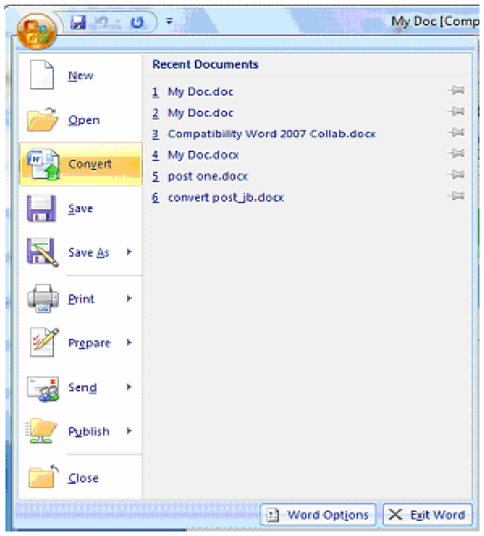


**Figure 5.** The 2007 Main Menu

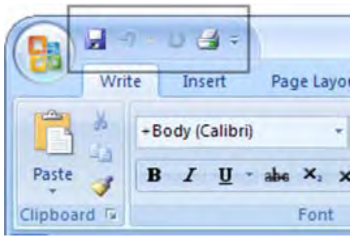


**Figure 6.** The Quick Access Toolbar

We believe this is because it was the only option that they could see/find quickly enough that would do what the participant wanted. Once the participant realized it was not the right option, they began looking for the 'Save As' option. One participant gave up on finding the 'Save As' option altogether.

The user requiring 135.4 seconds to find the 'Save As' option did not use the new menu, from Figure 4, to find the open function. The participant proceeded to place the icon on the QAT. The QAT does not allow the option to place a 'Save As' icon for quick access. In opening a file in this manner the user did not have the experience of opening the menu that 'Save As' resided on. The high for someone that did not use the QAT was 58.35 seconds, which is still significantly high.

It was apparent that there was a wide range of users based on their familiarity with the UI, and their comfort in transitioning from one UI to the other. Table 4 shows the times to complete the tasks in 2003 and Table 5 shows the times in 2007. Analysis of the task times displayed in the tables that it is evident there is a notable separation between the minimum times and the maximum times. This indicates that there were users familiar or had never preformed that task with the old UI and users unfamiliar with the UI. For the 2007 UI, that does not apply. 18% of the participants have used the 2007 UI previously. This proves exposure to the UI but not necessarily knowledge or comfort with it. Those that have used the new UI have been exposed infrequently and for short periods of time.

**Table 4.** Word 2003 Task Times (in seconds)

| Word 2003 | Task 1 | Task 2 | Task 3 | Task 4 |
|---|---|---|---|---|
| **Average** | **19.02** | **18.60** | **14.29** | **42.36** |
| Min | 10.81 | 8.26 | 3.54 | 10.33 |
| Max | 30.72 | 57.34 | 70.21 | 112.37 |

**Table 5.** Word 2007 Task Times (in seconds)

| Word 2007 | Task 1 | Task 2 | Task 3 | Task 4 |
|---|---|---|---|---|
| **Average** | **21.9** | **33.02** | **15.98** | **51.97** |
| Min | 5.91 | 7.88 | 2.8 | 5.32 |
| Max | 91.79 | 120.06 | 83.55 | 300.65 |

The lower numbers, shown in the Min rows of Tables 4 and 5, indicate that some users can find things easily in the new UI whereas others experienced extreme difficulty in finding the desired action. The participant that produced the data of 300 seconds spent all of those seconds trying to find the desired function. This participant eventually gave up looking and continued on with the test without finding the function they were looking for. Figure 7 visually shows the difference in the average times from one UI to the other. The positive number indicates that the user did the task, on average, faster with the 2003 UI, the negative numbers indicate that the user did the task faster in 2007.

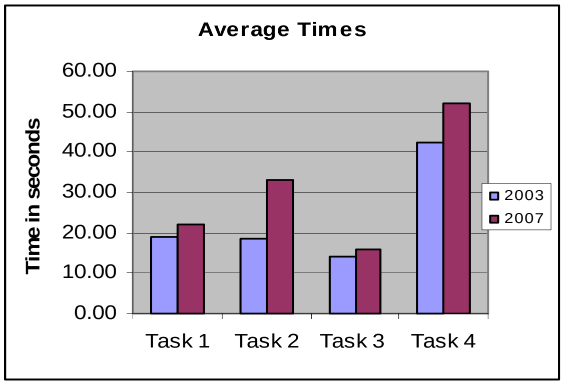


**Figure 7.** Time Differences

Upon analysis of the post test surveys, we discovered that 4% of the participants felt they did not complete the tasks in Word 2003 with ease. The comments the user gave were prominently focused on difficulties with completing Task #4. In Word 2007 36% felt they did not complete the tasks with ease. That 36% commented that the interface was unfamiliar to them and they had difficulties finding the desired functions. 34% of the participants felt the transition from the old UI to the new UI to be difficult. Their comments were that they were more familiar with the old and the new was too confusing at first use. The main features of the UI that the participants did not like were the way the menus were taken out and put into the ribbon and context tabs. They had a difficult time finding what they were looking for, though the majority of the participants liked the new look of the UI such as the colors and appearance.

The participants were just about split evenly on some aspects of how they felt about one UI over the other. Table 4 shows that a majority of the participants felt that 2003 was a faster, smoother and easier UI.

**Table 4.** User opinions on the UI's

| | Faster | Smoother | Easier |
|---|---|---|---|
| 2003 | **52%** | **82%** | **54%** |
| 2007 | 48% | 18% | 46% |

The participants were asked to choose which UI they preferred when performing a certain action, such as saving a document, or manipulating text. The results were varied, as seen in Figure 8. Given the choice between the new UI and the old UI, the user was asked which UI they would prefer to use. Only 40% preferred to use the old UI.

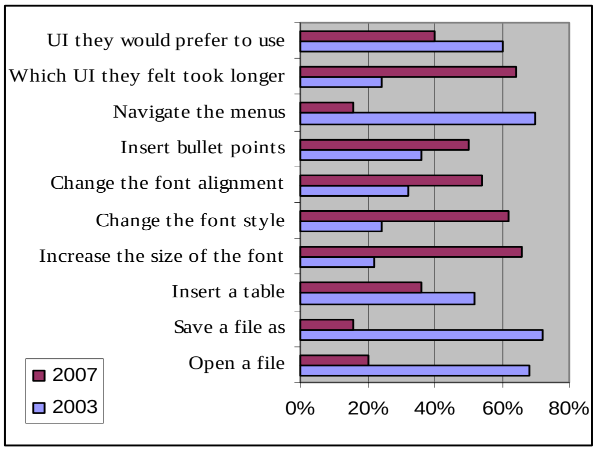


**Figure 8.** User Preferences

All participants noticed the context tabs throughout the test, but approximately 42% prefer the old menu toolbar styles, 6% felt that they liked both styles, 28% preferred the new style and 24% felt once they got more time with the UI they would prefer the new style over the old.

Where the mini-tool bar is concerned 32% of the users didn't see it. Overall 24% said they didn't use it or like it, some said that it got in the way, while 44% had favorable opinions of it and 12 % used the mini toolbar.

## 4. CONCLUSION

The results above showed, users on average spent more time finding the functions they wished to use while using the newly designed UI. From the participants' comments from the post-test survey, we concluded that the time difference is due to how the traditional menus were taken out and put into the new Ribbon and context tabs and the complete change to the UI design. The new organization of these tabs proved to be confusing and difficult to navigate to the users causing difficulties with the new UI, though many of the participants indicated that over time the UI will become easier to use. This is an opinion that we can agree with, typically everything should become easier to use over time. The point to these drastic changes was that the new UI is going to be easier to use, regardless of the users knowledge or comfort of the Office UI. The test results show that the new UI is in fact not easier for a new to the UI user or a user that has already been exposed to the UI. For a user to become comfortable with the new UI they will have to learn how to use its different features.

Regarding Fitt's law and the larger images with text on the icons located on the ribbon, the participants did notice the larger icons, and the added text, but the way they are organized on the tabs were confusing to many and the tabs weren't as helpful as anticipated. Where the Mini-tool bar is concerned, the users were split on whether it is effective or not. Some felt it got in the way and others felt that it made doing font related functions easier and liked that it showed up near the newly highlighted text.

## 5. FUTURE WORK

We feel more in-depth testing is still needed. A good test would be to find a large enough group of users that have had significant amount of time with the new UI and feel as comfortable with it as the do with the old UI. The tasks would be modified to focus on a more specific feature of the new UI, for example picking one of the new design concepts and requiring a user to access that feature of the UI. A researcher may choose to focus on a study with a group of users that have not used the old UI or the new UI. The test would focus on performing task with the new UI and show how an inexperienced user would respond to the new UI. The UI is still brand new and there will be more things to discover that can be studied, examined and questioned.

## 6. ACKNOWLEDGMENTS

We would like to thank the faculty of Computer Science Department of Saint Mary's University for all their assistance in setting up the labs and finding participants.

## 7. REFERENCES

[1]. J. Harris (2006) Designing against a Degrading Experience, Online Retrieved February 8, 2007 http://blogs.msdn.com/jensenh/archive/2006/03/02/542118.aspx

[2]. J. Harris (2006) Giving you Fitt's, Online. Retrieved February 8, 2007 http://blogs.msdn.com/jensenh/archive/2006/08/22/711808.aspx

[3]. J. Harris (2005) Its All about Context, Online Retrieved February 8, 2007 http://blogs.msdn.com/jensenh/archive/2005/09/16/468365.aspx

[4]. J. Harris (2005) Saddle up to the MiniBar, Online. Retrieved February 8, 2007. http://blogs.msdn.com/jensenh/archive/2005/10/06/477801.aspx

[5]. Microsoft Corporation (© 2007) Office Professional Home Page, Online Retrieved February 8, 2007 http://office.microsoft.com/en-us/suites/FX101674091033.aspx

[6]. TechSmith Corporation. (© 1995-2007) Morae Features Overview. Online Retrieved February 8, 2007. http://www.techsmith.com/morae/features.asp

[7]. P.M. Fitts, The information capacity of the human motor system in controlling the amplitude of movement, *Journal of Experimental Psychology* 47 (1954) (6), pp. 381–391.

## APPENDIX A: Pre-Test Survey

User ID:______

# Microsoft Word 2007 Pre-Test Survey

***This survey is to help us get a better understanding of your knowledge, familiarity and comfort with the Microsoft Word applications.***
***Please respond to the following questions and statements as honestly as you can.***
***Your feedback is extremely valuable to us and much appreciated.***

Are you a Saint Mary's Student? Yes No

If Yes: What is your major? ____________________________________________

Are you a Faculty of Staff Member? Yes No

If Yes: What is the department you teach or work in?_______________________

What is your gender? Male Female

What is your age? ______________

---

Have you ever used Microsoft Word 2003? Yes No

How often do you use the Word Application over the course of a week?
0-5 hours 5-10 hours 10-15 hours 15+ hours

Have you ever taken a course, or anything similar, that taught you how to use Word2003?
Yes No

Have you ever used the new Word 2007? Yes No

How often do you use the Word Application over the course of a week?
0-5 hours 5-10 hours 10-15 hours 15+ hours

Have you ever taken a course, or anything similar, that taught you how to use Word2007?
Yes No

If you have not used either Word 2003 or 2007, what word processing application do you use?
_____Word Perfect
_____Open Office.org Writer
_____Microsoft Works
_____Notepad

## APPENDIX B: Post-Test Survey

User ID:__________

# Post-Test Survey

Were you able to complete all the tasks using Word 2003 with relative ease?
Yes No

If no, describe the problems you had (Please be as descriptive as possible.)

Were you able to complete all the tasks using Word 2007 with relative ease?
Yes No

If no, describe the problems you had (Please be as descriptive as possible.)

If you experienced trouble with Word 2007, do you believe that with time it would get easier to use? Yes
No

Did you find the transition from using 2003 to 2007 difficult? Yes No
Please explain why you answered yes or no.

What feature do you like best about the Word 2003 version? Why?

What feature do you like least? Why?

What feature did you find that you liked best about Word 2007? Why?

What feature did you find that you liked least? Why?

In Word 2007, if you saw and used the Context tab, were they useful, or do you prefer the old menu toolbar? Please explain your answer.

In Word 2007, if you saw the Mini-Toolbar, did you use it? Please explain your answer

Continued on other side (next page).

| | | | |
|---|---|---|---|
| Which application did you feel was: | Faster: | Word2003 | Word2007 |
| | Easier: | Word2003 | Word2007 |
| | Smoother: | Word2003 | Word2007 |

| **Which version of Word did you like best when . .** | **Application Preferred** | | | |
|---|---|---|---|---|
| | **Word 2003** | | **Word 2007** | |
| Opening files | 1 | 2 | 3 | 4 |
| Saving files | 1 | 2 | 3 | 4 |
| Inserting a Table | 1 | 2 | 3 | 4 |
| Increasing text size | 1 | 2 | 3 | 4 |
| Changing Font style | 1 | 2 | 3 | 4 |
| Changing font alignment | 1 | 2 | 3 | 4 |
| Creating Bullet points | 1 | 2 | 3 | 4 |
| Navigating the interface(menus) | 1 | 2 | 3 | 4 |
| Which interface took longer to find the item to complete a task. | 1 | 2 | 3 | 4 |

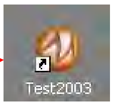
If you had the option which version would you prefer to use to create a document?

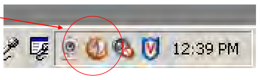
Word 2003  Word 2007  Other ____________________

Do you have any comments or concerns regarding the test?

## Begin Test

- Log on
- Find this icon in R:\
  - It must be Test2003.
- Once that has been clicked your task bar should look like this
- Open Word 2003

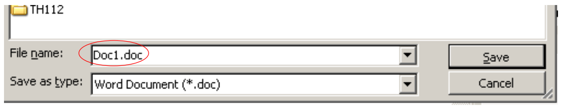




1

## Word 2003 Tasks

- Open the file located on your R:\TestFiles
- Complete the tasks given to you
- Once you are ready to save, change the file name to <UserID>_Test2003.doc
- Save it in R:\2003Test\



TH112

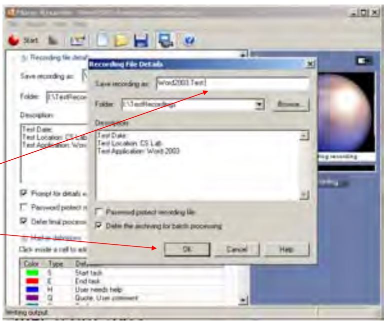
File name: Doc1.doc

Save as type: Word Document (*.doc)

2

## Saving the session recording

- This application will show up once you close Word 2003
- Enter your UserID after Word 2003
- Click OK



Recording File Details

Save recording as: Word2003 Test

Folder: I:\TestRecordings

Description:

Test Date:
Test Location: CS Lab
Test Application: Word 2003

Password protect recording file

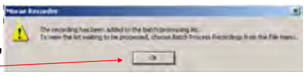
Defer the archiving for batch processing

| Color | Type | Description |
|---|---|---|
| 🟩 | S | Start task |
| 🟥 | E | End task |
| 🟦 | H | User needs help |
| 🟪 | Q | Quote, User comment |

3

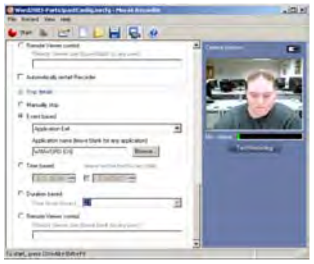
## Exit the Recording

- When this box shows, just click OK.
- Then close this window.



Morae Recorder

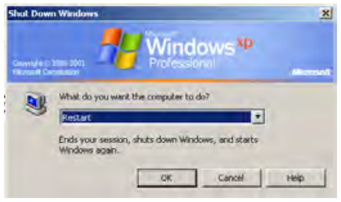
The recording has been added to the batch processing list.
To view the list waiting to be processed, choose Batch Process Recordings from the File menu.



4

## Shut Down the Computer

- Now restart the computer( Start → Shut Down)
- Select the restart option.



Shut Down Windows

What do you want the computer to do?

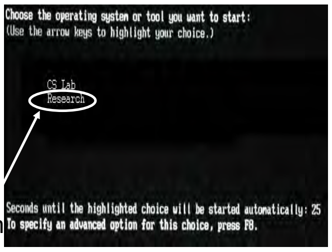
Restart

Ends your session, shuts down Windows, and starts Windows again.

5

## Restarting the Computer

- Once the computer begins to boot, this screen will show (it only will show for approx 10 seconds, so you have to pay attention)
- Use the arrows to select the Research option.

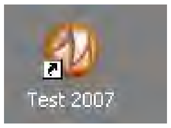


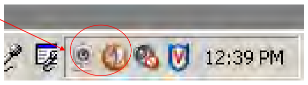
```
Choose the operating system or tool you want to start:
(Use the arrow keys to highlight your choice.)

    CS Lab
    Research

Seconds until the highlighted choice will be started automatically: 25
To specify an advanced option for this choice, press F8.
```

6

## Beginning the Word 2007 Test

- Log on
- Find this icon in R:\
  - It must be Test2007.
- Once that has been clicked your task bar should look like. .
- Open Word 2007
- Begin the test.

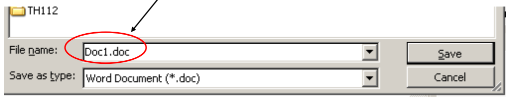




7

## Word 2007 Tasks

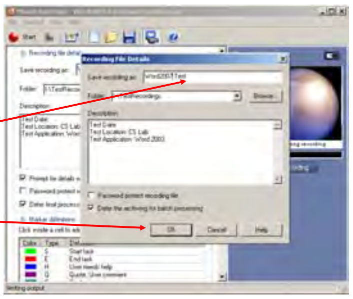
- Open the file located  in your R:\TestFiles
- Complete the tasks given to you from before
- Once you are ready to save, change the file name to <UserID>_Test2007.doc
- Save it in R:\2007Test



8

## Saving the session recording

- This application will show up once you close Word 2007
- Enter your UserID after "Word 2007 Test"
- Click OK
- Click OK on the next pop-up and exit the application



Test Date:
Test Location: CS Lab
Test Application: Word 2003

9

## Restart

- When ready to save, same principals as before though any 2003 should be 2007.
- Once again Restart the computer, but ignore this screen (it will automatically choose the CS Lab option)

10

_ Find the yellow highlighted text.
  - Change the alignment to right alignment; enter ID given to you for test. Hit return and insert the date.

_ Find the red highlighted text.
  - Align it to the center, Change the font to Georgia, and the font size to 20.

_ Find the green highlighted text.
  - Tab the beginning of each paragraph, and change the spacing to double.

_ Find the aqua highlighted text.
  - Select the section and insert numbered bullets

_ Find the table.
  - Change the alignment of all the cells to center, and change the theme of it using Table Properties.

# PerfiTrak – A Web-based Personal Finance System with Broad I/O Features

Matthew Lieder
Department of Computer Science
Winona State University
Winona, MN 55987

lieder_matthew@hotmail.com

## ABSTRACT

Since money is an integral part of most economies, it is usually important for people to keep track of their personal finances. Doing so by hand is difficult, and hence, with the advent of computers (and then the Internet), applications have been developed and refined to make it easier. Unfortunately, available applications are far from perfect. Observed deficiencies include poor interoperability, inflexible categorization of transactions, and lackluster online management. This paper presents a web application, PerfiTrak, which addresses these problems. Specifically, PerfiTrak includes robust import/export functionality (supporting CSV, QIF, and OFX formats), hierarchical tagging of transactions, a cohesive user-friendly interface (using PHP and AJAX), and easy offline access to data (through SOAP web services).

## Categories and Subject Descriptors

D.2.12 [**Software Engineering**]: Interoperability – *data mapping*.

H.3.5 [**Information Storage and Retrieval**]: Online Information Services – *data sharing, Web-based services*.

J.1 [**Administrative Data Processing**] – *Financial*.

## General Terms

Design, Economics, Security, Human Factors, Standardization

## Keywords

identity, authentication, open standard, open source, AJAX, personal finance, interoperability

## 1. INTRODUCTION

Money is an integral part of most economies, and so keeping track of its flow is a very important activity. Not only should the flow of money in businesses and groups be tracked, it is important that individuals keep track of their own money flow too. If one does not, for example, a big purchase might drain an account so much that not enough money is left to cover an upcoming bill. Buying a nice new TV and then not having enough money to pay the electric bill would be a very unfortunate (and embarrassing) occurrence. As another example, one might forget about money they owe a friend or, less likely, they forget about money owed to them.

So how does one go about keeping track of their money flow? Some recording is most likely already being done: receipts from stores, carbon copies of checks, a checkbook ledger, monthly statements from checking or debit/credit card accounts, pay stubs from one's job, and/or even just one's memory. What may be noticed from that list though is that 100% coverage is unlikely to be achieved, accuracy is far from guaranteed (especially in regards to human memory), and the sources are highly disparate and difficult to combine to get an overall picture. Not only that, but questions such as "How much has my car cost me this past year?" or "Can I afford to buy this car?" or "How much do I owe people?" take a lot of manual labor to fully answer.

With the advent of personal computers, a viable solution to such problems finally started to emerge. Applications were developed to allow people to store their transactions on their computers and then run queries to view charts and graphs showing their cash flow. With the advent of the Internet and subsequently online banking, it became no longer necessary to manually input every transaction. Transaction exchange formats were developed, allowing people to download transactions from their bank websites and then import them into their financial transaction software. Some applications even entirely automate that process, regularly automatically retrieving transaction data from banks so the user is saved from having to do the download/import process themselves.

Though the area of existing personal finance applications may seem to be mature and hardly lacking, we have observed clear deficiencies. One of the major ones is with respect to interoperability amongst the applications. Most existing applications allow importing of financial transactions in standard formats, but very few allow exporting to those same formats – presenting much difficulty if a person wants to switch to a different application or use multiple applications in tandem. In addition, options are usually very limited (if even existent) when importing. We have observed situations where banks provided transaction data which had the Description and Memo fields reversed and all data in visually-unfriendly uppercase lettering. With existing applications, corrections like those would be very time-consuming.

Another deficiency involves the categorization of transactions. The vast majority of applications allow assigning just one

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

*Proceedings of the 5th Winona Computer Science Undergraduate Research Seminar,* April 19, 2007, Winona, MN, US.

category per transaction, taken from a tree of categories. That was the way things were done before computers (as evidenced by filing cabinets), so it was natural to continue using that form of organization when applications started being developed. However, often there will be transactions that do not neatly fit into a single particular category, causing organizational data to be basically thrown away and thus limiting the power and usefulness of reports. For example, with the traditional categorization system finding out how much money was spent on a particular person would be impossible – transactions would have been categorized by where the money came from or what it was used for, not who it was spent on. As another example, suppose someone wants to find out how much money they spent on computer-related items in the past year – those transactions would have been under categories such as "Business" or "Entertainment", without anything identifying them as computer-related purchases. Not every application uses that category system, however: some web applications [4, 16] use something they call "tags", allowing any number of arbitrary labels to be attached to any transaction. That is an improvement on the single category system, making the aforementioned examples actually possible, but because the tags have no hierarchical structure (no relationships between tags) anything more than a small number of tags quickly becomes unwieldy and counter-intuitive.

Finally there is a need for web personal finance applications, especially with the usability and capabilities of their desktop equivalents. It is becoming common for people to have more than one computing device (such as desktops, laptops, PDA's, and cell phones). At the same time, people are often finding themselves using computers they do not own (such as at work, libraries, cybercafés, friends' houses, etc). Desktop applications (especially personal finance applications) almost always store their data locally, so it becomes very difficult for the same person to use the same desktop application across multiple computers – it quickly becomes practically impossible to manually keep track of all the disparate data. With the proliferation of Internet access, web applications present an appealing solution to that problem because they allow access to the data stored on them from the web browser of virtually any Internet-connected computer. While personal finance web applications already exist, their interfaces tend to be more static (like traditional web pages) than dynamic (like desktop applications) and they are often lacking in regards to features and customizability. They also are fully dependent on Internet access and provide no real means to manage transactions without it.

To remedy those deficiencies, we have developed a personal finance web application called PerfiTrak. For interoperability it supports both importing and exporting of common financial transaction formats, for transaction categorization it implements a fully user-customizable hierarchical tagging scheme, and for web access it provides a visually interactive and cohesive cross-browser/operating system web interface while also allowing transaction management when Internet access is unavailable. It was developed to potentially have a future as a commercially-viable project, with a target audience of people without much financial management skills or interest while at the same time offering functionality to appeal to others.

## 2. BACKGROUND

Many personal finance management applications exist, but two in particular rise above the rest both in terms of features and market share: Intuit's® Quicken® [9] and Microsoft Money® [14] both currently at version 2007. Both applications support account management, budgeting, and bill-paying, making extensive use of Internet connectivity to automatically download transaction details from supporting banks. For banks where automatic downloads are not supported, the applications can import transaction data in various industry-standard transaction exchange formats.

In addition, Personal finance applications are not just limited to desktop software anymore. Very recently websites such as moneytrackin' [16] and foonance [4] have appeared, which allow financial transactions to be recorded and simple reports to be generated from almost any Web-connected computer. Currently they have very simple interfaces compared to their desktop counterparts and are lacking in features, though as time progresses they are likely to improve.

## 3. SYSTEM OVERVIEW

Because of tight time constraints a "Rapid Application Development" (RAD) software engineering approach [13] was used, involving rapid, incremental prototyping and the use of as many existing software components as possible. Since this needed to be a commercially-viable project, with real users and a future, practicality and cost factors were also involved in decisions made.

The functional requirements of the application are as follows:

- The application must support both importing and exporting of common financial transaction exchange formats. Because of their widespread use, Comma-Separated Values (CSV) [26], Quicken® Interchange Format (QIF) [23], and Open Financial Exchange$^{SM}$ (OFX) [8, 19] are the formats that should be supported.
- The application must allow the user to create and organize categories in a hierarchical structure, while also allowing multiple categories to be assigned to a single transaction. The categories should be referred to as "tags", to distinguish them from ordinary categories and to be consistent with terminology used in applications with similar organizational systems.
- The application must be accessible from as many Internet-connected computers as possible. Because of their market share, it must work in both Internet Explorer 6+ and Mozilla® Firefox® 1+. It also must work under Microsoft® Windows® XP or newer, Linux®, and Mac OS® X or newer.
- The user interface should be easy for users to figure out without requiring them to read a manual. Users are unlikely to spend the time to read a manual and if they cannot figure something out on their own they will likely just give up.
- The user interface should be visually appealing, and utilize effects to make relationships amongst data easier to visualize while at the same time making the experience more enjoyable. That is an important part of

drawing in the market segment of people who lack personal finance skills and would not normally be interested in tracking their personal finances.

- An easy way for users to access and manage their transactions stored in the application when Internet access is unavailable (thus making the application inaccessible) must be provided. Users should be able to transfer their transaction data from the application to their computer while connected to the Internet, view and change their transaction data while not connected, and then when connected again be able to transfer their changes back to the application.

## 4. DESIGN DECISIONS

### 4.1 Web Support

One of the first things that had to be done was to choose the best way to present the application to the user over the Internet. There have been various solutions developed over the years specifically to help web applications attain the richness and responsiveness of desktop applications, including JavaScript (Web browser-embedded scripting language), Java Applets, and Adobe® (previously Macromedia®) Flash, so it was wise to make use of one of them. To help narrow down the search we came up with some criteria specific to our situation that the ideal solution must fulfill:

- **Cross-browser**: while Internet Explorer® is dominant in the web browser market, a number of other browsers (including Mozilla® Firefox®, Opera, and Safari™) are also in use and it would reduce our user base (and potentially generate bad press) to restrict our users to just one browser.

- **Cross-platform**: though Microsoft® Windows® is by far the most popular desktop operating system, there is a small (but often vocal) minority using other operating systems such as Linux® and Apple's® Mac OS®. For similar reasons as in the previous point we would prefer to not restrict our users to a single operating system.

- **Seamless with browser**: most web browsers allow the user to change the font size, zoom in on content, and resize the browser window, and supporting those functionalities would improve the usability of PerfiTrak. It would also be helpful to be able in the future to take advantage of the browser's printing capabilities.

- **Open standard**: since once we chose a solution it would be hard to change to a different one later, we want to make sure the chosen solution will be well-supported and improved upon in the future – with an open standard those are likely to happen, since they generally have a broad audience and are not tied to the fate of a single company. A lot of documentation is also usually available then, helping to minimize the learning and development time.

Given those criteria, both Java applets and Flash are out of the running (both do not work very seamlessly with the browser, and Flash is additionally not an open standard). One solution, not previously mentioned, does fulfill all those criteria however: AJAX (Asynchronous JavaScript + XML) [6]. It is not technically a new technology in itself, but is a term for a specific collection of already-existing technologies being used in concert:

- standards-based presentation using XHTML and CSS
- dynamic display and interaction using the Document Object Model
- data interchange and manipulation using XML and XSLT
- asynchronous data retrieval using XMLHttpRequest
- JavaScript binding everything together

Those technologies make it possible for a web page to be completely, dynamically altered without requiring a page refresh, since the communication with the web server happens quickly and invisibly in the background. Though relatively new, AJAX is already in use by many major web applications including Google's Gmail™, Maps™, and Docs™; Microsoft's® Hotmail® and Virtual Earth™; Yahoo!™ Mail; and many bulletin board sites around the world. Its technologies are also all open standards with documentation, tutorials, and examples in abundance, helping make development as easy as possible.

However, because AJAX involves many different technologies that are each complex in their own right, developing an AJAX web application of almost any size from scratch is prohibitively difficult. Fortunately, because of its popularity there exist many frameworks to make development using it much easier. Since PerfiTrak will involve lots of data processing, the ideal framework should work seamlessly with a server-side programming language and supporting database management system. It was decided that PHP© 5 [22] would be the best server-side language to use, given that it is free, very popular (and thus well-tested and maintained), and supports the object-oriented programming paradigm (known for its flexibility and maintainability). With PHP© 5 selected, MySQL® [17] for the database management system and Apache 2 [2] for the web server were natural choices, given how well they work together (and are supported). Given those choices the field of AJAX frameworks was narrowed down further, and one particular framework rose above the rest: Qcodo [24]. It is specifically intended for Rapid Application Development, makes use of AJAX functionality to approximate the interactivity and responsiveness of a desktop application, and is free and open-source, so it fit the bill quite nicely. One of its two major components is Qforms, a "completely object-oriented stateful event-driven architecture for HTML forms processing and rendering" that imitates the way desktop applications are programmed and separates the logic layer from the presentation layer (making the code easier to understand and the user interface easier to customize). Its other major component is the Code Generator, which intelligently analyzes the database schema and generates data access object (DAO) classes to easily (and more securely) interface with the database (saving a lot of time than if it was done from scratch). Qcodo is also open-source and easily modifiable/extensible, which will help us to overcome any user interface (UI) issues we encounter and make it much easier to create our own UI solutions.

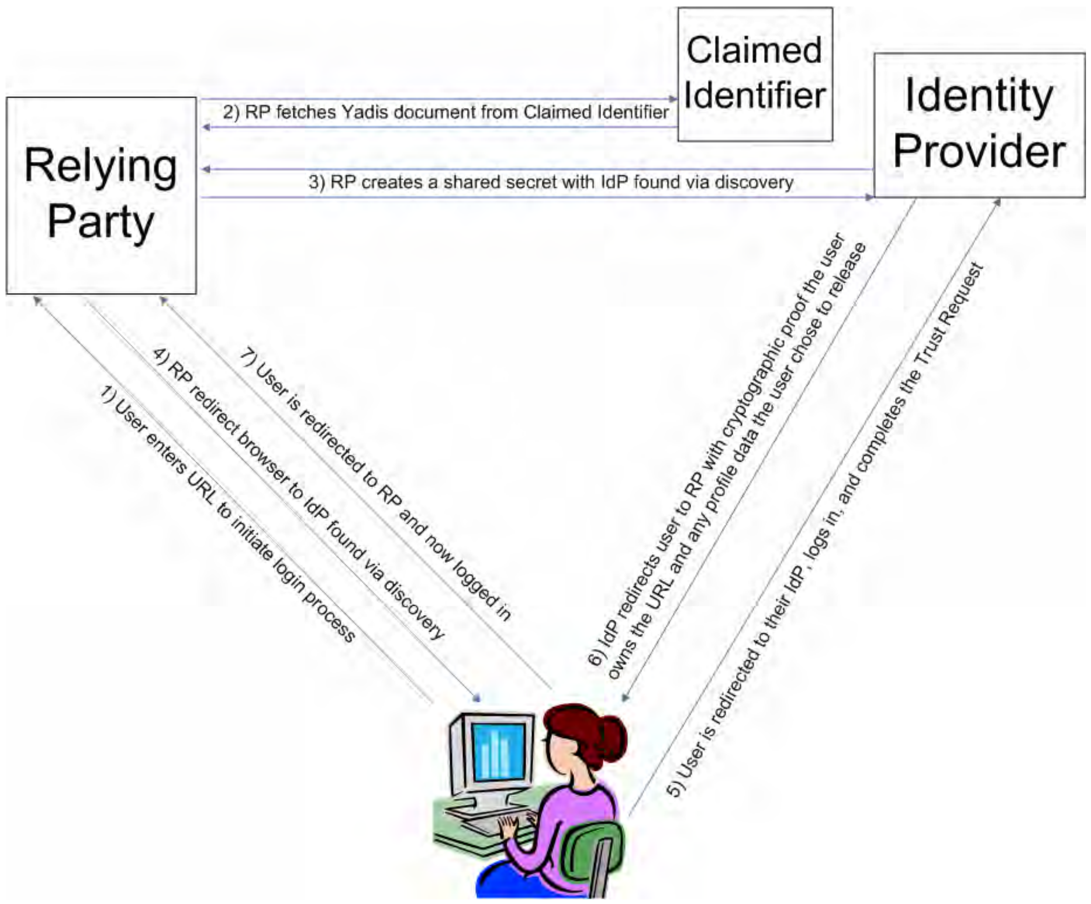

Claimed Identifier
Identity Provider
Relying Party
2) RP fetches Yadis document from Claimed Identifier
3) RP creates a shared secret with IdP found via discovery
1) User enters URL to initiate login process
4) RP redirect browser to IdP found via discovery
7) User is redirected to RP and now logged in
6) IdP redirects user to RP with cryptographic proof the user owns the URL and any profile data the user chose to release
5) User is redirected to their IdP, logs in, and completes the Trust Request

**Figure 1.** OpenID 1.1 protocol flow [25]

## 4.2 User Authentication

Since PerfiTrak is a web application, it must have a way to identify the different users accessing it so that the right data can be displayed. There would be a huge privacy issue if people's financial transaction data was open to the public. Therefore some sort of user authentication system had to be implemented, providing a secure means for users to identify themselves ("log in") to the web application.

### *4.2.1 Traditional*

The traditional way of authenticating users is by username and password. The user registers with a site by providing a unique word (the username), while also providing a private word or phrase that only they know (the password). When they go to log in to the site, they provide their username and password and if they match up to what the site has in its database, the site accepts them as who they say they are (the username). It is simple and thus easy to implement in a web application; however, there are unfortunately a number of drawbacks.

For one thing, unless the web application is running over a secure HTTP connection (HTTPS) the password is transferred in plain text, making it possible for any malicious person on any of the networks between the user and the web application's server to see the password. That can be solved by using HTTPS, but most web hosting companies charge extra for that feature and to keep operating costs low it is preferable to avoid needing HTTPS.

Another drawback stems from the fact that username/password authentication is widely in use. Since one person often has usernames and password at many different sites, to remember them all they usually use the same username and oftentimes even the same password for all of them. Because of that, people rarely change their passwords (a recommended security practice) since it would be too time-consuming changing passwords at every site or remembering where the password was not changed. Unfortunately those facts mean that if a person's username and password for one site is discovered (by the site's database being compromised or even just simple social engineering), their data at other sites where they use the same username and password is also compromised.

Finally, one more drawback is all the care that must be taken to make sure each user's password is stored securely and accessible by only that one person. A hacker seeing a few financial transactions is likely not going to cause any harm, but on the other hand a hacker seeing a few users' passwords could cause all sorts of harm (because of the previously-mentioned drawback).

### *4.2.2 OpenID*

The above-mentioned drawbacks are well-known in the industry, so fortunately numerous solutions have been proposed and developed. One such solution is OpenID, a decentralized single sign-on system [20, 25]. With that particular system, users are not required to come up with a username and password for every participating site. Instead, they register their username and password with a third-party “identity provider” of their choosing and are given a unique identifier (a URL or XRI) which they give to every site (called a “relying party”) they want to log in to. When the relying party is given the identifier, it communicates with the identity provider and then forwards the user’s browser to the identity provider’s web site. Once there, the user authenticates themselves to the identity provider (often using a traditional username and password). Finally, the identity provider forwards the user’s browser back to the relying party’s site along with confirmation that the user is the person to whom the identifier is registered. See Figure 1 for a more detailed description of the protocol flow.

The advantages of using OpenID in PerfiTrak are numerous. For users, they do not have to be concerned about coming up with a unique username and password if they already have an OpenID identifier (an ever-increasing possibility, with Microsoft®, AOL®, LiveJournal®, and WordPress all pledging support for the system [11, 12, 30]. Additionally, users that want to change their password just need to do it in one spot to be in effect for every site they use their OpenID identifier at. For the application, the advantage is that most of the user authentication details and security is offloaded onto identity providers. PerfiTrak only needs to store the users’ OpenID identifiers. As a bonus, PerfiTrak automatically benefits from security and functionality improvements done by identity providers, like HTTPS support (which many have), anti-phishing measures (like personal photos), and web browser integration (in development on Mozilla® Firefox® 3 [5], and Microsoft® Internet Explorer® 7 through Windows CardSpace™ [15]). Identity providers do not even have to use passwords for authentication: they could use soft tokens (like Windows CardSpace™), hard tokens (like RSA SecurID®), cognometrics (like Passfaces™ [21]), or even biometrics. All those choices and implementation details are left to the identity providers, without involving PerfiTrak at all.

Using OpenID in PerfiTrak does have some disadvantages as well, but measures have taken to greatly mitigate them. A big problem with single sign-on systems like OpenID is that phishing attacks (where users are misled into revealing their passwords by being presented with nefarious facsimiles of the real login page) have a much greater potential of causing harm [1]. Another problem is that since users can use any identity provider that they want, their particular choice might have lax security or poor reliability. To help prevent those problems PerfiTrak recommends certain well-respected and highly-secure identity providers (specifically JanRain’s® MyOpenID [10] and Verisign’s® Personal Identity Provider [28]), and provides a way to add a secondary OpenID identifier that can be used in the circumstance that a user’s primary identifier’s identity provider is not accessible.

## 4.3 Interoperability

A major component of PerfiTrak is its ability to import and export transaction data using major financial transaction exchange formats, enabling it to process statements from banks and interoperate with existing personal finance applications. Each format has its own unique advantages and disadvantages, presenting many interesting challenges when incorporating them into PerfiTrak.

### *4.3.1 CSV*

Probably the first format used widely to store financial transactions was CSV, Comma-Separated Values. It is a textual format where each line of text generally contains all the information for a single transaction, with information fields usually separated by commas and sometimes enclosed by quotation marks. The first line is usually a header, describing what information the various fields contain. An example can be seen in Figure 2.

```
Date,Description,Amount,Running Bal.
01/24/2007,Beginning balance,,"712.30"
01/25/2007,"WMATA SMART BENEFITS","-5.00","707.30"
01/26/2007," ATM DEPOSIT","983.37","1690.67"
```

**Figure 2.** CSV from Bank of America’s® online banking site

CSV’s main benefit is that it is generally very simple and easy to understand, able to be opened by spreadsheet software and easily visualized and edited. Unfortunately, the generic format itself is only loosely consistent (fields might or might not be enclosed by quotation marks, or may be separated by characters other than commas) and the format when applied to financial transactions is an absolute mess. As seen in Figure 1, the basic fields required to describe a transaction are its date, its description, and its amount. However, other banks have their own fields. For example, NetTeller® (see Figure 3) stores only the absolute value of the amount in its Amount column and has a separate column of fields label “CR/DR” wherein a “DR” means the money was leaving the account and a “CR” means the money was entering the account.

```
Posted Date,Serial Number,Description,Amount,CR/DR
01/02/07,,"POS DEBIT",0000000100.80,DR
,,"GOOGLE *DreamHost.com",,
01/02/07,3039,"CHECK",0000004000.00,DR
01/05/07,,"POS DEBIT",0000000021.37,DR
```

**Figure 3.** CSV from a NetTeller® online banking site

Additionally, it does not follow a strict one-line-per-transaction rule. Transactions which have additional information attached (usually referred to as the “Memo” field) put that info in a second line.

So far the differences have been relatively minor. However, some banks differ greatly in one important area: their CSV files lack headers. If all fields meant the same thing and were in the same order that would not be such a big problem, but unfortunately that is not the case. As shown in Figure 4, Wells Fargo® (a major US bank) skips on the header, stores the amount in the second column, and puts the description in the last column.

```
02/16/07,-7.38,"*",,"CHECK CRD PURCHASE"
02/15/07,-1.95,"*",,"BILLMATRIX BILL PAYMT"
02/15/07,-5.99,"*",,"CHECK CRD PURCHASE"
```

**Figure 4.** CSV from Wells Fargo’s® online banking site

That greatly increases the difficulty of deciphering its contents. Other banks without headers in their CSV's even differ in other ways; for example, TD Canada Trust skips the Amount column and instead has two mutually-exclusive columns for credit and debit. While those issues are difficult, we were able to develop algorithms to correctly deal with the vast majority of them. However, banks could change their CSV output at any time or users might try to import CSV's from previously-unknown banks, so it will require regular maintenance to keep accurate.

Unfortunately even bigger issues arise when trying to support CSV formats from foreign banks. Semicolons might be used instead of commas to separate fields, labels might not be in English, date formats rarely follow the US format of Month/Day/Year, and others. Being able to properly automatically deal with all those issues becomes virtually impossible, so a solution had to be devised. It was decided that when PerfiTrak does not recognize a particular field layout, it will present a visual interface to the user allowing them to manually identify what each column means and what format the date is in. In the worst case (neither PerfiTrak nor the user can properly identity the format's layout), PerfiTrak will recommend that the user instead use a different, more well-defined transaction format (QIF or OFX).

### *4.3.2 QIF*

QIF, Quicken® Interchange Format, was specifically designed by Intuit for storing financial data, and thus is less ambiguous than CSV. Data is also stored textually, but a transaction spans multiple lines with each line being an explicitly labeled information field (see Figure 5). While not as easy for humans to understand as CSV is, the computer has a much easier time reading it since information fields are all individually labeled. Unfortunately QIF still shares a major problem with CSV: dates and amounts do not follow specific formats, instead varying depending on the language and country of the generating program [7]. Basic support for reading transactions from QIF's was easily added to PerfiTrak, though support for non-US date and amount formats is currently lacking.

```
!TYPE:CCard
D02/28/2007
MMIDTOWN FOODS WINONA MN
N2
PMIDTOWN FOODS WINONA MN
T-5.5
^
```

**Figure 5.** QIF from Capital One's online banking site

### *4.3.3 OFX*

As a response to the ambiguity and diversity of existing forms of storing financial data, in 1997 Microsoft®, Intuit®, and CheckFree® announced that they were working together to create a single, unified specification called Open Financial Exchange$^{SM}$ (OFX) for the exchange of financial data over the Internet. It is a very rich format, supporting the storing of almost any sort of financial data out there. A primary goal of the format was to enable direct client-server communication between financial software and financial institutions, though it also works fine (and is often used) for static storage of transactions like CSV and QIF are used. It is much more precise than either of those other two formats, with dates stored in international ISO format and language and currency explicitly listed. It uses SGML or XML as the base format, making it very easily read by computers. In addition it specifies that each transaction should be given a unique identifier, making it easier for finance applications to recognize if a user is importing the same transaction more than once. See Figure 6.

```
...
<STMTTRN>
        <TRNTYPE>DIRECTDEP</TRNTYPE>
        <DTPOSTED>20050722120000</DTPOSTED>
        <TRNAMT>450.00</TRNAMT>
        <FITID>48397299</FITID>
        <NAME>ABC Manufacturing Inc</NAME>
        <MEMO>Direct Employer Deposit</MEMO>
</STMTTRN>
...
```

**Figure 6.** OFX excerpt from format documentation

Further analysis of OFX output from banks led to the discovery of a major complication however. It was found that banks usually store bank account or credit card numbers directly in the file. For desktop personal finance applications that is not a big issue, since all of the user's data remains on their single computer. On the other hand, web personal finance applications (including PerfiTrak) process and store data in a centralized location not under the control of the user. Therefore, extreme care must be taken by web applications to prevent the possibility of that info getting into the wrong hands. Even though PerfiTrak does not actually store account or card numbers in its database, there is still a short period between the user uploading the OFX file and the application reading the transactions from it where the raw data (with the account or card numbers) could potentially be read. Since currently PerfiTrak does not use a secure HTTP connection and it is running on a shared server (increasing its attack profile), it was decided that it would be best for OFX support to be postponed until those things can be rectified.

## 4.4 Storing Tags in the Database

One challenge while designing PerfiTrak was finding the best way to store each user's hierarchy of tags in the MySQL® database we were using. There are two main models for storing trees in SQL databases: adjacency list and nested set [3]. In the adjacency list model, each node is stored in the database with the ID of its parent node. In the nested set model, each node is stored in the database with special "left" and "right" numeric fields that are populated using a modified preorder traversal algorithm (see the reference for more details). The adjacency list model is the simplest, but it makes determining descendents of nodes very difficult. In the nested set model, finding ancestors and descendents is very easy (for the latter, just look for nodes whose left and right fields are greater than the node's left field while less than the node's right field). However, when nodes are added, moved, or deleted while using that model, other nodes' left and right fields must be updated to reflect the change in the model: a major complexity and inefficiency. The nested set model still would have been acceptable for PerfiTrak's usage, except for one particular issue that came up: changing the parent of a node (a needed ability, since the user can completely customize their tag structure) was an incredibly complex operation while using nested sets. Fortunately, there exists a hybrid model that uses an extra field in the adjacency list model to make finding descendents a lot simpler [27]. The extra field stores all the IDs of parent nodes as a single delimited string, allowing the finding of a node's descendents through a simple SQL query that performs a textual wildcard search. For example, finding all the children of a root node with an ID of 1 would consist of searching for every node whose field starts with '/0/1/'. That was the model we ended up using for PerfiTrak, trading a little data redundancy for a lot of simplicity.

### 4.5 Working Offline

In order for the user to be able to manage their transactions while not connected to the Internet, there needed to be a way for transaction data to be downloaded to a computer, to be viewed and edited, and to be uploaded back to PerfiTrak. Essentially two problems needed to be solved: how to have an offline interface, and how to synchronize the transaction data between that offline interface and PerfiTrak's online database.

When deciding what form the offline interface should take, a number of considerations had to be made. Not having much development time to spare, creating our own desktop application from scratch was out of the question since that would be a whole big project in itself. Thus, an existing desktop personal finance application had to be found. Having to be freeware and multi-platform (like PerfiTrak), many of the more popular personal finance applications were subsequently excluded from the search. Under active development and possessing a good plug-in architecture were also major considerations. At the end, the decision was made that an application called "Buddi" [18] would be used. It is an open-source Java personal finance and budgeting application, aimed at people with very little financial background. It seemed to share the same audience as PerfiTrak, and its interface was also designed to be simple and easy to use. Being coded in Java, it runs on a wide variety of operating systems. It also fully supports third-party Java plug-ins, for both importing and exporting. The only potential issue was that Buddi uses a single-category organizational system for transactions, but it was decided that was an acceptable limitation since people could always easily add more tags to transactions when they go back to PerfiTrak.

Now that an application was chosen for the desktop interface, a way had to be devised to somehow transfer transaction data between it and PerfiTrak. One way of course would be to just make Buddi able to import and export files in one of three transaction exchange formats that PerfiTrak already supports (CSV, QIF, or OFX). However, that would require a lot of manual work on the part of the user and be far from simple to do. Thus, a better way needed to be found. Ideally, with just a few button clicks Buddi should be able to automatically get transaction data from PerfiTrak's database and then send the changes (or vice versa). Fortunately, PHP© 5 and Qcodo have a solution to that technical issue: SOAP web services. SOAP (also known as Simple Object Access Protocol) [29] is an XML-based messaging framework, containing a convention for representing remote procedure calls (RPC) and responses. PHP 5 and Qcodo specifically provide wrappers around that technology to make it easy to implement and use. In addition, Java has libraries that make it very simple to interface with SOAP web services. However, an issue arose from doing it that way: since the user would be accessing PerfiTrak invisibly from inside Buddi, it would not be possible to use its OpenID authentication because there would be no good way to send the user to their identity provider (no browser in use). Our solution was to automatically give every user a randomly-generated "pass code", having them input their OpenID and pass code in Buddi and then sending that info along with the SOAP RPCs. In the end, we created a SOAP web service on the PerfiTrak site which had a number of appropriate RPCs (GetTransactions, AddTransaction, etc.) and then created import and export plug-ins for Buddi (it did not explicitly support a "synchronize" type of plug-in) which asked for the user's OpenID and pass code (along with a few options, like which accounts and what dates) and made the appropriate RPCs to transfer transaction data to/from PerfiTrak's database.

## 5. IMPLEMENTATION

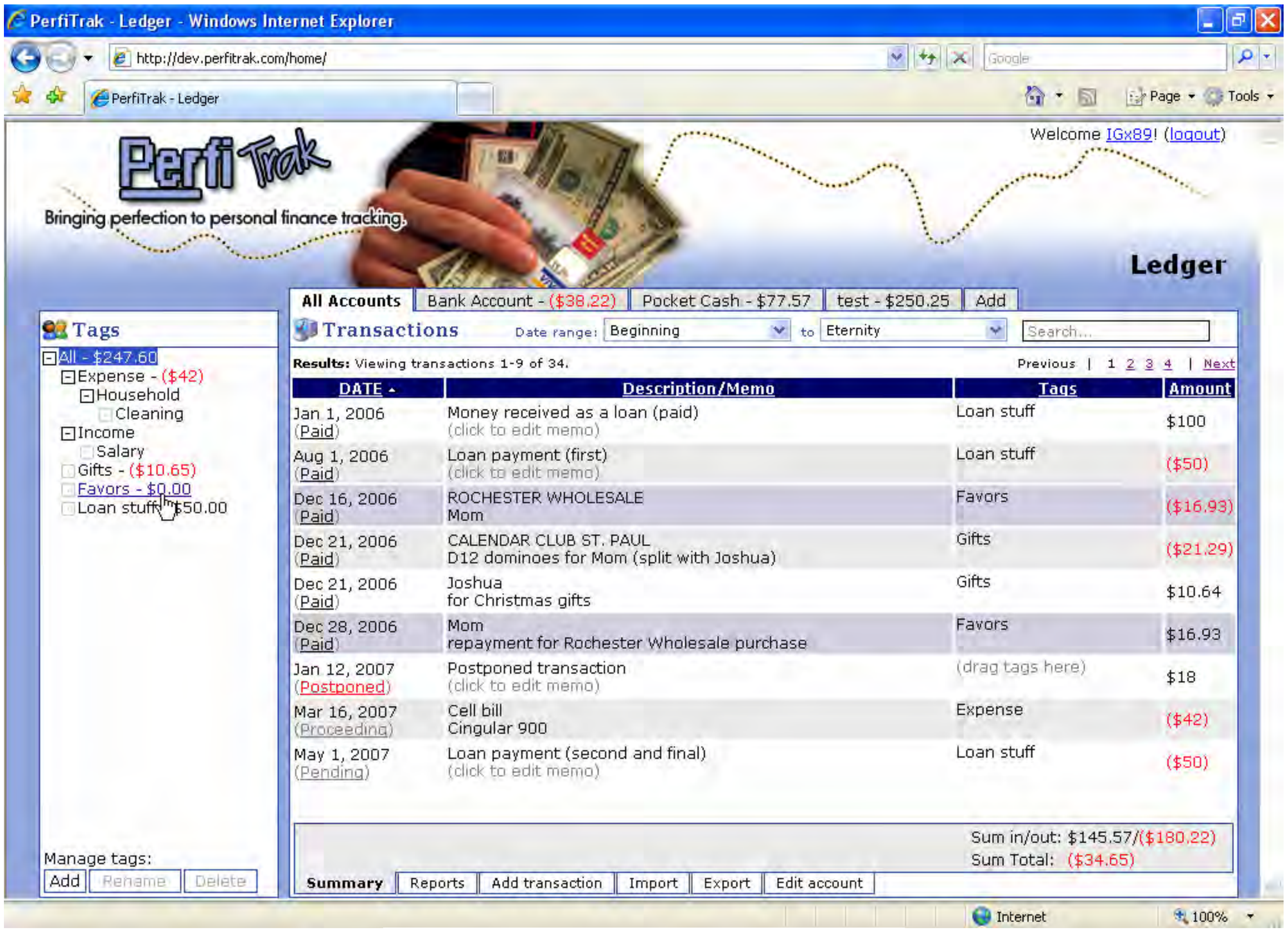
The implementation went relatively smoothly, save for a couple minor issues. We decided to model the interface similar to those of many desktop applications that involve navigating amongst categorized data, with the organizational structure (the tags) on the left, searching/limiting and additional organization data (accounts) on the top, extra info and functions (importing, exporting, adding transactions, etc.) on the bottom (as tabs), and the main data (the transactions) front and center. See Figure 7.

One issue was how to allow the user to assign tags to transactions. Most personal finance applications have a drop-down list to choose a category or a simple textbox to type in tags, but the former wouldn't work in our case and the latter seemed to be too tedious and error-prone. Eventually we decided to use drag and drop, with the user dragging a tag from the list of tags on the left onto the transaction they want that tag to be on.

Another issue was how to allow the user to organize their tags, specifically how to let them re-arrange the hierarchy. The easiest way for the user was unquestionably allowing them to just drag and drop tags onto their new parents (like many tree components on desktop applications allow), but unfortunately the tree control that Qcodo had didn't have support for that. Eventually we decided it was worth the effort to implement that support ourselves, so a big chunk of development time (about 30 hours) was spent on making Qcodo's tree control (QTreeNav) allow drag/drop re-arrangement.

## 6. SUPPORT FOR FUTURE DEVELOPMENT

The choice to use PHP© 5 and Qcodo proved to be a big help in making the application implementation easy to comprehend, making maintenance down the road easier to perform. It is easy to re-use code and components (like the tree listing of tags or the listing of transactions) too, so extending functionality in the future could be done very cleanly with minimal impact on existing


PerfiTrak - Ledger - Windows Internet Explorer

http://dev.perfitrak.com/home/

Bringing perfection to personal finance tracking.

Welcome IGx89! (logout)

Ledger

All Accounts | Bank Account - ($38.22) | Pocket Cash - $77.57 | test - $250.25 | Add

Tags

- All - $247.60
  - Expense - ($42)
    - Household
      - Cleaning
  - Income
    - Salary
  - Gifts - ($10.65)
  - Favors - $0.00
  - Loan stuff - $50.00

Manage tags: Add | Rename | Delete

Transactions Date range: Beginning to Eternity

Results: Viewing transactions 1-9 of 34. Previous | 1 2 3 4 | Next

| DATE | Description/Memo | Tags | Amount |
|---|---|---|---|
| Jan 1, 2006 (Paid) | Money received as a loan (paid) (click to edit memo) | Loan stuff | $100 |
| Aug 1, 2006 (Paid) | Loan payment (first) (click to edit memo) | Loan stuff | ($50) |
| Dec 16, 2006 (Paid) | ROCHESTER WHOLESALE Mom | Favors | ($16.93) |
| Dec 21, 2006 (Paid) | CALENDAR CLUB ST. PAUL D12 dominoes for Mom (split with Joshua) | Gifts | ($21.29) |
| Dec 21, 2006 (Paid) | Joshua for Christmas gifts | Gifts | $10.64 |
| Dec 28, 2006 (Paid) | Mom repayment for Rochester Wholesale purchase | Favors | $16.93 |
| Jan 12, 2007 (Postponed) | Postponed transaction (click to edit memo) | (drag tags here) | $18 |
| Mar 16, 2007 (Proceeding) | Cell bill Cingular 900 | Expense | ($42) |
| May 1, 2007 (Pending) | Loan payment (second and final) (click to edit memo) | Loan stuff | ($50) |

Sum in/out: $145.57/($180.22)
Sum Total: ($34.65)

Summary | Reports | Add transaction | Import | Export | Edit account

Internet 100%

**Figure 7.** Screenshot of PerfiTrak's main interface

components. Using Qcodo also proved a major benefit when changes to the appearance of the user interface had to be made, since the code was physically separate from the presentation markup (the XHTML and CSS). That also made it very easy to add support for user-customizable appearances ("themes"), of which we hope to have more of in the future.

In addition, while implementing the code for the importing and exporting of the various transaction exchange formats we followed an extensible structure (a "framework") that would permit new import and export formats to easily be added into the application. The code was separated into classes based on which format it was for and if it was doing importing or exporting, with each class extending a common super-class. Doing so allowed the exact details of how each format was implemented to be hidden from the code involved with the UI or interfacing with the database.

## 7. CONCLUDING REMARKS

PerfiTrak introduces a rich, easy-to-use web interface to the area of personal finance applications, hopefully enticing more people to keep track of their finances and make better, more-informed financial decisions. It does not restrict users' transactional data, allowing them to both import and export that data whenever they want, in a variety of common formats. It provides a robust organization system, allowing any number of fully-customizable hierarchical "tags" to be attached to individual transactions for better tracking and analysis. Finally, it allows users to access its interface (and thus their data) from almost any Internet-connected computer, and even provides a way for transaction management when the Internet is unavailable. We were able to fulfill our original goals and requirements, and now plan to offer the application to the general public at http://www.perfitrak.com/.

Out of general curiosity, while developing PerfiTrak we also kept track of the time spent researching and coding various sections of the application; see Figure 8. It took 5 hours to add OpenID support, 30 hours to implement drag/drop support on Qcodo's tree control that we used to display all the user's tags, 10 hours to design and implement the database model for the tags, 10 hours to add basic support for CSV importing, 13 hours to create an Import plug-in for Buddi and design the SOAP web service it used, and 92 hours on the rest. Those hours add up to 160 spent overall so far, with more work remaining to finish up a few more areas and fix any bugs that most likely will be found. Only one developer was involved, and those hours were done within a time period of just a little more than 3 months. It definitely took longer than we expected and prevented us from implementing all the features we had originally planned, but that was more because of us underestimating the magnitude of a project like this than the actual difficulty of doing it. Our choice to use PHP 5 and Qcodo were unquestionably big aids in helping us to get done what we were able to do in the time given.

| **Area** | **Hours** |
|---|---|
| OpenID implementation | 5 |
| Tree control drag/drop support | 30 |
| Tag database model | 10 |
| CSV importing | 10 |
| Buddi Import plug-in & interface | 13 |
| Other | 92 |
| **Total Hours:** | 160 |

**Figure 8.** PerfiTrak development time breakdown

## 8. FUTURE WORK

Personal finance management encompasses a wide variety of potential features, and so PerfiTrak currently only implements a small subset of what is possible. Part of the reason for that is lack of time and resources, but another part is that PerfiTrak aims to be simple to use and the number of features and complexity of a software application are generally directly proportional to each other. However, there still a few useful features that we would like to implement in the future:

- **Splitting transactions**: many existing personal finance applications allow the user to easily split a single transaction into multiple transactions, a time-saving feature, so it would be beneficial to also have that in PerfiTrak
- **Transfers**: often people find themselves transferring money between accounts (a common example being paying credit card bills), so making it possible for the user to deduct money from one account and put it into another in a single step would be helpful.
- **Budgeting**: one of the big reasons for keeping track of personal finances is to figure out how much money can be spent in various spending categories and make sure those guidelines are being followed. Somehow making it possible for users to set spending limits on tags and visualize how well they are following those limits would be a big benefit to users.
- **Charts and graphs**: currently PerfiTrak only allows users to see how much was spent in a particular area just by numbers, but having charts and graphs to help users visualize their transaction activity would be a lot friendlier and would allow more complex relationships and trends to be identified and analyzed.
- **Investments**: many people have money invested in areas such as stocks, bonds, and mutual funds, so having a way for users to keep track of that money would likely be very welcome.
- **Efficiency**: until now the vast majority of work on PerfiTrak was on making it work and adding features, with little time spend on optimizing areas for the best speed and efficiency. Doing so would help decrease the CPU load on the web server and the amount of bandwidth used, allowing more users to be able to use PerfiTrak simultaneously and making the experience quicker and smoother for them. One particular area we would like to improve is in the listing of transactions. We had to restrict it to only show 9 transactions at a time (with links to show more) since listing more than 20 or so transactions at a time caused the page loading times to be unacceptable. We feel that with some effort spent on reducing the amount of XHTML markup needed for each transaction we should be able to do much better.
- **Mobile device interface**: with an increasing number of people having Internet access on their cell phones, they are likely to desire to be able to use PerfiTrak from them. Though most cell phones have web browsers, unfortunately because PerfiTrak was designed for large screens and connections with a lot of bandwidth the experience would be virtually unusable on them. However, it is completely possible to create a simplified, low-bandwidth interface specifically for mobile web browsers that would at least allow users to view their transactions and add new ones.

## 9. ACKNOWLEDGMENTS

I would like to thank Dr. Joan Francioni, my project advisor, without whom this paper would never have been written. Thanks also go to the author of Buddi, Wyatt Olson, for having created an open-source personal finance application that so easily could be used for PerfiTrak's offline interface, and to the many users of Buddi who graciously provided samples of CSV files from many different bank sites for my analysis. Special thanks also must go to QuasIdea Development, LLC, specifically Mike Ho, for creating the brilliant Qcodo framework, and to the creators of the various UI controls PerfiTrak makes use of; without all them, PerfiTrak may never have seen the light of day. Finally, I cannot forget to thank Isaac Johnson, who stepped in at just the right time to provide a graphic design for PerfiTrak that far rivaled the one I had originally thrown together myself.

## 10. REFERENCES

[1] Anderson, Tim. "OpenID still open to abuse." IT Week. 5 March 2007. http://www.itweek.co.uk/itweek/comment/2184695/openid-open-abuse. Accessed 29 March 2007.

[2] Apache Software Foundation. "Apache." http://httpd.apache.org/. Accessed 2 March 2007.

[3] Celko, Joe. "Joe Celko's Sql for Smarties." San Diego: Morgan Kaufmann, 1999.

[4] foonance. "foonance.com." http://www.foonance.com/. Accessed 28 September 2006.

[5] Forrest, Brady. "Firefox 3.0 Requirements Are Out." O'Reilly Radar. 11 January 2007. http://radar.oreilly.com/archives/2007/01/firefox_30_requ.html. Accessed 2 April 2007.

[6] Garrett, Jesse James. "Ajax: A New Approach to Web Applications". Adaptive Path. 18 February 2005.

http://www.adaptivepath.com/publications/essays/archives/000385.php. Accessed 25 February 2007.

[7] Gribble, Bill et al. “QIF file format.” GnuCash. 6 August 2001. http://svn.gnucash.org/trac/browser/gnucash/branches/2.0/src/import-export/qif-import/file-format.txt. Accessed 1 April 2007.

[8] Horadan, Peter H, et al. “Method and system for transferring a bank file to an application program.” US Patent 5842211. 24 November 1998.

[9] Intuit. “Quicken 2007.” http://quicken.intuit.com/. Accessed 8 February 2007.

[10] JanRain, Inc. “MyOpenID.” https://www.myopenid.com/. Accessed 2 April 2007.

[11] Kveton, Scott. “OpenID: Signs point to momentum.” JanRain, Inc. 20 February 2007. http://kveton.com/blog/2007/02/20/openid-signs-point-to-momentum/. Accessed 29 March 2007.

[12] LiveJournal. “OpenID.” http://www.livejournal.com/openid/. Accessed 29 March 2007.

[13] Maner, Walter. “Rapid Application Development.” 15 March 1997. http://csweb.cs.bgsu.edu/maner/domains/RAD.htm. Accessed 6 February 2007.

[14] Microsoft. “Microsoft Money 2007.” http://www.microsoft.com/money/. Accessed 8 February 2007.

[15] Microsoft. “Microsoft Outlines Vision to Enable Secure and Easy Anywhere Access for People and Organizations.” Microsoft PressPass. 6 February 2007. http://www.microsoft.com/presspass/press/2007/feb07/02-06RSA07KeynotePR.mspx. Accessed 29 March 2007.

[16] moneytrackin. “moneytrackin’ – online accounting for real people.” http://mo.neytrack.in/. Accessed 28 September 2006.

[17] MySQL AB. “MySQL.” http://www.mysql.com/. Accessed 2 March 2007.

[18] Olson, Wyatt. “Buddi – Personal budget software for the rest of us”. http://buddi.sourceforge.net/. Accessed 26 December 2006.

[19] Open Financial Exchange. http://www.ofx.net/. Accessed 25 January 2007.

[20] “OpenID.” http://openid.net/. Accessed 6 February 2007.

[21] Passfaces Corporation. “Passfaces.” http://www.realuser.com/. Accessed 29 March 2007.

[22] The PHP Group. “PHP.” http://www.php.net/. Accessed 2 March 2007.

[23] “QIF Definition.” Intuit Inc. 1996. http://web.intuit.com/support/quicken/docs/d_qif.html. Accessed 25 January 2007.

[24] QuasIdea Development, LLC. “Qcodo.” http://www.qcodo.com/. Accessed 25 January 2007.

[25] Recordon, D. and Reed, D. 2006. “OpenID 2.0: a platform for user-centric identity management.” In Proceedings of the Second ACM Workshop on Digital Identity Management (Alexandria, Virginia, USA, November 03 - 03, 2006). DIM '06. ACM Press, New York, NY, 11-16. DOI= http://doi.acm.org/10.1145/1179529.1179532.

[26] Repici, Dominic John. “The Comma Separated Value (CSV) File Format.” Creativyst Software. 2006. http://www.creativyst.com/Doc/Articles/CSV/CSV01.htm. Accessed 21 February 2007.

[27] Shick, Trevor. “Varchar-based Nested Set.” threebit. http://threebit.net/tutorials/nestedset/varcharBasedNestedSet.html. Accessed 14 March 2007.

[28] VeriSign, Inc. “Personal Identity Provider.” http://pip.verisignlabs.com/. Accessed 2 April 2007.

[29] W3C. “Web Services Architecture”. 11 February 2004. http://www.w3.org/TR/ws-arch/. Accessed 12 April 2007.

[30] WordPress. “OpenID.” 6 March 2007. http://wordpress.com/blog/2007/03/06/openid/. Accessed 6 March 2007.

# Cheating Detection and Prevention in Massive Multiplayer Online Role Playing Games

Kevin Warns
Saint Mary's University
700 Terrace Heights
Winona, MN 55987
kmwarn04@smumn.edu

## ABSTRACT

Due to the amount of time needed to gain experience and items in massively multiplayer online role-playing games, some people have resorted to cheating to take a short cut to the higher levels of the game. This ruins the game play for people who abide by the rules. Game software companies need to deal with cheaters in a fast, effective and efficient manner to control the problem. This paper shows that the current methods of detecting and preventing cheating in older games are not sufficient. Newer games have fixed a few of the problems, but still have some to deal with. The problem lies within the client-server architecture, and how much control the client computer has over the game. This paper discusses the strategies employed by people who cheat, methods of detecting people who cheat, and how to prevent people from cheating.

## Keywords

Massively multiplayer online role-playing game (MMORPG)

## 1. INTRODUCTION

Playing massively multiplayer online role-playing games (MMORPGs) is a rapidly growing hobby [1]. With this growing market, there are many complications. The most common type of complication is that of cheating. This paper will focus on the cheating that is prevalent in the online game of Everquest [2] by using third-party programs that interact with the game. Using the classification scheme set forth by Randall and Yen, this kind of cheating would fall under "Cheating by Exploiting Misplaced Trust" and "Cheating by Exploiting Lack of Secrecy" [3]. "Misplaced Trust" deals with having too much information about the game on the client side, and "Lack of Secrecy" deals with how information is passed between server and client, and how it might not be secure.

## 2. BACKGROUND INFORMATION

MMORPGs are a computerized version of normal pen-and-paper role-playing games (RPGs), only on a much larger scale. While only a handful of people can play traditional RPGs together, a much larger number of people can simultaneously play MMORPGs. Even with this difference in number of players, many of the same concepts apply. When you enter the game, Everquest in this example, you create a character that will represent you in the game world. You usually choose a race from a variety of choices which range from normal humans to standard role-playing races such as dwarves and elves, to more exotic special races created for the game, such as lizard-men. You then would choose a specialty, or class. This would determine your role in the game world. It can vary from a magic-user, to a thief, to a warrior. Once you have created a character for yourself, you can enter the game world. Once in the game world, you are presented with a user interface consisting of customizable hot-buttons, chat windows, and many other windows. Using the keyboard and mouse, you can interact with other player characters (PCs), or computer-controlled players, called non-player characters (NPCs). The whole idea of the game is to interact with other players, to gain "experience points" which will make your character become more powerful, and to gain more power through items. Experience points are gained through killing certain NPCs. For every NPC, you gain a small amount of experience points. Those experience points fill up a, by default, unlabeled progress bar. The user-interface is customizable to show an integer percentage, but there is not an exact percentage visible to the player. Items are gained through checking the body of your now-slain foe for money and items.

The idea of why people cheat in these games is apparent: it takes a long time to create a character with lots of experience and gain powerful items. Game software companies feel they need to combat cheating to prevent the game being ruined for people who play by the rules [4]. This paper will answer the question of what can be done to make MMORPGs more secure and less prone to cheating

## 3. CHEATING STRATEGIES

There are many different ways users can abuse the game. One such method is by using third-party programs to take advantage of the information that is sent to the client. The first third-party program this paper will discuss does just that. It is for the game Everquest, and is called ShowEQ [5].

ShowEQ works by analyzing the network traffic for information relevant to Everquest. This information is in the form of packets that are sent to the client during game play. It then displays the information gathered from these packets on the screen on a 2-dimensional map. The information gathered includes the position of all PCs and NPCs, as well as information about both, such as level, class, race, and what items they are holding. In Figure 1, it shows the output of a typical ShowEQ window. On the left are all of the PCs and NPCs in a color-coded list (colors are based on the

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

*Proceedings of the 7th Winona Computer Science Undergraduate Research Seminar*, April 19, 2007, Winona, MN, US.

difficulty of the NPC), including all known information about them. On the right is a map view of the current area. The small dots on the map are the locations of the PCs and NPCs.

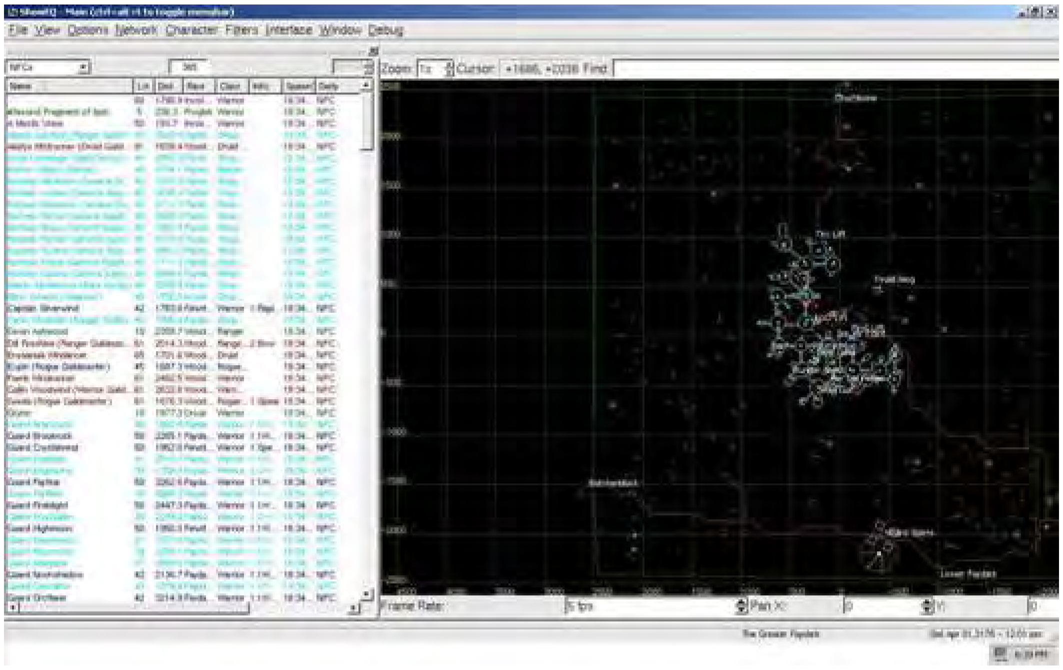

File View Options Network Character Filters Interface Window Debug
Zoom:
Cursor:
Find
Frame Rate:
Pan X:

Figure 1. PC and NPC Information and Map [6].

This is one security flaw of client-server MMORPGs. To have low-latency game play, that is, game play that feels like it is taking place in real time, there must be very little time between when the packet is sent and received, therefore the packet must not be heavily encrypted.

Caltagirone et al. do not go in-depth on the subject of encryption of packets in their paper on the architecture of MMORPGs [7]. They do not take into account that the encryption key must be stored on the client computer. This is another security flaw of MMORPGs using the client-server model. The client has to have the encryption key stored locally, therefore it can be found by the person on the client side. An example of this is the battle between Sony and the makers of ShowEQ, which has been very well documented [8] and will be discussed in Section 5.

A step up from ShowEQ is the program Macroquest [9]. It also is used to cheat in Everquest. The authors of Macroquest require the users to run a small program on each computer they wish to use Macroquest on, which generates two strings in the form of hashes. These strings are used to compile a version of the program that will work only on the computers whose hashes are included. This is done to prevent "everyone" from using the program, requiring only basic knowledge of computers and how to read directions. Macroquest is the successor of ShowEQ, and does all the things ShowEQ does, and more. It provides an in-game application programming interface (API) which is used to access data stored on the client side, as well as data objects. For example, the experience bar mentioned in Section 2 can now be viewed as a decimal percentage using this API and data objects. To view this information while playing the game with Macroquest running, the user would type "/echo ${Me.PctExp}" into the console, and the game would output the percentage of experience the character currently has. The "/echo" tells it to output the information to the console, the "Me" is the data object for the current character, the "PctExp" is the percentage experience, and the "${ }" surrounding it tells Macroquest that the information inside the braces is a variable. [10]

In addition to the API that Macroquest provides, it also gives a way of automating a character in game. It accomplishes this by use of a scripting language [11]. These scripts are called macros, and are the primary reason people use Macroquest. They make monotonous tasks repeatable without human interaction, which otherwise would require the player to sit at the keyboard. These Macros vary on complexity, ranging from simple "click one thing repeatedly" to more complex macros that will automate a character to gain experience and gather items unassisted [12].

To understand these macros, an example of the code that would be used to "click one thing repeatedly" would be discussed. This specific macro clicks a button to "combine" items gathered from NPCs into a new item. The more times you combine items, the better you get at it. On a successful combination, the new item appears on your cursor. If you fail the combination, the items are gone. In Everquest, these are called "trade-skills" and range from baking and brewing, to tailoring and blacksmithing.

The first line of the code, printed below, starts with "#event", which watches the chat output for the contents in the quotes. It then sets up the entry point of the program (Main method), and sets up a loop to place items from the cursor into the inventory. Once the cursor is empty, it left-clicks on the "Combine" button, checks if the #event happened, then loops back to empty the cursor. If the event did happen, the macro ends.

```
//Start Code for Combination Macro [13]
#event  OutOfStuff  "Sorry,  but  you  don't  have
everything  you  need  for  this  recipe  in  your
general inventory."

Sub Main
  :Loop
   :ClearCursor
    /if (${Cursor.ID}) {
     /autoinventory
     /goto :ClearCursor
    }
  /notify TradeskillWnd CombineButton leftmouseup
  /doevents
  /goto :Loop
/return

Sub Event_OutOfStuff
   /endmacro
/return
```

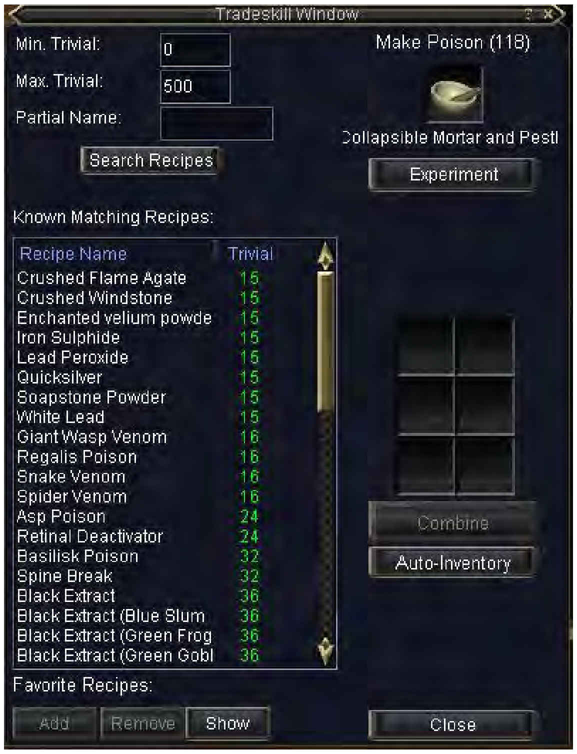
Since this is a very simple macro, it does not account for many things, such as the absence of the "Combine" button, or a certain combination to be selected. Figure 2 shows the layout of this window, including the list of combinations available (on the left), the list of items needed for the combination (middle, on the right. It is empty because no item from the list is selected) with the "Combine" button right below. The value listed in the "Trivial" column describes the difficulty in creating an item, the lower the number, the easier it is to not only create, but create with more success as your skill (located in the upper right, next to the name of the trade-skill being used) increases. The screenshot was provided by the author on a "thief" class character, who specializes in making poisons.

Macroquest also can add in customized plug-ins to do other things that make use of the information stored on the client side. These plug-ins can perform tasks anywhere from providing a ShowEQ-like map on the in-game mapping system, to the ability to "warp" from one side of the map to the other, an ability that normal characters do not possess.

All these ways of cheating lead up to the question of how to deal with them. There are a few ways that are currently being implemented, and a few that have been implemented and failed. The next section will talk about past, current, and future methods of detection.

## 4. CHEATING DETECTION

One method of detection depends on the non-cheating users of the game community. Any member of the gaming community can create a "petition" in which they describe the actions of someone who they believe is not following the rules, and it will be reviewed by a game master, or GM. Due to the low GM to player ratio [15] and the amount of petitions for other matters that need to be dealt with, it takes considerable time for a GM to answer a cheating petition. This is the main way of reporting cheating in Everquest, and according to George Scotto, head of the Everquest customer service department, has increased in efficiency since its creation [16].



Min. Trivial: 0
Max. Trivial: 500
Partial Name:

| Recipe Name | Trivial |
|---|---|
| Crushed Flame Agate | 15 |
| Crushed Windstone | 15 |
| Enchanted velium powde | 15 |
| Iron Sulphide | 15 |
| Lead Peroxide | 15 |
| Quicksilver | 15 |
| Soapstone Powder | 15 |
| White Lead | 15 |
| Giant Wasp Venom | 16 |
| Regalis Poison | 16 |
| Snake Venom | 16 |
| Spider Venom | 16 |
| Asp Poison | 24 |
| Retinal Deactivator | 24 |
| Basilisk Poison | 32 |
| Spine Break | 32 |
| Black Extract | 36 |
| Black Extract (Blue Slum | 36 |
| Black Extract (Green Frog | 36 |
| Black Extract (Green Gobl | 36 |

Figure 2. Tradeskill Window [14].

A recent article on the official Everquest site titled "A Day in the Life of a Game Master" mentions the use of third-party programs to help automate game play for people who play two characters at the same time, a style of play called "boxing". The article states that automating one character with a third-party program while the other character is being played by a person is acceptable [17].

Another method of detection takes place on the servers. To combat against people cheating, the servers keep logs of everything that happens, and analyze them for suspicious behavior. Such behavior would include "warping" that was talked about in Section 3. Since the server would keep track of player locations, it would be possible to calculate the distance moved in a certain amount of time. If the distance is greater than the maximum distance a player could realistically cover, it would flag the character as a possible cheater. If a character gets flagged for this behavior repeatedly, a GM would investigate and take appropriate action if necessary.

The last method of detection is scanning the memory of the client computer to check for third-party cheat programs. Everquest does not use this method of detection, but a newer game, World of Warcraft, does implement this method of detection [18]. Players of World of Warcraft have mixed thoughts about this method of detection. Some have said it is good for the game. It catches cheaters and makes the game fair for all to play. Others say it is spy-ware because it looks at more information than it needs to. Greg Hoglund posted that he noticed the program used to scan memory, called The Warden, was looking at "email addresses of people I was communicating with on MSN, the URL of several websites that I had open at the time, and the names of all my running programs, including those that were minimized or in the toolbar" [19]. He, along with many others, believes that this is a massive invasion of privacy.

A recent court case between Blizzard (the company behind World of Warcraft) and MDY Industries (the creators of a third-party cheat program for World of Warcraft called WoWGlider [20]) states that having one program (cheat program) read the memory locations that are in use by another program (the game) is copyright infringement. The case is currently in court, so the outcome of the case is not yet known [21].

## 5. CHEATING PREVENTION

One method of prevention is the obfuscation of memory locations [22]. With the Everquest example, the use of Macroquest depends on the unchanging memory address locations to run properly. To prevent Macroquest from working all the time, Everquest would have to change the memory addresses of all the internal data every time it was run, which would require a large change in the internal structure of the game. This change, however, would prevent Macroquest, in its current state, from working. The way Macroquest reads memory would have to change dynamically, and would possibly be left in a non-working state for quite some time. Doing this for an older game like Everquest is not very likely, so this method of prevention would have to be implemented in new games that are in early development phases.

Another method of prevention is packet encryption. Everquest implements a basic encryption scheme for packets, but third-party programs easily bypass this. They bypass it by locating the encryption key in local memory and using it to decrypt the packets that are sent to the client. In 2002, the makers of Everquest tried to implement a stronger encryption, but it lead to more overhead and created more network latency [8]. Stronger encryption leads to more latency, but weaker encryption is easier to break. In either case, the client computer has the necessary information to decrypt the packets stored locally.

The last method of prevention that will be discussed is to move away from the client-server model to a more server-based model, where most of the important information would be stored only on the server. Since online games are very bandwidth and memory intensive, it would be a tradeoff to make a more secure game, but have less features and robust graphics than the games currently on the market.

## 6. CONCLUSION

It would take lots of time and resources for older games to implement new ways of prevention. For new games that are in development, the implementation of dynamically-changing obfuscated memory locations would be less of a challenge. Packet encryption adds too much of an overhead to online games, making it an unsuitable solution to the problem. Besides being unsuitable, the encryption key would reside on the client machine, making it easier to locate. A move from the client-server model to a more server-based model would create games that could not handle the amount of data needed for online games. These would have to sacrifice graphics and playability for security.

The current state of the gaming industry can not get rid of cheaters entirely. The best way of dealing with them is by using the community of non-cheaters to report suspicious behavior, and analyzing server logs to check for suspicious activity.

These concepts do not apply only to gaming software. Due to the need to have a program on a client's computer, it is easy for a client to change things in the program. By changing values in the registry, or changing the executables, clients can bypass restricted access programs like trial version or demo version software. New developments in programs that run online could take the place of client-based software. It could be possible that programs which run solely over the internet would be less vulnerable to the same types of exploits as their client-based counterparts.

## 7. WORKS CITED

[1] MMOG Growth Chart. http://www.mmogchart.com. Last Updated June 29th, 2006. Accessed on February 11th, 2007.

[2] Everquest. Official Everquest site. http://eqplayers.station.sony.com. Accessed on March 25th, 2007.

[3] Yan, Jeff and Brian Randell. "Cheating and fairness: A systematic classification of cheating in online games", Proceedings of 4th ACM SIGCOMM workshop on Network and system support for games. New York, NY: ACM Press. October 2005.

[4] Kuo, Andy. "A (very) Brief History of Cheating". <http://shl.stanford.edu/Game_archive/StudentPapers/BySubject/A-I/C/Cheating/Kuo_Andy.pdf>. STS 145 Papers Collection, March 2001.

[5] ShowEQ. Official ShowEQ site. http://www.showeq.net. Accessed on February 11th, 2007.

[6] ShowEQ Image. http://www.breneware.com/paseq/showeq.jpg. Accessed on March 25th 2007.

[7] Caltagirone, Sergio and Matthew Keys, Bryan Schief, Mary Jane Willshire. "Architecture for a Massively Multiplayer Online Role Playing Game Engine". Journal of Computing Sciences in Colleges, Volume 18 , Issue 2, Pages 105-116. USA: Consortium for Computing Sciences in Colleges. December 2002.

[8] Malda, Rob. "EverQuest/Sony Fights Code Wars With Latest Expansion". http://features.slashdot.org/features/02/12/01/1558220.shtml?tid=127. Posted on December 1st, 2002. Accessed on March 25th, 2007.

[9] Macroquest. Official Macroquest site. http://www.macroquest2.com. Accessed on February 11th, 2007.

[10] Macroquest: Macroquest Slash Commands. <http://www.macroquest2.com/includes/wassup/manual.php#macslash>. Accessed on March 25th, 2007.

[11] Macroquest: Macroquest Macro Reference. http://www.macroquest2.com/wiki/index.php/Macro_Reference. Accessed on March 25th, 2007.

[12] Macroquest: Macros. http://www.macroquest2.com/wiki/index.php/MacroQuest2:Macros. Accessed on March 25th, 2007.

[13] "Meatball", "Vanilla" Combine macro. http://www.macroquest2.com/phpBB2/viewtopic.php?t=7101. Posted on March 9th, 2004. Accessed April 10th, 2007.

[14] Everquest Screenshot of Tradeskill Window. Screenshot Taken by Kevin Warns, April 10th, 2007.

[15] Tychsen, Anders and Michael Hitchens, Thea Brolund, Manolya Kavakli. "The Game Master", ACM International Conference Proceeding Series; Vol. 123, Pg. 215-222. Sydney, Australia: Creativity & Cognition Studios Press. November 2005.

[16] Scotto, George. Interview with Stratics.com. http://www.stratics.com/content/interviews/eq/interviews/eqcsinterview.php. Accessed on March 25th, 2007.

[17] Julie, the Community Relations Intern. EQPlayers.com Article "A Day in the Life of a Game Master". http://eqplayers.station.sony.com/news_article.vm?id=50399. Posted April 2nd, 2007. Accessed on April 10th, 2007.

[18] Ward, Mark. "Warcraft game maker in spying row". http://news.bbc.co.uk/1/hi/technology/4385050.stm. October 31st, 2005. Accessed on March 25th, 2007.

[19] Hoglund, Greg. "4.5 Million copies of EULA-compliant software". http://www.rootkit.com/blog.php?newsid=358. Posted on October 5th, 2005. Accessed on March 25th, 2007.

[20] WoWGlider. WoWGlider Official Home page. http://www.wowglider.com/. Accessed on April 10th, 2007.

[21] MDY Industries, LLC v. Blizzard Entertainment, Inc and Vivendi Games, Inc CV 06-2555 PHX DGC. United States District Court: District of Arizona. http://www.theimageplace.net/uploads/7b600767fb.pdf. Accessed on April 10, 2007.

[22] Bhatkar, Sandeep and Daniel C. DuVarney and R. Sekar. "Address Obfuscation: an Efficient Approach to Combat a Broad Range of Memory Error Exploits". 12th USENIX Security Symposium, Washington, DC, August 2003.

# Network Throughput Analysis with Electromagnetic Interference

Christopher G. Popp
Department of Computer Science,
Winona State University
Winona, MN 55987
cgpopp5419@winona.edu

## ABSTRACT

There are often many obstacles when planning and implementing a design for a computer network. One of the more mysterious and poorly quantified aspects of network design is dealing with potential network instability caused by electromagnetic interference, or EMI. There are guidelines to follow for crossing electrical wiring or fluorescent lighting, but why those guidelines are necessary is not always clear. This paper covers experiments that will quantify the effects of EMI on the throughput of both wired and wireless networks. By exploring EMI of varying intensity and frequency, we show the impact that results on a typical network.

## General Terms

Performance, Experimentation, Measurement, Reliability.

## Keywords

EMI, EMF, Networks, Magnetic Field, Interference.

## 1. INTRODUCTION

Electromagnetic fields (EMF) induce interference in cabling. The interference is known as electromagnetic interference, or EMI. This impacts the quality of a computer network, which depends on the cabling for data transfer. The correlation between EMF intensity and frequency to network quality degradation can be mysterious. The amount of previous research widely available in this area is relatively limited. From a previous study by Faber et al. [2], the point emphasized was the lack of quantitative results relating to the effects of EMI on network traffic. The study goes on to show minimal effects from the EMI emitted by common appliances.

The results from the study were relatively limited in that the study only covered a 100BASE-TX network. Also, the previous study took the approach of maintaining 30% network utilization and checking for packet errors. The research we conducted includes 100 Mbps, and 1000 Mbps wired networks. Each wired network is tested with Cat 5e and Cat 6 UTP cabling. Additionally, we explore the effect of EMI on an 802.11g wireless network.

Although Cat 5e cabling is currently more prevalent than Cat 6, Cat 6 cabling has a greater number of twists that allow for it to reduce interference and provide a higher level of reliability [6]. We explore both to compare the effects of EMI on current networks as well as the networks of the future.

The experiment was conducted by measuring the maximum throughput for each type of network without EMI. We then introduced various levels of EMI and performed the same throughput benchmark. In each case, we also track the number of packets that needed to be retransmitted during transfer. Next we analyzed the levels of EMI necessary to negatively impact the network in a significant manner. We compared the levels of EMI with different sources that may be encountered while setting up a network in a real environment. Our final conclusions show the importance of considering EMI when planning a network infrastructure.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

*Proceedings of the 7th Winona Computer Science Undergraduate Research Seminar,* April 19, 2007, Winona, MN, US.

## 2. QUESTIONS AND HYPOTHESES

The paper explores the overall impact of EMI on a wired and wireless network connection. The experiment was conducted with a few hypotheses in mind. First, we suspected that Cat 5e cabling will be more affected than Cat 6. Second, the impact of EMI was suspected to be greater for 1000 Mbps Ethernet compared to 100 Mbps. And lastly, we expected to find that wireless networks are more susceptible to performance drops relative to wired networks.

## 3. METHODS

There are three main aspects to our methodology: Setting up the wired and wireless networks, generating the EMF, and finally conducting, and analyzing the actual network tests.

### 3.1 Network Configurations

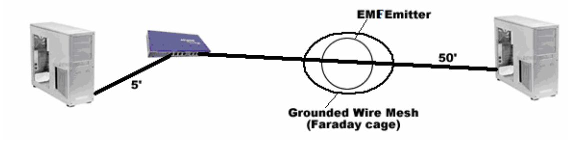


**Figure 1.** Wired Test Infrastructure

The wired networks are setup with two workstations, and a switch. The 100 Mbps and 1000 Mbps networks differ in the speeds supported by the workstation network cards, and switch. There is 5' of network cable between the first workstation and switch, and 50' between the second workstation and switch. The experiment is done with both Cat 5e and Cat 6 cabling. In the middle of the 50' segment of cable we place our EMF emitter. The emitter and cable are surrounded by a grounded wire mesh that acts as a Faraday cage to prevent the EMF from introducing EMI at locations other than the cable. The setup is as shown in Figure 1. It is also important to note the operating frequency of the wired networks. Both the 100 Mbps and 1000 Mbps networks operate at 125 MHz [1, 4, 5].

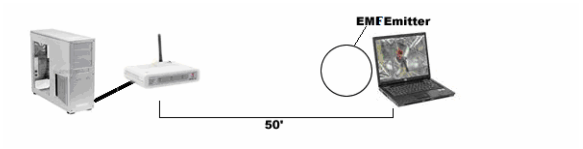


**Figure 2.** Wireless Test Infrastructure

The 802.11g wireless network consists of a laptop with a PCMCIA 802.11g network card, an 802.11g wireless access point, and a 100 Mbps wired connection between the access point and a workstation. The laptop is placed 50' away from the access point. This can be seen in Figure 2. The EMF emitter is placed close to the laptop, and the strength of the EMF measured near the PCMCIA 802.11g card exterior.

## 3.2 Generating the EMF

Generating the EMF is done by using existing household appliances. We are able to generate different frequency EMFs by choosing appliances with different frequencies, for instance a microwave operates at about 2450 MHz. We can also vary the intensity of the field by adjusting the placement of the emitting appliance [7].

In order to determine the intensity and frequency of the EMF, we use Vernier magnetic field sensors as shown in Figure 3. These are capable of measuring the EMF along one axis. An EMF may have components in the X, Y and Z axis [7]. We use three sensors; one oriented for each axis. We calculate the magnitude of the EMF based on combined values of each sensor.

The sensors also allow for us to view the frequency of the EMF. With the ability to know the intensity and frequency, we are able to find appliances (and placement distances) that produce the desired EMF.

## 3.3 Result Collection and Analysis

We perform the tests between computers using FTP transfers. The overall transfer time required for a given file size allows for the calculation of average throughput for the duration of the file transfer. We also keep track of the number of TCP segments sent, and retransmitted during the FTP transfer. We do this using the *netstat –s –p tcp* command under Windows XP. We record the number of segments sent/retransmitted before and after the transfer, and calculate the difference. The software aspects of the experiment are similar for the different network types. The only variance will occur in the file size used for FTP.

**Figure 3.** Vernier Magnetic Field Sensor [3]

We then analyze these results and calculate the percentage drop in throughput compared to our reference throughput with no EMF. We also compare the change in segment retransmission rates, based on the percentage of segments that needed to be retransmitted during the transfer.

This analysis allows for exploration into the overall impact that EMI has on a given network type, and relative comparisons between the network types. Comparisons of the impacts of EMI on throughput at different frequencies, but equal intensity, will offer insight on specific appliances and sources that may cause trouble in real world situations.

## 4. RESULTS

The results can be seen in Table 1 below. The intensity of the magnetic field is given in milliTeslas, and the frequency in Hertz. A reference test is given for each network type, shown in the row with zero intensity for the EMF.

## 5. ANALYSIS

The only network which appears to be adversely impacted is the 1000 Mbps wired network over Cat 5e. Even in this setup the difference between the control case and each case with EMI is relatively minimal. This does however support our initial hypotheses that Cat 5e would be more affected than Cat 6, and that 1000 Mbps networks are more susceptible than 100 Mbps networks.

Because of the fact that both the wired and wireless networks operate at much higher frequencies than the EMF generated, it is possible that the affects would be more prevalent in very high frequency situations.

## 6. CONCLUSION AND FUTURE WORK

We began the study with the question of how electromagnetic fields will impact common types of computer networks. We noted this is an important area of study due to the lack of quantified research, and hence a bit of mystery for wiring standards that mention EMI.

In the end, we came to the realization that lower frequency EMFs do not have a large impact on the wired and wireless networks tested. We then went on to note that since the operating frequency of these networks is much higher than those tested, the effects may be different, and that would be a good extension to our research.

There are a number of additional areas that could be expanded upon. We only tested UTP cabling, so it would be worthwhile to see how resilient properly grounded STP cabling would fair in more extreme conditions. In addition, it may be interesting to see how other network types compare to those we tested. Perhaps different protocols form their data units in such a way that the malformations caused by EMI have a different impact.

## 7. ACKNOWLEDGEMENTS

I would like to thank Dr. Gerald Cichanowski for providing critical suggestions and guidance that helped to improve the research. Second, I would like to thank Dr. Andrew Ferstl for providing access to measuring equipment for the project. And lastly, thanks to Chris Lohfink, for initially suggesting the area of research, and providing continual interest in the progress.

## 8. REFERENCE

[1] Buis, Paul. "Common 100 Mbps Hardware Variations." Sept. 1996. Ball State U. Accessed 7 Apr. 2007 http://www.cs.bsu.edu/homepages/peb/cs637/ethernet/100mbps.htm.

[2] Faber, Robert Y., and Valerie A. Rybinski. UTP Cabling and the Effects of EMI. Siemon. 1997. Accessed 14 Feb. 2007 http://www.siemon.com/us/white_papers/97-10-02-presentation.asp.

[3] "Magnetic Field Sensor." Vernier Software & Technology. Accessed 4 Apr. 2007 http://www.vernier.com/probes/mg-bta.html .

[4] Marsh, David. "Category 6 Cable—Gigabit Ethernet Over Copper." 9 Dec. 1999. Accessed 4 Apr. 2007 http://www.edn.com/article/CA46370.html .

[5] Rao, Sailesh K., and Juan M. Jover. "Gigabit Ethernet Over Copper." Aug. 1998. Accessed 4 Apr. 2007 http://www.commsdesign.com/main/9808fe1.htm .

[6] Shimonski, Robert J., Richard T. Steiner, and Sean M. Sheedy. Network Cabling Illuminated. Sudbury: Jones and Bartlett, 2006.

[7] Young, Hugh D., and Roger A. Freedman. University Physics. 11th ed. San Francisco: Addison Wesley, 2004.

**Table 1.** Test Results

| Network Type | File Size (MB) | Time (s) | Bandwidth (MB/s) | Intensity (mT) | Frequency (Hz) | Segments Sent | Segments Retransmitted |
|---|---|---|---|---|---|---|---|
| 100 Mbps(Cat 5e) | 897 | 86.51 | 10.4 | 0 | n/a | 1751773 | 0 |
| 100 Mbps(Cat 5e) | 897 | 89.4 | 10.0 | 4.6 | 60.03 | 1751740 | 0 |
| 100 Mbps(Cat 5e) | 897 | 86.1 | 10.4 | 3.5 | 2000 | 1751982 | 0 |
| 100 Mbps(Cat 5e) | 897 | 88.4 | 10.1 | 2.3 | 65000 | 1751899 | 0 |
| 100 Mbps(Cat 5e) | 897 | 88.8 | 10.1 | 2 | 20000 | 1751867 | 0 |
| 100 Mbps(Cat 6) | 897 | 88.0 | 10.2 | 0 | n/a | 1752877 | 0 |
| 100 Mbps(Cat 6) | 897 | 87.1 | 10.3 | 4.6 | 60.03 | 1751787 | 0 |
| 100 Mbps(Cat 6) | 897 | 87.2 | 10.3 | 3.5 | 2000 | 1751793 | 0 |
| 100 Mbps(Cat 6) | 897 | 89.5 | 10.0 | 2.3 | 65000 | 1751938 | 2 |
| 100 Mbps(Cat 6) | 897 | 87.7 | 10.2 | 2 | 20000 | 1751847 | 0 |
| 1000 Mbps(Cat 5e) | 1270 | 71.49 | 17.8 | 0 | n/a | 166043 | 0 |
| 1000 Mbps(Cat 5e) | 1270 | 86.1 | 14.8 | 4.6 | 60.03 | 166894 | 0 |
| 1000 Mbps(Cat 5e) | 1270 | 81.83 | 15.5 | 3.5 | 2000 | 165793 | 0 |
| 1000 Mbps(Cat 5e) | 1270 | 77.28 | 16.4 | 2.3 | 65000 | 167320 | 0 |
| 1000 Mbps(Cat 5e) | 1270 | 78.9 | 16.1 | 2 | 20000 | 165877 | 0 |
| 1000 Mbps(Cat 6) | 1270 | 60.5 | 21.0 | 0 | n/a | 167964 | 0 |
| 1000 Mbps(Cat 6) | 1270 | 58.5 | 21.7 | 4.6 | 60.03 | 166788 | 0 |
| 1000 Mbps(Cat 6) | 1270 | 66.8 | 19.0 | 3.5 | 2000 | 166228 | 0 |
| 1000 Mbps(Cat 6) | 1270 | 56.2 | 22.6 | 2.3 | 65000 | 165495 | 0 |
| 1000 Mbps(Cat 6) | 1270 | 56.7 | 22.4 | 2 | 20000 | 165012 | 0 |
| 802.11g Wireless | 230 | 100.2 | 2.3 | 0 | n/a | 449210 | 25 |
| 802.11g Wireless | 230 | 97.5 | 2.4 | 4.6 | 60.03 | 449178 | 22 |
| 802.11g Wireless | 230 | 103.3 | 2.2 | 3.5 | 2000 | 449225 | 25 |
| 802.11g Wireless | 230 | 102.7 | 2.2 | 2.3 | 65000 | 448596 | 24 |
| 802.11g Wireless | 230 | 104.3 | 2.2 | 2 | 20000 | 449873 | 28 |